SOFTWARE DIDÁCTICO APRENDA ÁLGEBRA BOOLEANA

SARA TERESA CARRASCAL ALMENTERO

DIANYS PATRICIA MORENO CASTAÑEZ

Trabajo de grado presentado
para optar el título de
Ingeniero de Sistemas

Director
MOISÉS QUINTANA ALVAREZ

CORPORACION UNIVERSITARIA TECNOLOGICA DE BOLIVAR

FACULTAD DE INGENIERIA DE SISTEMAS

CARTAGENA D.T Y C.

2001

SOFTWARE DIDÁCTICO APRENDA ÁLGEBRA BOOLEANA

SARA TERESA CARRASCAL ALMENTERO

DIANYS PATRICIA MORENO CASTAÑEZ

Trabajo de Grado presentado
para optar el título de
Ingeniero de Sistemas

Director

MOISÉS QUINTANA ALVAREZ

CORPORACION UNIVERSITARIA TECNOLÓGICA DE BOLIVAR

FACULTAD DE INGENIERÍA DE SISTEMAS

CARTAGENA D.T Y C.

2001

Cartagena de Indias, Enero 15 de 2000

Señores:
CORPORACION UNIVERSITARIA TECNOLOGICA DE BOLIVAR
Comité de Evaluación de Proyectos
Facultad de Ingeniería de Sistemas.
La Ciudad.

Respetados señores:

Por medio de la presente nos permitimos presentar para su estudio y calificación el proyecto de grado "SOFTWARE DIDÁCTICO APRENDA ÁLGEBRA BOOLEANA", con el propósito de obtener el título de Ingeniero de Sistemas.

Cordialmente,

SARA CARRASCAL **DIANYS MORENO**

Cartagena de Indias, Enero 15 de 2000

Señores:
CORPORACION UNIVERSITARIA TECNOLOGICA DE BOLIVAR
Comité de Evaluación de Proyectos
Facultad de Ingeniería de Sistemas.
La Ciudad.

Respetados señores:

Por medio de la presente nos permitimos presentar para su estudio y calificación el proyecto de grado "SOFTWARE DIDÁCTICO APRENDA ÁLGEBRA BOOLEANA", con el propósito de obtener el título de Ingeniero de Sistemas.

Cordialmente,

_____

MOISÉS QUINTANA

Nota de aceptación

_____

Presidente del Jurado

Jurado

Jurado

Cartagena de Indias D.T. Y C., Enero 15 de 2001

Esta observación debe quedar impresa en parte visible del proyecto”.

A Dios y a la Virgen. A mis padres Antonio y Carmen. A mis hermanos Yuri, Toño, Gabriela, Carmen y Mayo. A mis sobrinos Carlos David y Antonio Juan.

Sara

A Jehová Dios. A mis queridos padres Haroldo y Abadys gestores de este logro. A mis hermanos Dairo y Abadys. A mi sobrino Steven David. A mis hijos Lewis y Dylan. A mi amado esposo Lewis.

Dianys

AGRADECIMIENTOS

Los autores expresan sus agradecimientos a:

Dios por darnos la fuerza y la sabiduría para seguir adelante.

Nuestros hermanos y demás familiares por su estímulo para seguir adelante y lograr nuestro objetivo.

Moisés Quintana y a su esposa Claudia Prieto por su asesoría y apoyo incondicional.

Gonzalo Garzón y Juan Carlos Mantilla por su paciencia y colaboración.

Nuestros compañeros Yasmina Gómez, Leandro Contreras, Maria Cristina Mancuso, Olivia Narvaez, Clara Colón, quienes aportaron un granito de arena en la elaboración de este proyecto.

Johan Maldonado, Hermelinda Hoyos, Julio Carrascal, Feligno Mejia, Euclides Castilla, Ilsa de Castilla, Yaninis Fernández por su apoyo incondicional.

CONTENIDO

LISTA DE CUADROS

LISTA DE TABLAS

LISTA FIGURAS

RESUMEN

El trabajo de Grado Software Didáctico Aprenda Álgebra Boleana, está diseñado para utilizarse como soporte educativo en el proceso de enseñanza aprendizaje de un tema base de la ingeniería, el Álgebra de Boole.

En este software se toman en cuenta los modelos apropiados del alumno y del conocimiento para presentar los conceptos adecuadamente. El software le permite al usuario ir construyendo los conceptos del Álgebra de Boole y sus conocimientos, a la vez que los ejercita mediante las evaluaciones y de acuerdo a los resultados obtenidos en estas evaluaciones se le aplican reforzamientos. Además es una herramienta que le puede ayudar a solucionar problemas propuestos por el alumno.

INTRODUCCION

La experiencia indica que cuando los estudiantes pasan del bachillerato a primer semestre en la universidad, llegan con deficiencias en los hábitos de estudio que no permiten la adquisición adecuada del aprendizaje, se presentan problemas a la hora de resolver los ejercicios en casa y de definir los conceptos adquiridos. Surgiendo así la necesidad de contar con una herramienta que apoye el auto estudio de temas básicos, uno de éstos es el Álgebra Booleana, seleccionado para el desarrollo de este proyecto, por sugerencia de profesores y alumnos consultados.

Está herramienta es capaz de asistir al alumno, apoyando las actividades de supervisión del profesor, tanto en la escuela como en forma remota o en casa. Además cuando en el transcurso de la carrera se vuelvan a necesitar los conocimientos del Álgebra Booleana en asignaturas posteriores (como Técnicas Digitales, Arquitectura del Computador y otras) no es necesario que el

profesor dedique mucho tiempo para recordarlo a sus alumnos, pues se puede utilizar esta herramienta para el auto estudio.

Una de las actividades más comunes de un profesor consiste en la asesoría y supervisión del profesor al alumno, directamente sobre el tema que se está considerando. Esta actividad puede considerarse como benéfica, pues permite que el instructor se adapte a las necesidades de cada alumno, detectando los puntos que no se han entendido así como los problemas a los que el alumno personalmente se enfrenta en el aprendizaje del Álgebra Booleana. Además la experiencia y observación que el instructor encuentra en este proceso de asesoría, le permiten resaltar los puntos importantes y le proporcionan una retroalimentación con la que puede controlar mejor el desarrollo de su clase, adaptándola a las circunstancias.

Esta actividad puede verse limitada por tiempo, espacio y recursos. Entre los problemas y limitaciones que se observan al analizar la actividad de asesoría descrita anteriormente, están las siguientes:

- La atención del profesor está dividida y no siempre basta el tiempo que dedica a un alumno en particular a resolver todas sus dudas; si bien el alumno aprecia y se beneficia por la asesoría, no a todos les basta la información recibida de ella

y quieren buscar otros medios que le ayuden a satisfacer sus dudas en el momento que no tengan disponible a su profesor. Inclusive si un alumno resuelve todas sus dudas, con más tiempo el profesor tendría la posibilidad de ampliar el conocimiento que le transmite. Este problema se agudiza cuando el grupo es grande.

- Cuando el alumno estudia por su cuenta, o no tiene al profesor disponible para dicha asesoría en un tema como el Álgebra Booleana donde surgen muchas dudas a la hora de resolver los ejercicios en casa.

- Cuando el alumno realiza sus tareas normalmente no es durante la hora de clase. Por ello, no siempre puede resolver sus dudas en el momento que surgen, sino que suele existir un lapso de tiempo desperdiciado entre el momento que surge la duda y cuando el profesor está disponible para resolverla. Por otro lado, si el alumno no registra la duda, puede más tarde olvidarla y dejarla pendiente, con lo que no se consiguen completamente los objetivos del proceso de enseñanza- aprendizaje.

Entre las consecuencias del problema que hemos planteado se pueden presentar las siguientes:

- Las fallas en el aprendizaje del Álgebra Booleana, inciden en el aprendizaje de temas posteriores ligados con el primero, tema base en la ingeniería.

- El profesor emplea gran parte de su tiempo resolviendo repetitivamente las mismas dudas a distintos alumnos, durante distintos tiempos, algunas de ellas ya mencionadas en clase; tiempo que podría ser aprovechado para ampliar los conocimientos que se cubren, agregar mayor número de ejemplos, y otras diversas actividades académicas.

El objeto del presente trabajo es: Implementar un software didáctico que apoye la enseñanza de un tema base en la ingeniería, como es el Álgebra de Boole. Brindándole al estudiante una herramienta que le permita reforzar los conceptos adquiridos en clase y además lo ayude a solucionar problemas de circuitos lógicos que tienen como base este tema.

En esta investigación se organiza adecuadamente el proceso de enseñanza aprendizaje y se hace uso de las vías y métodos de la enseñanza asistida por computador para lograr cumplir con los siguientes criterios:

- Exponer en el software didáctico de una forma clara y amena que el estudiante pueda asimilar los conceptos básicos del Álgebra Booleana y su aplicación en circuitos lógicos como son: variables booleanas, tablas de verdad, operaciones lógicas, teoremas del Alebra de Boole, compuertas lógicas, métodos de simplificación.

- Ayudar al usuario a diseñar un circuito lógico a partir de una expresión booleana.

- Guiar al usuario a determinar la salida de un circuito lógico.

- Ayudarle a simplificar o llevar a una forma más simple pero equivalente una expresión booleana, utilizando métodos de simplificación.

- Brindar una ayuda de ejercicios modelos resueltos.

- Evaluar el progreso del alumno al utilizar el software proponiéndole ejercicios para que los resuelva

Este trabajo de grado le facilita al usuario(alumno) el aprendizaje del Álgebra Booleana, contando con modelos apropiados del

alumno y del conocimiento, para lograr utilizarla como herramienta para el análisis y diseño de sistemas digitales. Se enfoca al proceso de entrenador y tutor, que apoya al alumno después de haber pasado la etapa de Receptor, le permite al usuario observar su avance por medio de evaluaciones interactivas, además asesorarlo en la solución de problemas del Álgebra Booleana. Este software puede ser utilizado como apoyo a los docentes que dicten este tema o en futuras asignaturas para recordar fundamentos del tema.

A medida que el usuario interactúa con el **MEC(Material Educativo Computarizado)** en las diferentes lecciones que visite se va almacena toda la información referente a él en la Base de Datos.

El tipo de estudio que se lleva a cabo en la realización del proyecto es una investigación descriptiva aplicada, pues se dirige a la aplicación inmediata.

Los métodos de investigación empleados son: entrevistas, análisis bibliográfico y de otras fuentes, método inductivo-deductivo.

1. ANÁLISIS BIBLIOGRAFICO Y FUNDAMENTACION TEORICA

1.1 ALGEBRA BOOLEANA.

El Álgebra Booleana es una herramienta para el análisis y diseño de sistemas digitales(lógicos). Las compuertas lógicas, son los circuitos lógicos más fundamentales y su operación puede describirse mediante el uso del Álgebra Booleana, estas compuertas lógicas pueden combinarse para producir circuitos lógicos y describirse, analizarse, mediante el Álgebra Booleana. En el Álgebra Booleana no hay fracciones, decimales, números negativos, raíces cuadradas, raíces cúbicas etc.

1.1.1 Operaciones Básicas del Álgebra Booleana. Solo existen tres operaciones básicas: **OR, AND Y NOT,** llamadas operaciones lógicas.

OPERACIÓN OR: La operación OR se representa con el signo +, cuyas reglas se dan en la tabla de verdad, Tabla 1.

De la Tabla 1 se concluye:

- La operación OR produce un resultado de uno, cuando cualquiera de las variables de entrada es uno.

- La operación OR genera un resultado de cero, solo cuando todas las variables de entrada son cero.

- Las dos conclusiones anteriores se cumplen también para más de dos variables de entrada.

Tabla 1. Tabla de verdad que define la operación OR.

A	B	X=A+B
0	0	0
0	1	1
1	0	1
1	1	1

OPERACIÓN AND: La operación AND se representa por el símbolo(●) se puede omitir es decir, X=A●B es lo mismo que X=AB, las reglas de esta operación se encuentran en la Tabla 2.

Tabla 2. Tabla de verdad que define la operación AND.

A	B	X=AB
0	0	0
0	1	0
1	0	0
1	1	1

De la Tabla 2 se concluye:

- La operación AND se ejecuta igualmente que la multiplicación ordinaria de 1 y 0.
- Una salida igual a 1 ocurre solo en el caso en que todas las entradas sean 1.
- La salida es cero en cualquier caso en que una o más entradas sean cero.
- Estas reglas se cumplen para dos o más entradas.

OPERACIÓN NOT: Esta operación se efectúa con una sola variable de entrada, es decir si la variable A se somete a esta operación

NOT, el resultado X puede expresarse como: X=-A. Se utilizó el símbolo (-) delante de la variable deseada para representar una inversión. La tabla de verdad para la operación NOT sobre la variable A se muestra en la Tabla 3.

Tabla 3. Tabla de Verdad para la operación NOT.

A	X=-A
0	1
1	0

OPERACIÓN NOR: La compuerta NOR de dos entradas se representa de la siguiente forma: X=-(A+B), las reglas de esta operación se encuentran en la Tabla 4.

De la Tabla 4 se concluye:

- La operación NOR es el resultado de la operación OR seguida de un INVERSOR.

- Una salida igual a 1 ocurre solo en el caso en que todas las entradas sean 0.

- La salida es uno en cualquier caso en que una o más entradas sean uno.

- Estas reglas se cumplen para dos o más entradas.

Tabla 4. Tabla de verdad que define la operación NOR.

A	B	X=-(A+ B)
0	0	1
0	1	0
1	0	0
1	1	0

OPERACIÓN NAND: La compuerta NAND de dos entradas se representa de la siguiente forma: X=-(AB), las reglas de esta operación se encuentran en la tabla de la Tabla 5.

De la tabla 5 se concluye:

- La operación NAND es el resultado de la operación AND seguida de un INVERSOR.

- Una salida igual a 0 ocurre solo en el caso en que todas las entradas sean 1.

- La salida es uno en cualquier caso en que una o más entradas sean cero.

- Estas reglas se cumplen para dos o más entradas.

Tabla 5. Tabla de verdad que define la operación NAND.

A	B	X=-(AB)
0	0	1
0	1	1
1	0	1
1	1	0

1.1.2 Forma de Suma de Productos. Los métodos de diseño y simplificación de circuitos lógicos que utilizaremos necesitan que la expresión lógica esté en forma de suma de productos.

Ejemplo: ABC + ABC

Cada una de estas expresiones de suma de productos consta de dos o más términos AND (productos) que se operan con OR (sumas).

1.1.3 Simplificación de Circuitos Lógicos. Después que se obtiene la expresión del circuito lógico, este se puede reducir a una forma más simple que contenga menos variables en uno o más términos. La nueva expresión puede utilizarse entonces para implantar un circuito que sea equivalente al original pero que contenga menos compuertas y conexiones.

Los métodos de simplificación de productos son los siguientes:

1.1.3.1 Simplificación Algebraica. Este método utiliza los teoremas del álgebra booleana, que son los siguientes:

1 X• 0= 0.

2 X•1=X.

3 X • X =X.

4 X - X = 0.

5 $X + 0 = X$.

6 $X + 1 = 1$.

7 $X + X = X$.

8 $X + -X = 1$

9 $X + Y = Y + X$.

10 $X \bullet Y = Y \bullet X$.

11 $X + (Y + Z) = (X + Y) + Z = X + Y + Z$.

12 $X(YZ) = (XY)Z = XYZ$.

13.a $X(Y+Z) = XY + XZ$.

13. b $(W + X)(Y + Z) = WY + XY + WZ + XZ$.

14. $X + XY = X$.

15. $X + -XY = X + Y$.

LEYES DE DEMORGAN:

16. -(X + Y) = -X • -Y.

17. -(X • Y) = -X + -Y.

El método se basa en:

- La expresión original se pone en forma de suma de productos mediante la aplicación repetida de los teoremas de DeMorgan y de la multiplicación de términos.

- Una vez se encuentra en esta forma, los términos del producto se verifican para ver si hay factores comunes y se realiza la factorización siempre que sea posible. Con suerte, la factorización da como resultado la eliminación de uno o más términos.

1.1.3.2 Método del Mapa de Karnaugh. El método de Karnaugh es un método gráfico que se utiliza para simplificar una ecuación lógica o para convertir la tabla de verdad de un circuito lógico correspondiente en un proceso simple y ordenado. Aunque un mapa de Karnaugh se puede utilizar para resolver problemas con cualquier número de variables de entrada, su utilidad práctica se

limita a seis variables. Los problemas con cinco y seis entradas son demasiado complicados.

FORMATO DEL MAPA DE KARNAUGH: El mapa de Karnaugh, al igual que una tabla de verdad, es un medio para demostrar la relación entre las entradas lógicas y la salida que se busca.

El mapa de Karnaugh se forma de la siguiente manera:

1. La tabla de verdad da el valor de la salida X para cada combinación de valores de entrada. El mapa de Karnaugh proporciona la misma información en un formato diferente.

2. Los cuadrados del mapa de Karnaugh se marcan de forma que los cuadrados horizontalmente adyacentes, solo difieran en una variable. Por ejemplo, el cuadrado superior de la izquierda del mapa de cuatro variables ABCD, en tanto que el cuadrado que se encuentra a la derecha es ABCD (sólo la variable D es diferente). De la misma manera los cuadrados verticalmente adyacentes difieren sólo en una variable. Por ejemplo, el cuadrado superior izquierdo es ABCD, en tanto que el que se encuentra debajo de él es ABCD (sólo la variable B es diferente). Cada cuadrado del renglón superior se considera adyacente al correspondiente cuadrado del renglón inferior. Así mismo, los cuadrados del

extremo izquierdo son adyacentes a los del extremo derecho. A fin de que los cuadrados que son adyacentes tanto vertical como horizontalmente difieran en una sola variable, el marcador de arriba hacia abajo deba hacerse en el orden indicado, AB, AB, AB, AB. Esto también es válido para el marcado de izquierda a derecha.

3. Una vez que el mapa de Karnaugh se ha llenado con ceros y unos, la expresión de suma de productos para la salida X se puede obtener operando con OR aquellos que contienen un 1. Ej. : Los siguientes son los mapas de Karnaugh para dos, tres y cuatro variables y su tabla de verdad correspondiente:

a. Mapa de Karnaugh y tabla de verdad para dos variables:

Tabla 6. Tabla de verdad para dos variables

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

La Tabla 6 corresponde a la tabla de verdad de la expresión:

X= -A-B + AB

EL mapa de Karnaugh se observa en la Figura 1.

	-B	B
-A	1	0
A	0	1

Figura 1. Mapa de Karnaugh para dos variables.

b. Mapa de Karnaugh y tabla de verdad para tres variables:

Tabla 7. Tabla de verdad para tres variables.

A	B	C	X
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

La Tabla 7 representa la tabla de verdad de la expresión:

$$X= -A-B-C+ -A-BC + -AB-C + AB-C$$

El mapa de Karnaugh se observa en la Figura 2.

	-C	C
-A-B	1	1
-AB	1	0
AB	1	0
A-B	0	0

Figura 2. Mapa de Karnaugh para tres variables.

c. Mapa de Karnaugh y tabla de verdad para cuatro variables. La Tabla 7 corresponde a la tabla de verdad de la expresión:

$$X= -A-B-CD+ -AB-CD+ AB-CD+ ABCD$$

Tabla 8. Tabla de verdad para cuatro variables.

A	B	C	D	X
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

El mapa de Karnaugh para cuatro variables se observa en la Figura 3.

	-C-D	-CD	CD	C-D
-A-B	0	1	0	0
-AB	0	1	0	0
AB	0	1	1	0
A-B	0	0	0	0

Figura 3. Mapa de Karnaugh para cuatro variables.

El método de Karnaugh requiere de un proceso de simplificación después que se obtiene el mapa de Karnaugh. Este proceso consiste en:

Existe una regla de agrupamiento de cualquier tamaño, la cual dice: *cuando una variable aparece en forma complementada y no complementada dentro de un agrupamiento esa variable es eliminada de la expresión. Las variables que son iguales en todos los cuadrados del agrupamiento deben aparecer en la expresión final.*

Un agrupamiento mayor de 1 elimina más variables. Es decir, un agrupamiento de dos elimina una variable, uno de cuatro elimina dos y uno de ocho elimina tres. Este principio se usa para obtener

una expresión lógica simplificada a partir de un mapa de Karnaugh que contenga cualquier combinación de 1 y 0.

Las etapas para simplificar una expresión booleana son las siguientes:

1. Construir un mapa de Karnaugh y colocar unos en aquellos cuadrados correspondientes a los unos de la tabla de verdad. Colocar ceros en los otros cuadrados.

2. Examinar el mapa para ver si hay unos adyacentes y repetir aquellos unos que no sean adyacentes a cualesquiera otros unos. A éstos se les llama unos aislados.

3. A continuación, buscar aquellos unos que sean adyacentes sólo a otro uno. Repetir cualquier par que contenga a dicho 1.

4. Agrupar cualquier octeto aún si algunos de los unos ya se han repetido.

5. Agrupar cualquier cuádruplo que contenga uno o más unos que no se hayan repetido.

6. Agrupar los pares que sean necesarios para incluir los unos que no se hayan repetido, asegurándose de utilizar el mínimo número de agrupamientos.

7. Formar la suma OR de todos los términos generados por cada agrupamiento.

1.1.3.3 Método De Quine McCluskey: El método de McCluskey es un método tabular que se utiliza para simplificar una ecuación lógica o para convertir la tabla de verdad de un circuito lógico correspondiente en un proceso simple y ordenado.

El siguiente Ejercicio indica los pasos a seguir para reducir una expresión booleana utilizando este método.

F = -a-b-c-d + -a-bc-d + -abc-d + -abcd + a-b-c-d + a-b-cd + ab-cd + abcd

Se creas una tabla de Agrupamiento Base en donde las columnas son:

1. Columna 1. Cada uno de los términos de la función.

2. Columna 2. El número binario correspondiente al termino, si es negada la variable se coloca 0 si no esta negada se pone 1.

3. Columna 3. El número decimal correspondiente al binario obtenido.

4. Columna 4. El índice que corresponde al número de unos que tiene el número binario obtenido.

La tabla de Agrupamiento base para el ejercicio propuesto es la siguiente (Cuadro 1).

Cuadro 1. Tabla de Agrupamiento Base.

-a-b-c-d	0000	0	Índice 0
-a-bc-d a-b-c-d	0010 1000	2 8	Índice 1
-abc-d a-b-cd	0110 1001	6 9	Índice 2
-abcd ab-cd	0111 1101	7 13	Índice 3
abcd	1111	15	Índice 4

Ahora se procede a comparar cada uno de los términos de la columna uno de igual índice con los del siguiente índice, donde se encuentre una sola diferencia se toma ese termino y se forma la Tabla de agrupamiento de Primer Orden(Cuadro 2), donde la primera columna es el resultado de la comparación colocándose – en el lugar donde estaba la diferencia, la segunda columna son los números decimales correspondientes a los términos en los que se halló una diferencia en la tabla del paso anterior y la columna tres al índice, números de unos que tiene el término de la columna 1. .

Cuadro 2. Tabla de Agrupamiento de Primer Orden.

00-0 -000	(0,2) (0,8)	Índice 0
0-10 100-	(2,6) (8,9)	Índice 1
011- 1-01	(6,7) (9,13)	Índice 2
-111 11-1	(7,15) (13,15)	Índice 3

Se analiza la tabla de Agrupamiento de primer orden de igual forma que se hizo con la de Agrupamiento Base y se saca una Tabla de Agrupamiento de Segundo Orden así sucesivamente hasta que no se puedan hacer más agrupamientos en el ejercicio no es posible realizar más agrupamientos se pasa entonces a obtener la tabla reductora final(Cuadro 3).

Cuadro 3. Tabla Reductora Final.

a b c d	**0**	**2**	**6**	**7**	**8**	**9**	**13**	**15**
00-0	X	X						
-000	X				X			
0-10		X	X					
100-					X	X		
011-			X	X				
1-01						X	X	
-111				X				X
11-1							X	X

Se trata seguidamente de obtener una ecuación con los valores literales equivalentes a las combinaciones binarias, de forma que con el menor número posible de ellas se representen todos los

términos de la tabla de agrupamiento base. Existen dos soluciones posibles.

F= -a-b-d + a-b-c + -abc + abd ó **F**= -b-c-d –ac-d +a-cd + bcd

En este Trabajo de Grado uno de los objetivos propuestos es conseguir implementar un método de simplificación que permita llevar una expresión booleana a una forma equivalente sencilla. Inicialmente se piensa en el método de simplificación con el mapa de Karnaugh, un método bastante utilizado pero con ciertas limitaciones, puesto que su manejo para ecuaciones con más de cuatro variables resulta muy complejo; sus pasos implican al usuario en muchas decisiones que de no ser acertadas no le permiten conseguir su objetivo: una ecuación simplificada equivalente. Al ir avanzando en nuestra investigación encontramos otros métodos de simplificación como el de Quine McCluskey, un método de simplificación que le permite al usuario el manejo de más variables sin demasiadas complicaciones, además el usuario goza de menos probabilidades de equivocarse en uno de sus pasos, que dependen menos de las decisiones de éste y más del método, permitiendo de igual forma obtener su objetivo: una ecuación simplificada. Se realiza una consulta a profesores del área y concuerdan en que es una mejor opción incluirlo en el

software didáctico y optamos por presentar esta alternativa a nuestros usuarios.

1.2 INFORMÁTICA Y EDUCACIÓN.

En la época actual la informática puede jugar un papel muy importante en la educación, puesto que se puede utilizar como herramienta pedagógica. Gracias a la tecnología han aparecido nuevas áreas del saber. Algunas como la informática educativa (CBT o Computer Based Training), se vinculan directamente con la educación. La informática educativa consiste en el uso de estas tecnologías para educar a los alumnos de las instituciones educativas, para los programas de educación a distancia y de autoaprendizaje y por el entrenamiento del personal de las empresas e instituciones que los requieran.

Otra posibilidad que brinda la informática educativa es el uso de multimedia. El concepto general de multimedia refiere a la posibilidad de combinar en un texto audiovisual e interactivo, conjunto de palabras escritas y habladas, gráficos, animaciones, video digital y video analógico, de manera que las acciones del usuario – lector influyan en el desarrollo del texto multimedia.

Entre los distintos tipos de aplicaciones educativas se tienen, en primer lugar, el software instruccional que consiste en la automatización de ejercicios o exámenes de distintas materias o en tutoriales que enseñan procesos o proporcionan datos enciclopédicos. En segundo lugar, los simuladores o modeladores de situaciones reales a las que el estudiante se debe enfrentar, por ejemplo, simuladores del uso de maquinaria industrial, de las variaciones de precios de acuerdo a las fluctuaciones de la oferta y la demanda o de las diferentes velocidades de caída de un cuerpo en distintas gravedades. Finalmente, los juegos en general, pueden ser de dos tipos: los juegos educativos y los juegos para pasar el rato. En el siguiente ítem se amplían más estos conceptos.

1.3 MODALIDADES DEL APRENDIZAJE ASISTIDO POR COMPUTADOR.

La modalidad Tutorial. Trata de presentar un material en la pantalla de la computadora y eventualmente hace preguntas sobre dicho material. En las versiones avanzadas de tutoriales, las preguntas se convierten en evaluaciones más o menos complicadas dependiendo de las cuales aparece una retroalimentación diferente y se toma un camino alterno para continuar con la presentación del material.

La modalidad de ejercitación y práctica trata de que los usuarios adquieran una habilidad sobre algo realizando ejercicios únicamente, es decir no se propone una teoría o explicación sobre el contenido de lo que se está haciendo, bajo él supuesto que esto ya se conozca (o se dio en clase) y que con esta modalidad lo que se hace es la labor de reforzamiento de lo aprendido y el adquirir o mejorar una habilidad (por ejemplo en la resolución de ejercicios aritméticos).

Las modalidades de juegos son aquellos programas en que emplean algún recurso divertido y cuya finalidad escondida es que el usuario o jugador aprenda algo, practique algo o desarrolle alguna habilidad. Para lograr jugar o participar en el mismo hay que conocer, practicar, o desarrollar conocimientos, habilidades etc.

Sin duda alguna ésta es la modalidad más difícil de describir y de realizar, puesto que se trabaja en dos planos simultáneamente el del entrenamiento y el del aprendizaje.

Los juegos pueden ser tan simples como uno de mesa, o de adivinanza, hasta autenticas aventuras gráficas en las que el sujeto es participante de un cuento fantástico. Sin embargo dada la competencia con auténticos juegos a través de los productos

empacados como Nintendo, Sega y otros juegos de vídeo, que tienden a ser cada vez más sofisticados, los juegos educativos no se pueden quedar atrás, so-pena de ser etiquetados como “aburridos”. La parte didáctica del juego, puede estar en el contenido, en la habilidad para manejar el juego o en el conocimiento asociado a las variantes del juego.

Frecuentemente se asocia también a una modalidad diferente al descubrimiento, aunque en el fondo no es diferente, ya que está comprendido en las modalidades anteriores. Se entiende por descubrimiento al conjunto de programas que permiten que el usuario aprenda algo por inferencia, deducción, etc. descubriéndolo por sí mismo y no presentado directamente. En la mayoría de las simulaciones o juegos se aprende de esta manera. El objetivo de esta manera es facilitar la creatividad del individuo, facilitar la capacidad de generación y de entender-haciendo. Actualmente también se habla como objetivo del descubrimiento el estimular el “pensamiento crítico”. Así por ejemplo, si se dan los instrumentos adecuados y un mínimo de técnica se podría en algunos casos, que el usuario dedujera alguna de las leyes de un fenómeno (físico, biológico, social etc.), donde se presenta la problemática de manera muy didáctica.

1.4 TEORÍAS DEL APRENDIZAJE.

Las teorías contemporáneas del aprendizaje o que se han desarrollado en el siglo **XX**, las podemos dividir en dos grandes familias:

* Las teorías cognoscitivas: De las que uno de los aportes principales a software educativos, es que ofrece pautas especificas y estrategias didácticas para su construcción los psicólogos cognoscitivos al presentar la información insisten en que se realicen asociaciones globales que les permita procesar la información por su cuenta.

* Las teorías del condicionamiento estímulo – respuesta: El hombre desde sus comienzos buscó explicación a los fenómenos que se sucedían a su alrededor y la primera respuesta que encontró fue dándoles alma y espíritu a todos los objetos, incluyendo las piedras, las rocas, etc.; a esto se le llamo animismo.

A medida que descubrían las causas de los fenómenos naturales, el animismo perdía popularidad y se sustentaba cada vez más la opinión de la casualidad de todos los fenómenos de la naturaleza. Poco a poco se comenzó a dar explicaciones mecánicas a todo lo existente, incluyendo al hombre. De ahí los dichos populares: “sin fuego no hay humo”, “no hay causa sin efecto”.

La psicología mecanicista realista reflejada en el conductismo, trató de seguir los anteriores lineamientos equiparando las causas con los estímulos y el efecto con la respuesta, única manera para que la psicología siguiera siendo científica.

En consecuencia, el hombre no es más que un mecanismo de estímulos y respuestas donde todo lo pueda saber acerca de su conciencia y de sus sentimientos, se obtendrá de las secreciones glandulares y de sus contracciones musculares. Como vemos los realistas científicos, consideran al hombre como una máquina inteligente, bien diseñada que aprende mediante un proceso aditivo de estimulo – respuesta; sus deseos, metas e intenciones no cuentan para nada.

Al abordar la educación, un conductista, considera inevitable controlar la conducta, el aprendizaje, los planes, los objetivos y metas de enseñanza.

Dentro de los modelos del aprendizaje del condicionamiento estímulo – respuesta tiene las siguientes.

1.4.1 El condicionamiento Clásico de Pavlov(1849-1936). En sus experimentaciones iniciales Ivan Petrovich Pavlov, estuvo interesado en el proceso digestivo de los perros y no en el aprendizaje o cualquier forma de proceso mental. Sin embargo, con el progreso de sus investigaciones surgió un modelo de

aprendizaje que ha sido llamado condicionamiento clásico. En el experimento de éste; un perro podrá comenzar a salivar ante el sonido de un tono o la presencia de una luz, si éste tono o luz ha estado apareado previamente con la presentación del alimento al perro.

Cuando se coloca carne en la boca de un perro se produce salivación; el alimento es el estímulo incondicionado y la salivación el reflejo. Después de algún tiempo, se coloca un estímulo arbitrario por ejemplo, una luz o un sonido que se le presenta en la comida. Después de repetir esta experiencia varias veces, el animal segregará saliva al oír el sonido o al ver la luz, sin la presencia del alimento. En este segundo paso del experimento, la saliva es el reflejo condicionado. Algunos psicólogos tienden a utilizar la expresión respuesta condicionada en vez de reflejo condicionado, por considerar que no todo lo condicionado es reflejo.

Como podemos ver, la forma básica del condicionamiento clásico consiste en aparear un estímulo, el cual es originalmente neutral con un estímulo que provoca la respuesta. Después de uno o más apareamientos, el estímulo previamente neutral produce la respuesta estudiada. Cuando esto sucede, decimos que se ha llevado a cabo un condicionamiento clásico.

1.4.2 El conexionismo de Thorndike (1874-1949). El fundamento principal del conexionismo, lo constituye la asociación entre las impresiones sensoriales y los impulsos de la acción; a ésta asociación le dio el nombre de “vehículo” o “conexión”, de ahí se deriva el nombre de “conexionismo”.

Para Thorndike, la forma más característica de aprendizaje es por ensayo y error; más tarde prefirió llamarlo por “selección y conexión”. Para explicar el anterior enunciado, Thorndike supone que el que este aprendiendo se enfrenta una situación problemática en la que tiene que lograr una meta, como por ejemplo, salir de una jaula o alcanzar un alimento. Para solucionar esta situación, selecciona la respuesta adecuada entre cierto número de respuestas o soluciones posibles. Define el ensayo por el número de errores que se cometen o por el tiempo que transcurre antes de elegir la respuesta correcta para alcanzar la meta o solucionar el problema.

El experimento típico por ensayo y error, lo llevó a cabo con un gato hambriento en una jaula que podía abrirse desde el interior al levantar una aldaba para alcanzar la comida que se hallaba afuera; los primeros ensayos estaban caracterizados por innumerables errores: arañazos, mordidas y movimientos inútiles, hasta que

accidentalmente levantaba la aldaba y el animal se encontraba libre para tomar el alimento. Se repetía el experimento y el gato se conducía de la misma forma, pero con la diferencia que disminuían los errores en forma lenta e irregular, hasta que eventualmente el gato aprendía a escapar de manera inmediata, sin actividades al azar. Del anterior experimento, dedujo que el gato no “pesca” realmente la manera de escapar, sino que la aprende grabando respuestas correctas y borrando las incorrectas.

De la conducta cronometrada de los gatos, llegó a concluir que el aprendizaje es un proceso de “introducción” de conexiones por ensayo y error, en el sistema nervioso y que no tiene nada que ver con la comprensión

Los principios del conexionismo han sido expresados en forma de leyes del aprendizaje.

- Ley de la disposición o preparación. Enuncia las circunstancias en las que el sujeto que aprende tiende a sentirse satisfecho o molesto, a aceptar o rechazar. Estas circunstancias pueden ser tres:

Primera, cuando un sujeto está preparado para ejecutar determinada acción, el llevarla a cabo le resulta satisfactorio.

Segunda, cuando un sujeto está preparado para ejecutar determinada acción, el no ejecutarla, le resulta molesto.
Tercera, cuando un sujeto no está preparado para realizar una acción, y es forzado a ejecutarla, le resulta molesto.

- Ley del ejercicio. Según esta ley cuanto más se repita una respuesta inducida por un estímulo más largo será su período de retención.

- Ley del efecto. Está ligada al principio de placer – dolor. Por lo cual, podemos decir que una respuesta se fortalece, cuando produce placer, y se debilita cuando produce dolor. En consecuencia, la ley del efecto hace relación al fortalecimiento o al debilitamiento de una conexión de acuerdo con sus consecuencias. Con esta ley, Thorndike afirmó que las recompensas o éxitos fomentan el aprendizaje de la conducta recompensada; mientras que los castigos a los fracasos, reducen la tendencia repetir la conducta que condujo al castigo, al fracaso o a la molestia.

1.4.3 Condicionamiento contiguo de Edwin Guthrie (1886-1959).. Consiste en que si un estímulo se produce contiguamente a una respuesta, la respuesta es este estímulo, seguirá produciéndose hasta que se condicione a este estímulo otra respuesta. Dicho de

otra manera, los estímulos que se aplican en el momento de una respuesta, al volverse a presentar, provocan la misma respuesta.

El condicionamiento contiguo tiene gran aplicabilidad en el aprendizaje de hábitos. De ahí que su encanto radique en las orientaciones prácticas para el entrenamiento de animales, la crianza de los niños y la pedagogía.

1.4.4 Condicionamiento operante de B.F. Skinner (1904-1974). B.F. Skinner, profesor de la universidad de Harvard, es considerado como la figura más representativa del conductismo actual. Descubrió y desarrolló un modelo o paradigma de aprendizaje, llamado condicionamiento operante. Este lo podemos definir como: Un proceso de aprendizaje que consiste en aumentar o disminuir la probabilidad de la conducta por medio de la aplicación del refuerzo. Refuerzo es cualquier evento que incremente la fuerza de la respuesta.

El condicionamiento operante se interesa en la relación existente entre la conducta de un organismo y el medio en que habita.

A esta clase de condicionamiento se le llama operante, porque las respuestas que emite el sujeto pueden verse como operaciones sobre el ambiente para alcanzar el refuerzo o la recompensa.

Ejemplo: todos los movimientos que hace una rata para alcanzar el alimento, o todos los esfuerzos que un alumno realiza por alcanzar una buena calificación, o todas las “pilatunas” del bebe por lograr la sonrisa de su madre.

El condicionamiento operante, tiene una condición única que:

El estímulo reforzante no se produce antes de la respuesta deseada, sino después de ella.

- Skinner y sus seguidores, consideran que casi toda la conducta humana es producto del reforzamiento operante, ya que mediante el refuerzo que reciben las personas aprenden a caminar. A hablar, a comportarse de ésta manera no de otra. De ahí que toda persona y organismo pueda ser reforzada a “elegir cualquier estado de cosas, mediante el condicionamiento operante”.

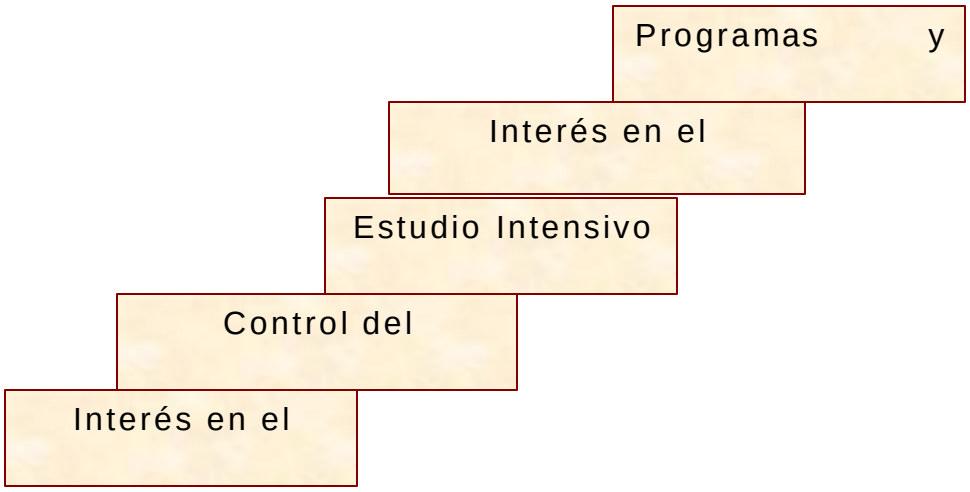


Figura 4. Características del Condicionamiento Operante

En el condicionamiento operante, el maestro es el arquitecto y constructor de la conducta de los alumnos, sus valores, sus metas y los objetivos del aprendizaje, ya que niega la intencionalidad de la conducta inteligente de los sujetos.

La psicología de Skinner, implica un determinismo estricto, el cual significa que la conducta tiene una causa y que solo puede producir el comportamiento observado (efecto). El determinismo lleva implícito que el ambiente determina a un individuo, donde su conducta es un diseño de su propia cultura. Como partidario del

realismo científico, considera que las leyes de la psicología son tan definidas como las de cualquier otra ciencia.

Esta es la esencia del condicionamiento operante: Las respuestas correctas se refuerzan; las incorrectas se ignoran o se castigan. Este reforzamiento puede ser de diversos tipos como:

1. Reforzamiento Positivo y Negativo. Un reforzador positivo añade una recompensa a cierta situación, por ejemplo comida. El reforzamiento negativo resulta eficaz porque substrae algo desagradable de la situación; el reforzamiento constituye en eliminar el estímulo. Por ejemplo, a un perro se le enseña a abrir las puertas no solo para obtener alimentos (reforzamiento positivo) sino también para evitar descargas eléctricas o ruidos molestos (reforzamiento negativo).

2. Reforzamiento primario y secundario. Un reforzador primario es aquel que resulta grato por sí mismo, sin nexo alguno con otro (el agua, el sexo, el alimento). Un reforzador secundario es aquel cuyo valor ha de aprenderse mediante una asociación con otros reforzadores. El dinero es un reforzador secundario, aunque no pasa de ser un simple papel o metal, gracias a su vinculación con el alimento, la ropa y otros reforzadores primarios, se convierte en un premio poderoso.

3. Reforzamiento Demorado. Hay un intervalo de tiempo entre el momento en que se obtiene el comportamiento deseado y el momento en que se aplica el reforzamiento. Cuanto mayor es este intervalo menor eficacia tendrá el reforzamiento.

¿El castigo es un reforzamiento? El castigo es un proceso diferente del reforzamiento, ya que mientras el castigo consiste en la presentación de un estímulo positivo, el reforzamiento incluye, tanto la presentación de un reforzador positivo, como la eliminación de uno negativo.

Un castigo puede consistir, por ejemplo, en privar a Juan Carlos de ver televisión durante una semana por haber perdido una materia; este castigo no implica ningún reforzamiento.

Dicho de otra manera, cuando un estímulo participa en el fortalecimiento de una respuesta, habrá reforzamiento; cuando se presenta o se retira un estímulo tratando de debilitar una respuesta, habrá castigo.

1.4.5 Asimilación De Piaget. Como se piensa que una de las medidas más prometedoras para el mejoramiento del aprendizaje consiste en el perfeccionamiento de los materiales didácticos,

debido al alto grado en que estos materiales facilitan el aprendizaje significativo se utiliza el método de EAC (Enseñanza Asistida por Computador) enfocado hacia los entrenadores - tutores, porque este es de gran utilidad ya que el alumno desempeña una función esencial en la determinación de la tasa del nuevo aprendizaje. Además se basa en el proceso de “Asimilación” de Piaget". Este concibe la asimilación en función de progresiones del desarrollo y su teoría trata básicamente de:

- El desarrollo del pensamiento como proceso opuesto a la compresión.

- Tiende a identificar las operaciones del pensamiento con las operaciones de la lógica.

- Atribuye al pensamiento la cualidad de acción implícita.

Además de esto, para Piaget existen dos tipos de aprendizaje:

- El aprendizaje que incluye la adquisición del organismo de nuevas respuestas a situaciones específicas, pero sin que el organismo tenga que entender necesariamente el razonamiento que está detrás del aprendizaje, ni que generalice lo aprendido a otras situaciones.

- El otro consiste en adquirir una nueva estructura de operaciones mentales en virtud del proceso de equilibrio. Este aprendizaje es estable y duradero comparado con el primer tipo que es transitorio, y conduce a la generalización basado en la comprensión. Supone cierto nivel de estructura cognoscitiva que depende a su vez del desarrollo del organismo.

1.4.6 Teoría de Vigosky: Aprendizaje Significativo. Trata sobre la relación entre el medio ambiente y la influencia o importancia que adquiere el conocimiento. En este se intenta relacionar la información nueva con lo que ya se sabe y por consiguiente se le da sentido. Se dice que en este se hace una construcción coherente y comprensiva del contenido en lugar de sólo memorizarlo. El aprendizaje significativo es retenido más tiempo que el aprendizaje memorístico. También es mucho más eficiente porque unos cuantos principios generales pueden acomodar una gran cantidad de aplicaciones específicas.

1.5 DIDÁCTICA.

1.5.1 Concepto de Didáctica. La didáctica es ciencia y arte de enseñar. Es ciencia en cuanto investiga y experimenta nuevas técnicas de enseñanza, teniendo como base, principalmente, la

biología, la psicología, la sociología y la filosofía. Es arte, cuando establece normas de acción o sugiere formas del comportamiento didáctico basándose en los datos científicos y empíricos de la educación.

La didáctica está representada por el conjunto de técnicas a través de las cuales se realiza la enseñanza. La didáctica es una disciplina orientada en mayor grado hacia la práctica, toda vez que su objetivo primordial es *orientar la enseñanza.* A su vez la enseñanza no es más que la dirección del aprendizaje. La didáctica está constituida por un conjunto de procedimientos y normas destinadas a dirigir el aprendizaje de la manera más eficiente que sea posible.

1.5.2 Elementos Didácticos. La didáctica tiene que considerar seis elementos fundamentales que son, con referencia a su campo de actividades: el alumno, los objetivos, el profesor, la materia, las técnicas de enseñanza y el medio geográfico, económico, cultural y social.

El Alumno. Es quien aprende, aquel por quien y para quien existe la escuela (entendiéndose por ésta como un ente donde se imparte educación); debe haber una adaptación recíproca entre escuela y alumno.

Los Objetivos. Toda acción didáctica supone objetivos. La escuela no tendría razón de ser si no tuviese en cuenta la conducción del alumno hacia determinadas metas, tales como: modificación el comportamiento, adquisición de conocimientos, desenvolvimiento de la personalidad, orientación profesional, etc. En consecuencia la escuela existe para llevar el alumno hacia el logro de determinados objetivos.

El Profesor (es quien imparte la enseñanza). Es el orientador de la enseñanza. Debe ser fuente de estímulos que lleva el alumno a reaccionar para que se cumpla el proceso de aprendizaje.

La Materia. Es el contenido de la enseñanza. A través de ella serán alcanzados los objetivos de la escuela.

Métodos y Técnicas de Enseñanza. Tanto los métodos como las técnicas son fundamentales en la enseñanza y deben estar, lo más próximo que sea posible, a la manera de aprender los alumnos.

Medio Geográfico, Económico, Cultural y Social. Es indispensable para que la acción didáctica se lleve a cabo en forma ajustada y eficiente, tomar en consideración el medio donde se está enseñando.

1.6 MOTIVACIÓN DEL APRENDIZAJE.

Es el proceso que provoca cierto comportamiento, mantiene la actividad o la modifica. Motivar es predisponer al alumno hacia lo que se quiere enseñar; es llevarlo a participar activamente en los trabajos escolares. Así, motivar es conducir al alumno a que se empeñe en aprender, sea por ensayo y error, por imitación o por reflexión.

La motivación consiste en el intento de proporcionar a los alumnos una situación que los induzca a un esfuerzo intencional, a una actividad orientada hacia determinados resultados queridos y comprendidos. Así, motivar es predisponer a los alumnos a que aprendan y, consecuentemente, realicen un esfuerzo para alcanzar los objetivos previamente establecidos.

Los propósitos de la motivación consisten en despertar el interés, estimular el deseo de aprender y dirigir los esfuerzos para alcanzar metas definidas.

La motivación, en el proceso de aprendizaje, puede provocar los siguientes pasos:

1. Se crea una situación de necesidad (motivación), estableciéndose simultáneamente una tensión.

2. Se vislumbra un objetivo capaz de satisfacer esa necesidad.

3. Se inicia el esfuerzo o la acción para solucionar la dificultad, de una manera desordenada u ordenada.

4. Dada la solución, o satisfecha la necesidad, disminuye la tensión y el individuo retiene (aprehende) la dirección o la forma de comportamiento, para actuar de una manera más o menos similar en situaciones parecidas.

1.6.1 Tipos de Motivación. Hay dos modalidades de llevar al alumno a estudiar: induciéndose hacia la aceptación y reconocimiento de la necesidad de estudiar, o bien obligándolo mediante la coacción. Sobre esta base, la motivación puede ser positiva o negativa.

Motivación Positiva. Cuando procura llevar al alumno a estudiar, teniendo en cuenta el significado que guarda la materia para la vida del alumno, el aliento, el incentivo y el estímulo amigable. La motivación positiva a su vez puede ser intrínseca o extrínseca.

a) Motivación Positiva Intrínseca. Cuando el alumno es llevado a estudiar por el interés que le despierta la propia materia, es decir, porque “gusta de la materia”.

b) Motivación Positiva Extrínseca. Cuando el estímulo no guarda relación directa con l a signatura desarrollada o cuando el motivo de aplicación al estudio, por parte del alumno, no es la materia en sí.

Motivación Negativa. S la que consiste en llevar al alumno a estudiar por medio de amenazas, reprensiones, y también castigos. El estudio se lleva a cabo bajo el imperio de la coacción.

Desde el punto de vista didáctico la motivación puede ser clasificada como inicial y de desenvolvimiento.

Motivación Inicial. Procura disponer a los alumnos para ejecutar los trabajos que serán realizados.

1.6.2 Motivación de Desenvolvimiento o Incentivación. Es la que se emplea durante el desarrollo del tema; debe ser planeada de modo que se renueve constantemente el interés de los alumnos y, así mismo, aprovechar las situaciones de cada momento para

reavivar dicho interés por lo que está siendo estudiado. De ese modo procura conservar el impulso y la disposición iniciales.

1.6.3 Fuentes y Técnicas De Motivación. Constituyen elementos, factores o circunstancias que despiertan en el alumno algún motivo o actitudes favorables para ciertas actividades, porque aguzan sus necesidades.

Algunas fuentes de motivación pueden, asimismo – y según la manera de encararlas – funcionar como técnicas de motivación.
Las principales fuentes son:

1. Necesidades del educando, que pueden ser de naturaleza biológica, psicológica o social.

2. Curiosidad natural del ser humano.

3. Vida social: acontecimiento de la actualidad.

4. Ambiente escolar adecuado.

5. Actividad lúdicra.

6. Personalidad del profesor.

7. Aprobación social.

8. Competición.

9. Deseo de evitar fracasos y puniciones.

10. Necesidades económicas.

11. Necesidades de conocimiento,

12. Afán de distinguirse.

13. Deseo de ser eficiente.

14. Tendencia a la experimentación.

15. Aspiraciones.

Son innumerables las técnicas de motivación existentes. Estas técnicas procuran suscitar motivos y activar posibilidades internas.

Algunas de estas técnicas son:

1. Correlación con lo real. Con ella el profesor procura establecer relación entre lo que está enseñando y la realidad circundante, con las experiencias de vida del alumno o con hechos de la actualidad.

2. Victoria Inicial. El Alumno es llevado a responder preguntas relativamente fáciles pero “pomposas”, presentadas con aspectos de difíciles. Naturalmente el alumno responderá con exactitud y, entusiasmado con su éxito, prosigue con atención los trabajos de la clase.

3. Participación del alumno. Mediante interrogatorios y de situaciones problemáticas interesantes, hace que los alumnos tomen parte en el trabajo escolar. Su preocupación debe ser de sustraer al alumno de la situación de simple espectador, para transformarlo en participante.

4. Auto superación. El profesor incentiva al alumno para que mejore su actuación. El alumno se le indica la marcha de su aprendizaje en diversos momentos, inducido a superarse,

5. Experimentación. Una tendencia común a todos es la de hacer algo. Esta tendencia puede ser explorada en la enseñanza de todas las disciplinas. Deben planearse actividades de

realización por parte del alumno, llevándolo a actuar física e intelectualmente.

6. Conocimiento preciso de los objetivos a alcanzar. Una excelente técnica de motivación es la que da a conocer, con toda claridad, los objetivos de la unidad y de la lección. Se trabaja con más ánimo y conciencia de responsabilidad cuando se sabe a qué están destinados los esfuerzos y se conoce el punto de llegada.

7. Reducción de los factores negativos y aumento de los positivos. Esta debe se la norma general de la motivación del aprendizaje. Es preciso reducir al mínimo las condiciones desfavorables para el trabajo escolar como: reprimendas, críticas exageradas, etc. Corresponde por el contrario, aumentar las condiciones favorables, de manea que el alumno se sienta a gusto acentuando los elogios.

8. Espíritu Lúbrico. Es propio de la naturaleza humana el interés por el juego, por la diversión, por la recreación. Siempre que fuese posible sería interesante desarrollar las clases a través de actividades lúdricas o en un clima de recreación.

9. Aplicar las técnicas o conocimientos adquiridos. La disciplina que transcurre solamente en el capo teórico tiene pocas probabilidades de motivar a los alumnos. Es recomendable que la teoría sea seguida de aplicaciones prácticas o, aún mejor, que la teoría sea extraída de la práctica.

10. Trabajos Graduados. Otra forma de incentivar consiste en presentar tareas adecuadas y graduadas según las dificultades, partiendo de su nivel de preparación y de capacidad. Mediante esta técnica se logrará, indefectiblemente, buen éxito, y este éxito será un excelente motivador. No obstante, estos loros exitosos no deben ser fáciles, ni facilitados, sino que deben ser alcanzados mediante el empeño del estudiante por cumplir eficientemente su tarea.

Motivos Estimulantes. Motivo no aprendido, como la curiosidad o la actividad, la exploración, manipulación y contacto, los cuales nos impulsan a investigar; estos dependen más de los estímulos externos que de los estados fisiológicos internos.

La exploración y curiosidad parecen ser motivos activados por lo nuevo y lo desconocido, sin que tengan otra meta específica de la de “descubrir algo”. La curiosidad es un motivo que nos impulsa a

investigar estímulos desconocidos, y conforme aprendemos a explorar nuestro ambiente, nuestra curiosidad se vuelve más ambiciosa. ¿Cómo funciona el televisor? ¿Para qué sirve aquella herramienta? Cuando hacemos la primera pregunta no se pretende en absoluto abrir un taller de reparación de televisores, y la herramienta cuya finalidad nos interesa, quizá nunca la usemos. En estos casos sólo queremos simplemente conocer las cosas.

A diferencia de la curiosidad y exploración, está la manipulación que se dirige a un objeto concreto que hemos de tocar, manejar, sentir y jugar para sentirnos satisfechos. La manipulación es un motivo que parecer ser exclusivo de los primates provistos de dedos ágiles en las manos y en los pies. La manipulación se relaciona más que todo con la necesidad de tener conocimiento de las cosas.

1.7 PROGRAMACIÓN ORIENTADA A OBJETOS (POO).

Es el método más reciente y actual de desarrollo de software que existe. Este consiste en que los datos y procedimientos se combinan en objetos.

OBJETO: Es un elemento que contiene las características de una entidad (sus datos) y su comportamiento (sus procedimientos).

Combinando estas características y comportamientos, un objeto conoce cualquier cosa que necesite para hacer su trabajo. El objeto dado contiene las declaraciones tanto de los datos como de los procedimientos. También, se puede decir que un objeto es una variable de un tipo definido por el usuario.

MÉTODOS: Son los procedimientos y las funciones declarados en el objeto.

HERENCIA: Propiedad que tienen los objetos de heredar los datos y procedimientos de otros objetos. Un objeto puede heredar un conjunto general de propiedades a las que pueden añadir características que son específicamente suyas. La herencia es importante porque permite que un objeto soporte el concepto de clasificación jerárquica. Además, hace posible describir un objeto estableciendo la clase general (o clases) a las que pertenecer, junto con aquellas características que le hacen único.

ENCAPSULACIÓN: Consiste en la creación de objetos que funcionan como entidades completas. Una de las reglas de la encapsulación es que el programador nunca necesita acceder directamente a los campos de datos de un objeto. Su principal ventaja es que al limitar el acceso sólo a los métodos, se pueden cambiar libremente los campos sin que se produzcan efectos

laterales. Una de sus desventajas es que es código completamente concentrado y parece ser mucho más engorroso que el simple acceso a un campo. Además, se puede decir que la encapsulación es el mecanismo que agrupa el código y los datos que maneja y los mantiene protegidos frente a cualquier interferencia y mal uso.

POLIMORFISMO: Es la cualidad que permite que un nombre se utilice para dos o más propósitos relacionados, pero técnicamente diferentes. El propósito del polimorfismo aplicado a la POO es permitir poder usar un nombre para especificar una clase general de acciones. Dentro de una clase general de acciones, la acción específica a aplicar está determinada por el tipo de dato.

De forma general, el concepto de polimorfismo es la idea de " una interfaz, múltiples métodos ". Esto quiere decir que es posible diseñar una interfaz genérica para un grupo de actividades relacionadas. Sin embargo, la acción específica ejecutada depende de los datos. La ventaja del polimorfismo es que ayuda a reducir la complejidad permitiendo que la misma interfaz se utilice para especificar una clase general de acción.

1.8 LAS VARIABLES.

En el ambiente educativo informático se deben definir variables desde diferentes puntos de vista, que nos permitan comparar el álgebra Booleana con los componentes de un MEC(MATERIAL EDUCATIVO COMPUTARIZADO).

1.8.1 Clase y Operacionalización de las Variables.

1.8.1.1 Variables Educativas.

Unidad Temática. Una guía que procure establecer el objetivo terminal según los conceptos, habilidades que se desean fortalecer en el usuario.

Sistema de Motivación. Motivación que incentive el aprendizaje del tema en mención.

Sistema de Refuerzo. Indicadores de recompensa y castigo de acuerdo a los logros obtenidos.

Sistema Evaluativo. Forma de medir los logros obtenidos en el proceso de enseñanza aprendizaje con el uso del MEC.

1.8.1.2 Variables De Computación.

Funciones de apoyo al usuario. Que alternativas se ofrecen al proceso de enseñanza aprendizaje: secuencias, reinicios, abandonos.

Estructura Lógica y de Datos. Que almacenamientos ejecuta el MEC, según los movimientos del usuario.

1.8.1.3 Variables de Comunicación.

Dispositivos de entrada y Salida. Qué periféricos se utilizarán para la toma de decisiones: mouse, teclado, pantalla sensible; y cuales para comunicar al usuario mensajes, resultados.

Interfaz de Entrada y Salida. Formas de Intercambiar mensajes el usuario con el MEC, de acuerdo a las interacciones que se estén realizando.

1.8.1.4 Variables de Entorno.

Area de contenido. Tratar los conceptos del Álgebra Booleana y aplicaciones de ésta, como por ejemplo a la Lógica Digital.

Limitaciones y recursos para los usuarios. Restricciones del software a sus usuarios.

Infraestructura. Establecer el equipo requerido, sistema operativo, y otros, para lograr un óptimo desempeño del MEC.

2. DISEÑO DEL MEC (Material Educativo Computarizado)

2.1 ENTORNO PARA EL DISEÑO DEL MEC

2.1.1 Población Objetivo. Este Modelo Educativo Computarizado va dirigido a estudiantes de Ingeniería que cursan asignaturas donde toman el tema de Álgebra de Boole como son: Lógica, Arquitectura del Computador, Técnicas Digitales, Circuitos Digitales, o que ya han visto pero que necesitan refrescar los conocimientos adquiridos para aplicaciones de este tema.

2.2 DISEÑO EDUCATIVO

2.2.1 ¿Por qué el MEC es tipo tutor – Entrenador? Se presenta la información sobre el Álgebra de Boole en la pantalla del computador de una forma dinámica, agradable y a la vez se va evaluando el aprendizaje del usuario por medio de preguntas o ejercicios sobre dicho material. La evaluación tiene un grado de complejidad que va aumentando de acuerdo a los resultados que se obtiene de esta.

2.2.2 Presentación Mr. Boole. Para hacer el software lo más amigable posible, lo cual es uno de los objetivos de un software didáctico, que mejor opción que encontrar en el un amigo, ese es Mr. Boole, quien aparece a continuación.

Figura 5. Mr. Boole

Buscando que Mr. Boole se identifique con la mayoría de los usuarios no se le da forma de animal definido, tal vez se parezca a una ardilla, un ratón u otro. Deseando inspirar al usuario: animo, seguridad, capacidad de realizar lo que se propone; se le da un aspecto que lo hace lucir estudioso, a la vez descomplicado pero con su propio estilo como el de los jóvenes universitarios a los que va dirigido especialmente este software.

¿Por qué se le da el nombre de Mr Boole? El Nombre de Boole se le otorga en honor al creador del Álgebra Booleana, y el Mister por la nacionalidad inglesa de éste.

2.2.3 Diseño de las Lecciones. Nuestro material didáctico está dividido en lecciones y esta a su vez en módulos. El módulo es la unidad básica de enseñanza o aprendizaje, que tiene sentido para la lección, generalmente alrededor de un sólo concepto. La lección articula un concepto con otros conceptos relacionados alrededor de una temática y puede contener a varios módulos. Finalmente el material, es una organización de enseñanza basada en el tiempo sobre conocimientos que cubren o pretenden cubrir el tema del Álgebra Booleana hasta cierto nivel de profundidad.

Las lecciones con sus módulos correspondientes son:

1. Lección Historia.

1.1 Módulo Biografía de George Boole.

1.2 Módulo Historia del Álgebra de Boole.

1 Lección Teoría Básica del Álgebra de Boole.

2.1 Módulo Variable Booleana.

2.2 Módulo Operaciones del Álgebra Booleana.

2.3 Módulo Tablas de Verdad.

2.4 Módulo Formación de ecuaciones Booleanas.

3. Lección simplificación Algebraica.

4. Lección Simplificación Método de Quine-McCluskey.

4.1 Módulo Expresiones minterms.

4.2 Módulo Tabla de Agrupamientos de orden: Primero. Segundo. Tercero. Etc.

4.3 Módulo de Reducciones.

6. Lección Aplicaciones.

6.1 Módulo Compuertas Lógicas.

6.2 Módulo Representación de Circuitos Digitales.

6.3 Módulo Deducción de ecuaciones Booleanas a partir de circuitos Digitales.

2.2.4 Evaluación y Reforzamiento. Los Ejercicios que se proponen para evaluar el progreso del estudiante en una lección se le asignan niveles de complejidad, se definen cuatro niveles de complejidad con su respectiva puntuación, como se indica en la Tabla 9.

Tabla 9. Puntuaciones por Complejidad

NIVEL	COMPLEJIDAD	PUNTUACIÓN
1	Baja	10
2	Media baja	20
3	Media	30
4	Alta	40

Un usuario nuevo se le inicia con un nivel de dificultad 1. en la tabla Estudiantes de la base de datos Tablas se determina el nivel en que se encuentra el usuario, con esta información se accede a la Tabla Ejercicios Propuestos se selecciona un ejercicio, se verifica en la tabla de la lección que va a ser evaluada que no lo haya realizado el estudiante. Si al ser evaluado se obtiene la respuesta deseada se le aplica un reforzador positivo subiéndolo de nivel y sumándole el valor de los puntos a su puntaje por lección y al total. Los puntos se obtienen para las evaluaciones tipo selección múltiple en su totalidad cuando la respuesta es correcta y cero cuando es incorrecta, y para las evaluaciones en que se califican una serie de pasos el puntaje depende del porcentaje de pasos acertados, el cual debe ser mayor del 60% para subir de nivel. Al estudiante además se le premia con estrellas, por cada veinte puntos que él obtenga en su evaluación consigue una

estrella que lo invita a seguir esforzándose por mejorar siempre y obtener más premios. Si la respuesta es incorrecta se le aplica un reforzador negativo para eliminar la conducta indeseada disminuyendo un 10 % del valor de la puntuación que pudo haber obtenido al puntaje total, se le baja de nivel de complejidad y se le propone resolver un nuevo ejercicio del mismo tema con su nuevo nivel; si su respuesta es correcta se le adiciona a su puntaje el total de la puntuación del ejercicio, si no es correcta se le propone visitar nuevamente la lección o el módulo evaluado. Todas las respuestas son reforzadas sean correctas o incorrectas, es decir, se aplica un reforzamiento continuo.

2.2.5 Ejercicios Resueltos. El usuario puede consultar en la base de datos la tabla Ejercicios_Resueltos que contiene ejercicios modelos del tema que está desarrollando, accediendo a estos desde la opción del menú principal.

2.2.6 Método de Enseñanza. Para dirigir el aprendizaje del estudiante hacia el objetivo propuesto, reforzar sus conocimientos del Álgebra Booleana, se utiliza un conjunto de técnicas lógicamente coordinadas, estas son las que presentamos a continuación.

2.2.6.1 Método Deductivo. Se presentan los conceptos o principios, definiciones o afirmaciones, de las cuales van siendo extraídas las conclusiones y consecuencias o se examinan ejemplos particulares sobre las afirmaciones que son presentadas.

2.2.6.2 Método de Sistematización Semirígida. Los esquemas de las lecciones le ofrecen cierta flexibilidad al estudiante como: poder visitar ejercicios resueltos, la ayuda, visitar otras lecciones en cualquier momento; pero restringe de otras maneras por ejemplo, realizar una segunda evaluación cuando se equivoca.

2.2.6.3 Método Activo. Se tiene en cuenta la participación del estudiante. El método se convierte en un recurso de activación e incentivo para que el estudiante sea quien actúe y realice un auténtico aprendizaje. Se convierte el programa en un orientador, guía, incentivador y no solo en un transmisor del saber.

2.2.6.4 Método Sintético. Se forma de un todo (El conocimiento del Álgebra Booleana), utilizando la unión de varios elementos (Las lecciones y sus respectivos módulos).

2.2.7 Manejo del Color. Se maneja con buen gusto para no agotar, se busca siempre despertar la atención e interés del usuario, ya que el ojo humano es sensible a la iluminación de los colores y

dependiendo de ésta así es el efecto que se puede lograr. Por lo tanto se tiene en cuenta las incidencias psicológicas de los colores para utilizarlos dependiendo de que se quiere despertar en el usuario.

Los colores son eficaces si se ajustan a ciertos criterios:

1. Poco Numerosos. Se necesita más tiempo para decodificarlos el cerebro. Lo ideal es utilizar de 5 a 7, si se desea utilizar más se debe agrandar el tamaño de los objetos.

2. Diferenciarse entre sí.

Los colores poseen sentido y se perciben como lo indica el Cuadro 4.

Cuadro 4. Sentido del Color.

Color	**Sentido**
Rojo	Fuerza vital actividad
Azul	Tranquilizante
Amarillo	Estimulante
Verde	Estabilidad
Morado	Metamorfosis
Gris	Negación
Negro	Negación Absoluta
Blanco	Positivo

2.3 DISEÑO EDUCATIVO DEL MODULO HISTORIA

2.3.1 Objetivo. El conocer el por qué, dónde y para qué, surgieron los conocimientos que poseemos, permite proyectarlos para darles nuevas aplicaciones y tal vez llegar a nuevos conceptos. Con este objetivo se desarrolla la lección Historia del Software Didáctico para Álgebra Booleana, en sus dos módulos: Biografía de George Boole e Historia de la Lógica donde el usuario puede conocer el origen de este tema.

2.3.2 Motivación. En la lección historia, se explota la curiosidad por lo nuevo y lo desconocido del ser humano. Se invita al usuario a descubrir que hay detrás de cada escenario que se le presenta, conforme el usuario va explorando su curiosidad se vuelve más ambiciosa, hasta que se logra que llegue a donde se desea.

2.3.3 Escenario. La lección historia se desarrolla en una biblioteca (entrada, estantes, libros), en la cual el usuario se desliza en ella en búsqueda del conocimiento que desea adquirir. Se selecciona como escenario la biblioteca, pues es un ambiente que motiva al estudio, la concentración, la tranquilidad; como se desea que el usuario se sienta, para que asimile los conceptos que se exponen.

2.4 DISEÑO EDUCATIVO DE LA LECCIÓN TEORIA BASICA.

2.4.1 Objetivo. Al desarrollar esta lección se busca que el usuario pueda:

- Definir los conceptos de: variable booleana, ecuación booleana, tablas de verdad, operaciones del álgebra booleana.

- Crear tablas de verdad.

- Formar ecuaciones booleanas.

2.4.2 Motivación: En la lección teoría básica. Para motivar al usuario en el módulo Variable Booleana, se usa la técnica de correlación con lo real donde se establece una relación entre el concepto de Variable Booleana y conceptos ya aprendidos ó experiencias vividas por el usuario; dándole así un sentido de realidad al concepto que se quiere enseñar, Variable Booleana.

En los módulos: Operaciones del álgebra Booleana, ecuaciones Booleanas y Tablas de Verdad; se motiva al usuario mediante la experimentación. Una vez se le da a conocer los conceptos de cada módulo, se le insta a experimentar, a hacer algo, y así aplica los conocimientos ya adquiridos. A la vez se utilizan prudentemente los elogios y las censuras de acuerdo a los resultados que se obtengan de la experimentación del usuario, ya que la tendencia del comportamiento humano es el deseo de aprobación y de éxito, el estudiante se esforzará por realizar lo que se le proponga de la mejor manera.

2.4.3 Escenario. Los espacios abiertos producen sensación de tranquilidad y relajación; propicios para tener una mejor asimilación de los conceptos que se desean adquirir; por esta razón se selecciona como escenario el firmamento del que se

destaca el color azul que inspira tranquilidad, seguridad, anhelo de oportunidades para disfrutar de nuevos triunfos.

2.4.4 Evaluación. Como se indica en el literal 2.2.4, se manejan cuatro niveles de complejidad en los que se puede encontrar quien va a ser evaluado. Para el módulo variable booleana y operaciones del álgebra booleana no se hace evaluación por lo elemental del concepto. Para los módulos tablas de verdad y formación de ecuaciones(Ecuaciones minterms y ecuaciones maxterms) se manejan los cuatro niveles de complejidad y estos niveles se determinan de acuerdo al número de variables que tenga el ejercicio, como lo indica la siguiente Tabla 10.

Tabla 10. Complejidad Ejercicios Teoría Básica

Nivel de Complejidad	Numero de Variables
4	5
3	4
2	3
1	2

Los ejercicios que evalúan el módulo tablas de verdad le piden al usuario formar una tabla de verdad a partir de una ecuación booleana. Para el módulo formación de ecuaciones booleanas se utilizan ejercicios donde el usuario determina una ecuación

booleana a partir de una tabla de verdad. Los resultados que se obtienen en la evaluación son almacenados en la base de datos Estudiantes y la de la lección Teoría Básica..

2.5 DISEÑO EDUCATIVO DEL MODULO LECCIÓN McCLUSKEY.

2.5.1 Objetivo. Al desarrollar esta lección se busca que el usuario pueda:

- Crear ecuaciones booleanas en la forma minterms.
- Crear una tabla de agrupamiento base para reducir una ecuación booleana por el método de McCluskey.
- Reducir una expresión booleana utilizando este método.

2.5.2 Motivación. En la lección McCluskey se desarrolla siempre invitando al estudiante a participar, para que él sea quien construya sus conocimientos y no se convierta en un simple espectador. A medida que él realiza lo que se propone, se le incentiva con elogios no exagerados, que lo impulsan a continuar realizando con éxito el trabajo propuesto. De esta manera el

estudiante aplica los conceptos aprendidos y permite así una mejor asimilación de estos.

2.5.3 Escenario. En la lección McCluskey se utiliza un fondo abstracto alusivo al Álgebra Booleana, sobre el que se resaltan los conceptos que el usuario va adquirir.

2.5.4 Evaluación. Como se indica en el literal 2.2.4, se manejan cuatro niveles de complejidad en los que se puede encontrar quien va a ser evaluado. En esta lección se asignan los niveles de complejidad de acuerdo al número de variables que posea el ejercicio(ver Tabla 11).

Tabla 11. Complejidad Ejercicios Simplificación Quine McCluskey

NIVEL	VARIABLES
1	3
2	4
3	5
4	6

Se manejan dos formas de evaluación; para el módulo expresiones minterms, se utiliza evaluación del tipo selección múltiple. Para el

módulo tabla de agrupamiento se utilizan ejercicios donde el usuario crea las tablas que se le piden, y para la reducción se parte de la tabla de agrupamiento base con la que se evaluó al usuario en ese módulo. Los resultados que se obtienen en la evaluación son almacenados en la base de datos Estudiantes y la de lección.

2.6 DISEÑO EDUCATIVO DE LA LECCIÓN SIMPLIFICACIÓN ALGEBRAICA.

2.6.1 Objetivos. Al desarrollar esta lección se busca que el estudiante pueda:

- Identificar los Teoremas del Álgebra de Boole y los de DeMorgan.

- Aplicar los teoremas del Álgebra de Boole y DeMorgan para simplificar ecuaciones Algebraicas.

2.6.2 Escenario. La lección se desarrolla en medio de tonos verdes propicios para la concentración y el estudio, producen estabilidad y se resaltan los conceptos a estudiar en tono azul.

2.6.3 Motivación. Para mantener al usuario predispuesto a lo que se le quiere enseñar, se le hace participe en esta lección ya sea por ensayo y error, o por reflexión.

Se le propone al alumno simplificar una ecuación algebraica utilizando los teoremas que se le muestran a un lado de la pantalla, el usuario deja de ser un espectador para convertirse en un participante seleccionando los teoremas que se pueden aplicar a la ecuación propuesta ya sea por ensayo y error, o por reflexión sobre la aplicabilidad de estos teoremas.

2.6.4 Evaluación. **En esta lección se manejan los cuatro niveles de complejidad en que se puede encontrar el usuario que va a ser evaluado, y se determina de acuerdo al número de variables booleanas que contenga el ejercicio(Ver Tabla 9).**

Tabla 12. Complejidad Ejercicios Simplificación Algebraica

NIVEL	VARIABLES
1	3
2	4
3	5
4	6

Para evaluar al usuario en esta lección se le propone simplificar una ecuación booleana, utilizando los teoremas que se exhiben al lado de ésta, el usuario procura simplificar la ecuación con el menor número de equivocaciones posibles.

Los resultados obtenidos son almacenados en las tablas Estudiantes y Lección_Simplificación.

2.7 DISEÑO EDUCATIVO LECCIÓN APLICACIONES.

2.7.1 Objetivos. En esta lección el estudiante logrará:

- Identificar cada una de las compuertas lógicas(and, or, nor, nand, nor).

- Aprender el funcionamiento de cada una de las compuertas lógicas.

- Representar un circuito lógico por medio de una ecuación booleana.

- Deducir una ecuación booleana de un circuito digital.

2.7.2 Escenario. **La lección se desarrolla en medio de una gama de colores verdes que inducen a la concentración, en medio de un ambiente adecuado lleno de tranquilidad, estabilidad donde el estudiante se siente a gusto, lo cual permite que él esté predispuesto a conocer lo que se le va a enseñar.**

2.7.3 Motivación. **En esta lección se busca motivar al estudiante haciendo que participe en la lección, se le solicita que él llene la tabla de verdad para una ecuación booleana, que utilice los componentes que se le ofrecen en la lección, elabore su circuito y el software educativo le permite determinar la salida del circuito que elabore.**

2.7.4 Evaluación. **En esta lección se manejan los cuatro niveles de complejidad en que se puede encontrar el usuario que va a ser evaluado, y se determina de acuerdo al número de variables booleanas que contenga el ejercicio(Ver Tabla 13).**

Tabla 13. Complejidad Ejercicios Simplificación Algebraica

NIVEL	VARIABLES
1	3
2	4
3	5
4	6

3. DISEÑO COMPUTACIONAL

3.1 DISEÑO DE LA BASE DE DATOS

Estudiantes. Esta tabla contiene la información de quienes hayan usado el software, sus campos son: Código del estudiante, Nombre del estudiante, Clave del estudiante, Puntaje Acumulado, Ultima lección vista, Nivel de Complejidad, Estrellas Ganadas.

Ejercicios_Resueltos. Esta tabla contiene ejercicios resueltos del álgebra booleana para consultas, sus campos son: Tema, Nivel de complejidad del ejercicio, Enunciado, Respuesta, Gráficos, Número de Ejercicio.

Ejercicios_Propuestos. Con estos ejercicios se busca evaluar el progreso del usuario en el aprendizaje de cada módulo, sus campos son: Tema, Nivel de complejidad, Enunciado, Respuesta, gráfico, Puntaje, Número de Ejercicio, Número de Variables.

Lección_Historia. Contiene la información del estudiante con respecto a la lección historia, la tabla esta definida con los

siguientes campos: Código del Alumno. Número de veces que ha visitado la lección.

Lección_TeoriaB. Esta tabla contiene la información del estudiante en la lección teoría básica. Los campos que contiene son: Código del alumno, número de veces que ha visitado la lección, puntaje obtenido en la lección, número de veces que consulta los ejercicios resueltos del módulo, tres campos más que controla los ejercicios que ha resuelto de cada módulo: Operaciones booleanas, tablas de verdad y ecuaciones booleanas.

Lección_McCluskey. Esta tabla contiene la información del estudiante en la lección del método de simplificación de Quine McCluskey. Los campos que contiene son: Código del alumno, número de veces que ha visitado la lección, puntaje obtenido la lección, consulta a ejercicios resueltos de la lección, tres campos más que controla los ejercicios que ha resuelto de cada módulo de la lección: Expresiones Minterms, Tablas de Agrupamiento Base, Reducción.

Lección_Aplicación. Esta tabla contiene la información del estudiante en la lección aplicaciones del álgebra Booleana. Los campos que contiene son: Código del alumno, número de veces que ha visitado la lección, puntaje obtenido en la lección, consulta a

ejercicios resueltos de la lección, tres campos más que controla los ejercicios que ha resuelto de los módulos .

Lección_simplificación. Esta tabla contiene la información del estudiante en la lección del método de simplificación algebraica. Los campos que contiene son: Código del alumno, número de veces que ha visitado el módulo, puntaje obtenido en el módulo, consulta a ejercicios resueltos del módulo: Circuitos Digitales y Ecuaciones Booleanas.

3.2 DICCIONARIO DE DATOS

Tabla Estudiantes. El Cuadro 5 Describe los campos de la tabla Estudiantes de la Base de Datos Tablas.

Cuadro 5. Tabla Estudiantes.

CAMPO	TIPO	DESCRIPCIÓN
Cod_Est (Llave)	Alfabético	Identifica al estudiante
Nom_Est	Alfabético	Nombre del usuario
Clave	Alfabético	Clave del usuario
Punt_Est	Real	Puntaje acumulado del usuario
Ult_Lek	Alfabético	Nombre de la ultima lección vista por el usuario
Estr_Est	Entero Corto	Numero de estrellas ganadas por el usuario
Niv_Est	Entero Corto	Nivel de complejidad en que se encuentra el usuario

Tabla Ejercicios_Resueltos. El Cuadro 6 describe los campos de la tabla Ejercicios_Resueltos de la Base de Datos Tablas.

Cuadro 6. Tabla Ejercicios_Resueltos.

CAMPO	TIPO	DESCRIPCIÓN
Tema	Alfabético	Tema relacionado con el ejercicio.
Enunciado	Memo	Enunciado del ejercicio
Nivel	Entero Corto	Nivel de dificultad del ejercicio.
Respuesta	Memo	Respuesta del ejercicio.
Gráfico	Gráfico	Gráfico del ejercicio si este lo tiene.

Tabla Ejercicios_Propuestos. El Cuadro 7 describe los campos de la tabla Ejercicios_Propuestos de la Base de Datos Tablas.

Cuadro 7. Tabla Ejercicios_Propuestos.

CAMPO	TIPO	DESCRIPCION
Tema	Alfabético	Tema relacionado con el ejercicio.
Enunciado	Memo	Enunciado del ejercicio.
Nivel	Entero Corto	Nivel de dificultad del ejercicio.

Respuesta	Memo	Respuesta del ejercicio.
Gráfico	Gráfico	Gráfico del ejercicio si este lo tiene.
Puntaje	Real	Puntaje que da el ejercicio.
Num_ejer	Entero Corto	Asigna a cada ejercicio un número.
Variables	Entero Corto	Número de variables que tiene el ejercicio.

Tabla Lección_Historia. El Cuadro 8 describe los campos de la tabla Lección_Historia de la Base de Datos Tablas.

Cuadro 8. Tabla Lección_Historia.

CAMPO	**TIPO**	**DESCRIPCION**
Cod_Est	Alfabético	Código del usuario
Num_Vist	Entero Corto	Número de veces que ha visitado la lección

Lección_TeoriaB. El Cuadro 9 describe los campos de la tabla Lección_TeoríaB de la Base de Datos Tablas.

Cuadro 9. Tabla Lección_TeoríaB.

CAMPO	**TIPO**	**DESCRIPCIÓN**
Cod_Est	Alfabético	Código del Usuario
NumLecc_Vist	Entero Corto	Número de Veces que ha Visitado la lección.
Cons_Ejer	Entero Corto	Número de veces que ha consultado los ejercicios resueltos.
Num_Operac	Alfabético	Controla los ejercicios que ha resuelto el usuario del tema Operaciones Booleanas.
Num_Tablas	Alfabético	Controla los ejercicios que ha resuelto el usuario del tema Tablas de Verdad.
Num_Ecuac	Alfabético	Controla los ejercicios que ha resuelto el usuario del Tema Ecuaciones Booleanas.

Punt_Lec	Real	Puntaje Obtenido en la lección

Tabla Lección_McCluskey. El Cuadro 10 describe los campos de la tabla Lección_McCluskey de la Base de Datos Tablas.

Cuadro 10. Tabla Lección_McCluskey.

CAMPO	**TIPO**	**DESCRIPCIÓN**
Cod_Est	Alfabético	Código del Usuario
NumLecc_Vist	Entero Corto	Número de Veces que ha Visitado la lección.
Cons_Ejer	Entero Corto	Número de veces que ha consultado los ejercicios resueltos.
Num_Exp	Alfabético	Controla los ejercicios que ha resuelto el usuario del módulo expresiones Minterms
Num_Tab	Alfabético	Controla los ejercicios que ha resuelto el usuario del módulo tablas de agrupamiento.
Num_Red	Alfabético	Controla los ejercicios que ha resuelto

		el usuario del módulo reducción.
Punt_Lecc	Real	Puntaje obtenido en la lección.

Lección_Aplicación El Cuadro 11 describe los campos de la tabla Lección_Aplicación de la Base de Datos Tablas.

Cuadro 11. Tabla Lección_Aplicación.

CAMPO	TIPO	DESCRIPCIÓN
Cod_Est	Alfabético	Código del usuario
NumLecc_Vist	Entero Corto	Número de veces que ha visitado la Lección.
Cons_Ejer	Entero Corto	Número de veces que ha consultado Los ejercicios resueltos.
Num_cir	Alfabético	Controla los ejercicios que ha resuelto el usuario del tema circuitos digitales.
Num_eco	Alfabético	Controla los ejercicios que ha

		resuelto el usuario del tema ecuaciones booleanas.
Punt_lecc	Real	Puntaje obtenido en la lección

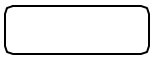
Tabla Lección Simplificación El Cuadro 12 describe los campos de la tabla Lección_Simplificación de la Base de Datos Tablas.

Cuadro 12. Tabla de la Lección Simplificación.

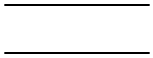
CAMPO	TIPO	DESCRIPCIÓN
Cod_Est	Alfabético	Código del usuario.
NumLecc_Vist	Entero Corto	Número de veces que ha visitado la lección.
Num_Simp	Alfabético	Controla los ejercicios que ha resuelto el usuario en la lección
Cons_Ejer	Entero Corto	Número de veces que ha consultado los ejercicios resueltos.
Punt_Lecc	Real	Puntaje obtenido en el módulo.

3.3 DIAGRAMAS DE FLUJOS DE DATOS DETALLADO.

CONVENCIONES: En la definición del diagrama de flujo de datos detallados utilizamos la siguiente convención:

 : Este simboliza un proceso.

 : Este simboliza un flujo de datos.

 : Este simboliza un almacenamiento de datos.

 : Este simboliza una fuente o destino de datos.

DIAGRAMA USUARIO

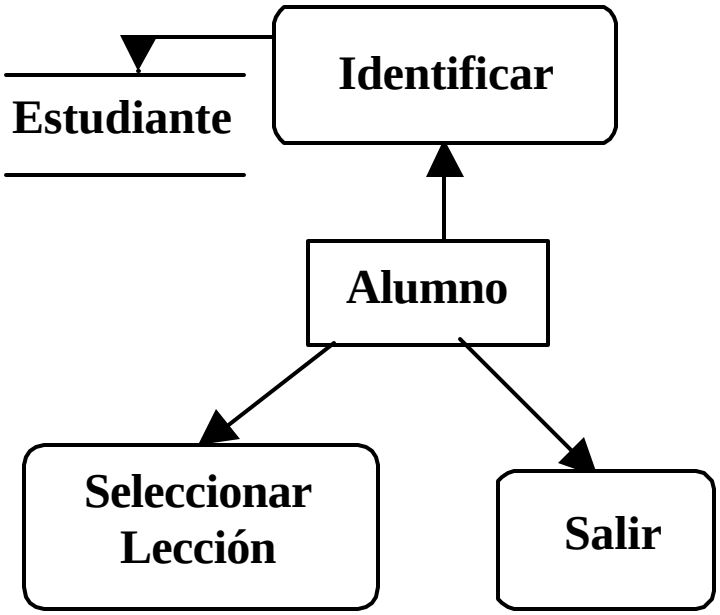


DIAGRAMA SELECCIÓN

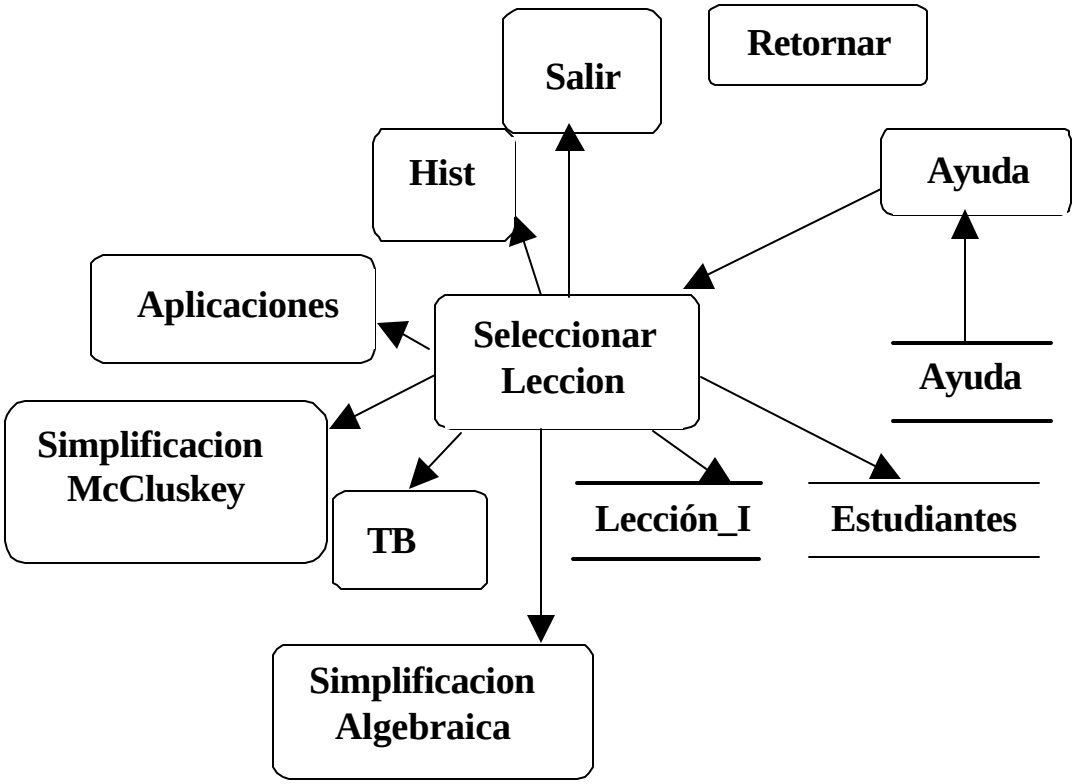


DIAGRAMA HISTORIA

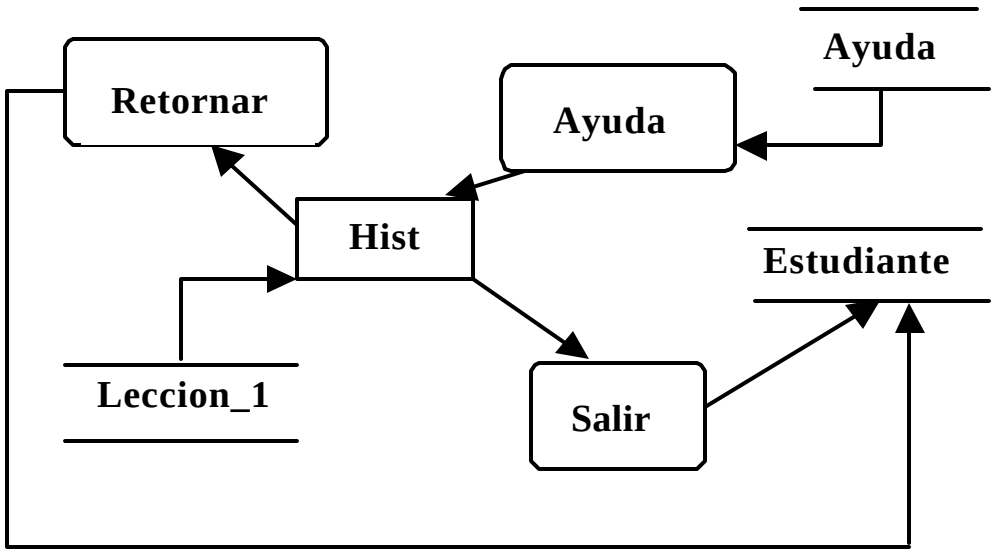
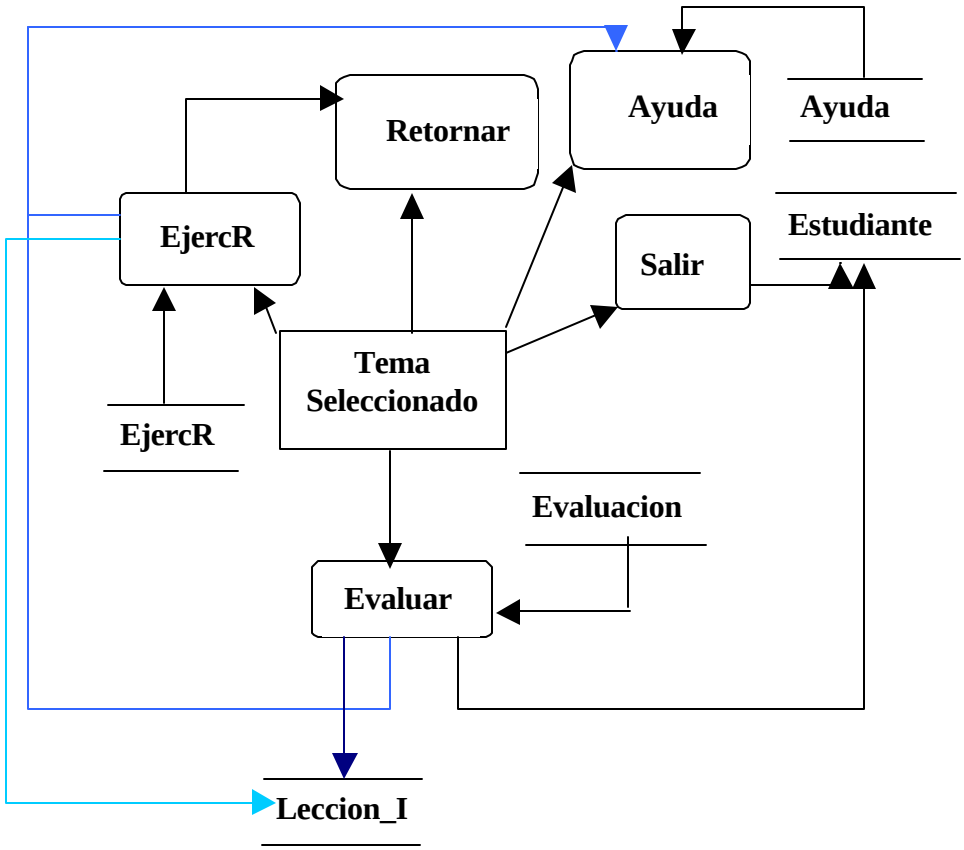


DIAGRAMA SELECCIONAR TEMA



En el diagrama Lección_I se refiere a las tablas: Teoría Básica, Simplificación, Aplicación, McCluskey, según el tema seleccionado.

Procesos:

1. Identificar: En este proceso se identifica si un usuario es nuevo o si ya es usuario. Si es nuevo se le toman todos sus datos y además se le pide que asigne una clave para poder ejecutar el software. Si no es nuevo se le pide que inserte su clave para así poder acceder al programa.

2. Salir: Sale definitivamente del programa.

3. Retornar: Retorna a la acción inmediatamente anterior.

4. Ayuda: Este proceso el usuario selecciona la ayuda que desea, ya sea para el manejo o aspectos del tema.

5. Seleccionar Lección: En este proceso un usuario puede escoger el módulo que desea seguir. Estos módulos son: Historia del Álgebra de Boole, Teoría Básica del Álgebra de Boole, Método de Simplificación Algebraica y Aplicaciones del Álgebra de Boole.

6. Iniciar con Tema Seleccionado (Historia del Álgebra de Boole, Teoría Básica del Álgebra de Boole, Método de Simplificación Algebraica, Método de Simplificación de McCluskey y Aplicaciones del Álgebra Boole): En este proceso se exhibe la información del módulo seleccionado.

7. Evaluar: En este proceso se hace un seguimiento del aprendizaje del usuario por medio de proponerle ejercicios de tema seleccionado para que los resuelva.

8. Ejercicios propuestos: En este proceso se le brinda al usuario la opción de observar ejercicios del módulo elegido resueltos

4. IMPLEMENTACION DEL SOFTWARE

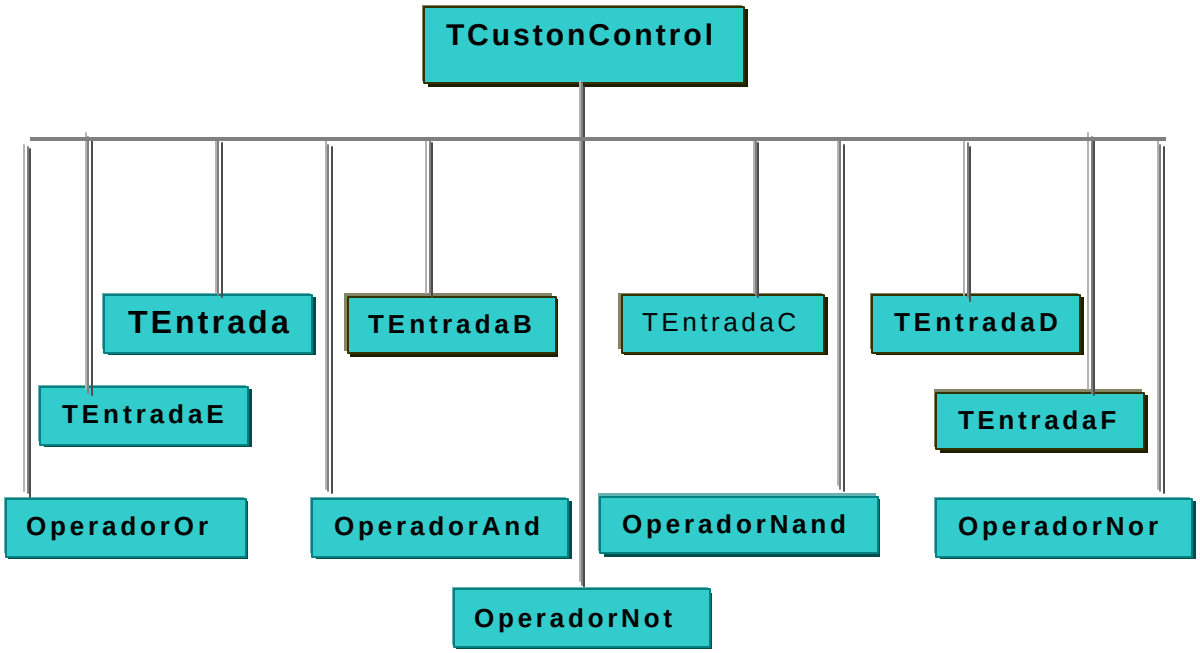
4.1 LENGUAJE UTILIZADO

El lenguaje que se utiliza para la elaboración de este proyecto es Delphi 5, es un lenguaje visual orientado a objetos que hereda todas las ventajas de Object Pascal y ofrece una gran flexibilidad para el desarrollo rápido de aplicaciones.

También permite la creación e instalación de componentes visuales y sirve para el diseño de interfaces gráficas de usuario.

Se selecciona este lenguaje porque es el que más se adapta a nuestras necesidades, es potente en la creación y manejo de objetos y ofrece ventajas ante otros lenguajes visuales por poseer una gran cantidad de componentes, para desarrollo de todo tipo de aplicaciones.

4.2 JERARQUÍA DE CLASES



4.3 CLASES

4.3.1 LA CLASE TcustomControl. La clase TcustomControl es una clase abstracta y es el nodo principal de la jerarquía de clases, de esta se derivan otras clases como: TOperadorAnd, TOperadorNand, TOperadorNot, TOperadorOr, ToperadorNor, TEntradaA, TEntradaB, TEntradaC, TEntradaD, TEntradaE, TentradaF.

4.3.2 LA CLASE TOperadorAnd.

Representa la compuerta And de un circuito eléctrico, puede tener de una a seis entradas y produce una salida cuyo resultado es la operación lógica And sobre sus respectivas entradas.

4.3.2.1 Propiedades de la clase

FEntrada. Entero. Contiene el número de entradas de la compuerta.

FSalida. Cadena. Guarda la salida de la compuerta.

FNumP. Entero. Número que ocupa la compuerta en la pantalla.

FTipo. Tipo. Contiene que clase de compuerta And es de acuerdo a su número de entradas.

FConecS. Booleana. Indica si la compuerta esta o no conectada a otra compuerta.

FEntrada1, FEntrada2, FEntrada3, FEntrada4, FEntrada5, FEntrada6. Cadena. Indica que componente entro a la compuerta.

FNEntrada1, FNEntrada2, FNEntrada3, FNEntrada4, FNEntrada5, FNEntrada6. Cadena. Contiene el nombre del componente que sirvió de entrada a la compuerta.

FConect1, FConect2, FConect3, FConect4, FConect5, FConect6. Bolean. Indica si una entrada esta o no conectada.

4.3.2.2 Métodos de la clase

ProcessClick: En tiempo de ejecución al dar clic derecho permite seleccionar en un menú el número de entradas para la compuerta, e inicializa las propiedades tipo y entradas.

Paint: Hace el dibujo del objeto fuente es decir la compuerta AND.

Validar. Permite en el diseño controlar que la compuerta tenga de dos a seis entradas.

ClicK. Muestra la compuerta en la pantalla.

EscogTipo: Durante el diseño permite cambiar la propiedad tipo de la compuerta y automáticamente cambia la propiedad Entrada.

SalirD. Asigna la salida a la propiedad Fsalida.

4.3.2.3 Eventos Utilizados en la clase

MouseDown: responde a un click del ratón. Evalúa que el click se dé sobre una zona sensible del elemento, si esto se cumple se actualizan las propiedades, relacionadas a cada zona sensible.

MouseMove: Responde al movimiento del señalador del ratón, al mover el señalador.

4.3.3 LA CLASE TOperadorNAnd. Representa la compuerta NAnd de un circuito eléctrico, puede tener de una a seis entradas y produce una salida cuyo resultado es la operación lógica NAnd sobre sus respectivas entradas.

4.3.3.1 Propiedades de la clase

FEntrada. Entero. Contiene el número de entradas de la compuerta.

FSalida. Cadena. Guarda la salida de la compuerta.

FNumP. Entero. Número que ocupa la compuerta en la pantalla.

FTipo. Tipo. Contiene que clase de compuerta NAnd es de acuerdo a su número de entradas.

FConecS. Booleana. Indica si la compuerta esta o no conectada a otra compuerta.

FEntrada1, FEntrada2, FEntrada3, FEntrada4, FEntrada5, FEntrada6. Cadena. Indica que componente entro a la compuerta.

FNEntrada1, FNEntrada2, FNEntrada3, FNEntrada4, FNEntrada5, FNEntrada6. Cadena. Contiene el nombre del componente que sirvió de entrada a la compuerta.

FConect1, FConect2, FConect3, FConect4, FConect5, FConect6. Bolean. Indica si una entrada esta o no conectada.

4.3.3.2 Métodos de la clase

ProcessClick: En tiempo de ejecución al dar clic derecho permite seleccionar en un menú el número de entradas para la compuerta, e inicializa las propiedades tipo y entradas.

Paint: Hace el dibujo del objeto fuente es decir la compuerta NAND.

Validar. Permite en el diseño controlar que la compuerta tenga de dos a seis entradas.

Click. Muestra la compuerta en la pantalla.

EscogTipo: Durante el diseño permite cambiar la propiedad tipo de la compuerta y automáticamente cambia la propiedad Entrada.

SalirD. Asigna la salida a la propiedad Fsalida.

4.3.3.3 Eventos Utilizados en la clase

MouseDown: responde a un click del ratón. Evalúa que el click se dé sobre una zona sensible del elemento, si esto se cumple se actualizan las propiedades, relacionadas a cada zona sensible.

MouseMove: Responde al movimiento del señalador del ratón, al mover el señalador.

4.3.4 LA CLASE TOperadorOr. Representa la compuerta Or de un circuito eléctrico, puede tener de una a seis entradas y produce una salida cuyo resultado es la operación lógica Or sobre sus respectivas entradas.

4.3.4.1 Propiedades de la clase.

FEntrada. Entero. Contiene el número de entradas de la compuerta.

FSalida. Cadena. Guarda la salida de la compuerta.

FNumP. Entero. Número que ocupa la compuerta en la pantalla.

FTipo. Tipo. Contiene que clase de compuerta Or es de acuerdo a su número de entradas.

FConecS. Booleana. Indica si la compuerta esta o no conectada a otra compuerta.

FEntrada1, FEntrada2, FEntrada3, FEntrada4, FEntrada5, FEntrada6. Cadena. Indica que componente entro a la compuerta.

FNEntrada1, FNEntrada2, FNEntrada3, FNEntrada4, FNEntrada5, FNEntrada6. Cadena. Contiene el nombre del componente que sirvió de entrada a la compuerta.

FConect1, FConect2, FConect3, FConect4, FConect5, FConect6. Bolean. Indica si una entrada esta o no conectada.

4.3.4.2 Métodos de la clase

ProcessClick: En tiempo de ejecución al dar clic derecho permite seleccionar en un menú el número de entradas para la compuerta, e inicializa las propiedades tipo y entradas.

Paint: Hace el dibujo del objeto fuente es decir la compuerta Or.

Validar. Permite en el diseño controlar que la compuerta tenga de dos a seis entradas.

ClicK. Muestra la compuerta en la pantalla.

EscogTipo: Durante el diseño permite cambiar la propiedad tipo de la compuerta y automáticamente cambia la propiedad Entrada.

SalirD. Asigna la salida a la propiedad Fsalida.

4.3.4.3 Eventos Utilizados en la clase

MouseDown: responde a un click del ratón. Evalúa que el click se dé sobre una zona sensible del elemento, si esto se cumple se actualizan las propiedades, relacionadas a cada zona sensible.

MouseMove: Responde al movimiento del señalador del ratón, al mover el señalador.

4.3.5 LA CLASE TOperadorNor.

Representa la compuerta Nor de un circuito eléctrico, puede tener de una a seis entradas y produce una salida cuyo resultado es la operación lógica Nor sobre sus respectivas entradas.

4.3.5.1 Propiedades de la clase

FEntrada. Entero. Contiene el número de entradas de la compuerta.

FSalida. Cadena. Guarda la salida de la compuerta.

FNumP. Entero. Número que ocupa la compuerta en la pantalla.

FTipo. Tipo. Contiene que clase de compuerta Nor es de acuerdo a su número de entradas.

FConecS. Booleana. Indica si la compuerta esta o no conectada a otra compuerta.

FEntrada1, FEntrada2, FEntrada3, FEntrada4, FEntrada5, FEntrada6. Cadena. Indica que componente entro a la compuerta.

FNEntrada1, FNEntrada2, FNEntrada3, FNEntrada4, FNEntrada5, FNEntrada6. Cadena. Contiene el nombre del componente que sirvió de entrada a la compuerta.

FConect1, FConect2, FConect3, FConect4, FConect5, FConect6. Bolean. Indica si una entrada esta o no conectada.

4.3.5.2 Métodos de la clase

ProcessClick: En tiempo de ejecución al dar clic derecho permite seleccionar en un menú el número de entradas para la compuerta, e inicializa las propiedades tipo y entradas.

Paint: Hace el dibujo del objeto fuente es decir la compuerta Nor.

Validar. Permite en el diseño controlar que la compuerta tenga de dos a seis entradas.

ClicK. Muestra la compuerta en la pantalla.

EscogTipo: Durante el diseño permite cambiar la propiedad tipo de la compuerta y automáticamente cambia la propiedad Entrada.

SalirD. Asigna la salida a la propiedad Fsalida.

4.3.5.3 Eventos Utilizados en la clase

MouseDown: responde a un click del ratón. Evalúa que el click se dé sobre una zona sensible del elemento, si esto se cumple se actualizan las propiedades, relacionadas a cada zona sensible.

MouseMove: Responde al movimiento del señalador del ratón, al mover el señalador.

4.3.6 LA CLASE TOperadorNot. Representa la compuerta Not de un circuito eléctrico, puede tener de una a seis entradas y produce una salida cuyo resultado es la negación de su entrada.

4.3.6.1 Propiedades de la clase

FEntrada. Entero. Contiene el número de entradas de la compuerta.

FSalida. Cadena. Guarda la salida de la compuerta.

FNumP. Entero. Número que ocupa la compuerta en la pantalla.

FTipo. Tipo. Contiene que clase de compuerta And es de acuerdo a su número de entradas.

FConecS. Booleana. Indica si la compuerta esta o no conectada a otra compuerta.

FEntrada1. Cadena. Indica que componente entro a la compuerta.

FNEntrada1. Cadena. Contiene el nombre del componente que sirvió de entrada a la compuerta.

FConect1. Boolean. Indica si una entrada esta o no conectada.

4.3.6.2 Métodos de la clase

Paint: Hace el dibujo del objeto fuente es decir la compuerta Not.

Validar. Permite en el diseño controlar que la compuerta tenga una entrada.

ClicK. Muestra la compuerta en la pantalla.

SalirD. Asigna la salida a la propiedad Fsalida.

4.3.6.3 Eventos Utilizados en la clase

MouseDown: responde a un click del ratón. Evalúa que el click se dé sobre una zona sensible del elemento, si esto se cumple se actualizan las propiedades, relacionadas a cada zona sensible.

MouseMove: Responde al movimiento del señalador del ratón, al mover el señalador.

4.3.7 LA CLASE TEntradaA, TEntradaB, TEntradaC, TEntradaD, TEntradaE, TEntradaF.

Representa las entradas a una de las compuertas del circuito lógico.

4.3.7.1 Propiedades de la clase

FNumP. Entero. Número que ocupa la Entrada en la pantalla.

FConect1, FConect2, FConect3, FConect4, FConect5, FConect6. Boolean. Indica si una entrada está o no conectada.

4.3.7.2 Métodos de la clase

Paint: Hace el dibujo del objeto fuente es decir la Entrada.

ClicK. Muestra la entrada en la pantalla.

4.3.7.3 Eventos Utilizados en la clase

MouseDown: responde a un click del ratón. Evalúa que el click se dé sobre una zona sensible del elemento, si esto se cumple se actualizan las propiedades, relacionadas a cada zona sensible.

MouseMove: Responde al movimiento del señalador del ratón, al mover el señalador.

4.3.8 LA CLASE TLínea. Representa las líneas que unen los componentes del circuito lógico.

4.3.8.1 Métodos de la clase

Paint: Hace el dibujo del objeto fuente es decir la Entrada.

ClicK. Muestra la entrada en la pantalla.

4.3.8.2 Eventos Utilizados en la clase

MouseDown: responde a un click del ratón. Evalúa que el click se dé sobre una zona sensible del elemento, si esto se cumple se actualizan las propiedades, relacionadas a cada zona sensible.

MouseMove: Responde al movimiento del señalador del ratón, al mover el señalador.

5. RECOMENDACIONES

A partir de la experiencia obtenida en el desarrollo del presente trabajo de investigación, los autoras recomiendan:

Continuar el uso de la Enseñanza Asistida por Computadoras incluyendo otros temas bases de la ingeniería. Trabajar en la creación de herramientas que permitan el uso de la Inteligencia Artificial en este proceso.

Validar la efectividad del software elaborado mediante su introducción en la carreras donde se aborde su contenido.

Desarrollar una herramienta que permite generar un mayor número de ejercicios para evaluar al usuario en las lecciones Aplicaciones y Simplificación algebraica.

6. CONCLUSIONES

El proceso planeación y desarrollo del trabajo de investigación, nos ha permitido a la las autoras arribar a las conclusiones siguientes:

- El uso de la Enseñanza Asistida por Computadoras (EAC) en el desarrollo del proceso de enseñanza-aprendizaje de una asignatura o disciplina conlleva la realización de un trabajo planificado y con la participación de un equipo multidisciplinario que permita integrar en forma de sistema los aspectos pedagógicos, sicológicos, metodológicos, el contenido de la asignatura y los recursos informáticos.

- El uso del software por los estudiantes de ingeniería les proporciona una herramienta de soporte para el auto estudio del Álgebra de Boole, le permite crear circuitos lógicos hallar su expresión correspondiente y los instruye para llevar a una forma simple pero equivalente una expresión booleana.

- La presentación amigable, clara y amena del software Didáctico Aprenda Álgebra Booleana, le permite al usuario una mejor asimilación de los conceptos básicos del álgebra de Boole.

GLOSARIO

Animación: movimiento de un dibujo o gráfica a través de la pantalla. Se emplea para efectos demostrativos de la evolución de un fenómeno en el tiempo, o simular el desplazamiento de un objeto así como el lograr efectos visuales. Generalmente se logra mediante la programación del dibujo rápido de un objeto gráfico, seguido del borrado de las partes que cambian y el volverlo a dibujar.

Aprendizaje Asistido por Computadora: enfoque de la educación basada en computadora en la cual se hace énfasis en el sujeto que incrementa su conocimiento, sin que a este se le enseñe o muestre explícitamente, el sujeto lo tiene que inferir o apropiarselo por sí mismo, empleando para ello algunos programas didácticos de computadora. El proceso ya no es de inculcación de información sino de capturar conocimiento. Las técnicas de descubrimiento y la programación en LOGO se señalan como buenos exponentes de este enfoque. Sin embargo este enfasis puede darsele a cualquier modalidad de la Enseñanza por Computadora, por lo que se le toma como sinónimo aunque en el fondo no lo sea.

Archivo ("file"): Información (o conjunto de datos estructurados) homogénea almacenada en un disco o diskette e identificados por un nombre. Está formado por un conjunto de registros ("records") que dan la información acerca de un individuo y están etiquetadas. Una base de datos consiste de un grupo estructurado y relacionado de archivos.

Base de datos: Es la organización y forma de manejo de una colección de datos (banco de datos) interrelacionados, estructurados y almacenados en disco (en archivos manejados de manera transparente para el usuario), dentro de un conjunto sin redundancias perjudiciales e innecesarias para su consulta y actualización. Frecuentemente bajo este nombre se quiere dar a entender al sistema manejador de una base de datos (DBMS) consistente en el paquete que permite entrar, actualizar, recuperar,

organizar, consultar, visualizar y generar reportes sobre la base de datos.

Cadena: Uno o varios caracteres alfanuméricos o especiales, incluyendo el espacio o blanco (en este caso tiene que estar encerrados entre comillas, para saber el comienzo y fin de la cadena). Es simplemente la secuencia de los códigos de esos caracteres y no representa un valor o cantidad, por lo que puede ser utilizada como nombre de una variable o instrucción en un lenguaje de cómputo (siempre que se respete la sintaxis del lenguaje) o como un dato sobre el cual se va en principio a realizar una operación no numérica como ordenar, consultar, comparar etc.

Categorías del aprendizaje: Son los cinco dominios de capacidades que un estudiante puede aprender, tal y como lo propuso Gagné: 1) habilidades intelectuales; 2) estrategias cognitivas; 3) información verbal; 4) habilidades motoras; 5) actitudes.

Código: es la representación de un caracter o dígito en la computadora para su uso.

Código absoluto o código objeto es cuando este conjunto de caracteres representan las instrucciones ejecutables directamente por la computadora. El código ASCII (American Standard Code for Information Interchange) es una representación homologada de los caracteres alfanuméricos y especiales que se usan en los lenguajes de computadora mediante una secuencia de 8 bits (Byte).

Código Binario: representación de cualquier número real mediante bits en el sistema base dos o sistema (numérico) binario.

Código Fuente: Es el conjunto de instrucciones de un programa escrito en un lenguaje de alto nivel y que ya se encuentran en la computadora o alguno de sus periféricos en una representación homologada como la ASCII, pero que no es directamente ejecutable

por la computadora y que necesita de una traducción (interpretación o compilación).

Comando: Es un elemento de un lenguaje sencillo de programación que consiste en órdenes o comandos eventualmente modificados; o precisados por una secuencia de parámetros.

Computadora: Conjunto de dispositivos, generalmente electrónicos, que permiten procesar y almacenar información siguiendo las instrucciones de un programa.

Cursor: Es un marcador que indica el lugar en la pantalla en el cual espera una entrada por parte del usuario. El cursor generalmente es un guión que parpadea pero puede ser una línea, un cuadrado, una flecha, dependiendo de la computadora. El cursor se puede mover tanto por las teclas de flecha como por algún dispositivo externo como el ratón.

Dato: Cualquier información utilizada por un programa tanto numérica como alfabética que no tiene algún significado en sí, si no dentro de un contexto con otros datos con los que se relaciona (información).

Diagrama de Flujos: Es un diagrama que muestra la secuencia de las instrucciones de un programa con todo y sus ramificaciones y la lógica necesaria para las ramificaciones, repeticiones, etc.

Directorio: Lista de los archivos almacenados en un disco u otro dispositivo periférico.

Disco duro: Disco rígido de plástico con sustrato magnético que almacena una gran cantidad de información. Tiene que estar al

vacío y generalmente es fijo o no reem plazable dentro de una computadora.

Documentación: Conjunto de manuales y comentarios en los programas de cómputo que explican los objetivos, el diseño, el uso correcto, las aplicaciones, ejemplos y posibles errores de un "software".

Ejercitación y práctica ("Drill and Practice"): Modalidad de la enseñanza asistida por computadora que pretende el aprendizaje de conocimientos o habilidades a través de la realización de ejercicios y la visualización de las repuestas apropiadas.

Enseñanza asistida por computadora (EAC): Referida también como instrucción asistida por computadora, define el campo de metodologías, técnicas, y programas que empleando la computadora pretenden enseñar algo a un estudiante. De alguna manera auxilian o sustituyen al maestro proveyendo información, ejercicios, demostrando conceptos y convirgiendo las ideas dentro de un modo interactivo que requiere respuesta del estudiante y que a su vez modula la dirección de la instrucción.

Estrategia del control del aprendizaje: Procedimientos instruccionales que permiten a los estudiantes de un "courseware" tomar decisiones sobre las lecciones, las actividades y exámenes a seguir.

Herramienta: Paquete de cómputo que ofrece todas las facilidades para desempeñar una actividad sin tener que programar la computadora. Por ejemplo el procesador de textos ofrece todas las facilidades de maquinilla y en general de producir textos. Se puede aprender algo indirectamente empleando herramientas, por ejemplo contabilidad empleando una hoja electrónica de cálculo. Así indirectamente mejoran sus facultades de descubrimiento, decisión y deducción. Se contrapone a la enseñanza o instrucción asistida por computadora cuyo énfasis y métodos se dedican a la

inculcación del conocimiento y habilidades, pero sobretod porque el contenido explícitamente trata sobre el tema a aprender, mientras que la herramienta es libre de contenido.

Iconos: Símbolos visuales empleados para representar opciones dentro de un menú o datos.

Instrucción Asistida por Computadora: Sinónimo de Enseñanza Asistida por Computadora, muy conocida por sus siglas en Inglés: CAI.

Interacción: Forma de operar con un lenguaje de computqadora en el cual durante la ejecución el usuario puede entrar datos, información y eventualmente instrucciones.

Interactivo o conversacional: forma de comunicación dialogal o = bidireccional entre el hombre y la computadora.

Lógica Binaria o Algebra de Boole (Booleana): Sistema lógico y algebraico en el que únicamente se tienen dos valores posibles: Verdadero y Falso, a veces tambien se maneja como 0 y 1. Shannon aplicó esta lógica a los circuitos de conmutación y creo lo que se conoce como Circuitos Combinatorios. En esta lógica solo hay dos constantes 0 o 1, y todas las variables y expresiones que se puedan armar pueden valeer sólo 0 o 1. Los circuitos de una computadora se basan en esta lógica y en la teoría de autómatas. No confundir con sistema binario.

Memoria: Parte de una computadora donde se almacena datos o instrucciones. Cuando ésta es de tipo permanente se le llama ROM y cuando es temporal o volátil se le llama RAM. Cuando está dentro de la computadora y trabaja directamente con el CPU se le llama memoria principal, cuando es externa a la computadora y trabaja directamente con la memoria principal, se le llama memoria secundaria.

Pantalla: Se emplea este término en dos sentidos, el primero como sinónimo de monitor o "display", pero dentro del contexto de enseñanza asistida por computadora como la información que en un momento dado se presenta en el monitor ante los ojos del estudiante. Aunque es deseable que un marco quepa en una pantalla no es indispensable. Monitor "bit-map": Es un tipo de monitor o pantalla de display que permite direccionar cada uno de los puntos de la pantalla, llamados "pixel". Esto se contrasta con los monitores de rastreo en el cual solamente pueden direccionar una línea completa y la gráfica se forma barriendo todas las líneas del monitor de arriba hacia abajo. Es claro que los monitores "bit-map" tienen un mayor poder de resolución.

Paginación: Técnica del "courseware" que contrasta con el barrido en la cual cada imagén llamada "pantalla" es reemplazada enteramente por otra.

Paquete o Software de Aplicación: Conjunto de Programas que realizan o satisfacen a toda una aplicación, por ejemplo: el procesador de textos, la hoja electrónica de cálculo, los sistemas manejadores de bases de datos etc.

Periféricos: Es cualquier dispositivo físico (de "hardware") de entrada, salida y almacenaje que puede ser conectado a una computadora.

Programa: Un conjunto de instrucciones escritas en un lenguaje de computadora que tienen un principio y un fin y realizan una tarea específica.

Ratón: Dispositivo de entrada empleado para mover el cursor dentro de la pantalla y eventualmente para seleccionar alguna opción. El ratón se desplaza sobre una superficie plana cuyo movimiento es repetido por el cursor en la pantalla.

Retroalimentación ("feedback"): conjunto de acciones y mensajes que genera un "courseware" como resultado inmediato de una petición o respuesta del estudiante.

Sistema Binario o sistema numérico binario: Se refiere al sistema de representación de números y a su aritmética, cuya base es dos. En este sistema todos los dígitos de un número cualquiera son formados exclusivamente con 0 ó 1. No confundir con lógica binaria o álgebra de Boole. También se emplea para indicar una decisión que involucra dos alternativas posibles, como por ejemplo verdadero y falso.

Sistema operativo: Conjunto de programas básicos de cualquier computadora que le permiten efectuar sus funciones primarias - y trabajar con sus periféricos.

"Software": Cualquier programa, paquete, o sistema que contiene instrucciones para que una computadora haga algo.

Teclas de función: Conjunto de teclas especiales numeradas que no tienen una definición fija y que por lo tanto pueden ser utilizadas por un programa para facilitar la realización rápida de algunas instrucciones.

Variables: Nombres o etiquetas dados a lugares en la memoria de la computadora que contienen datos (valores) en un programa, dado que son llamados por un programa como etiquetas el valor puede cambiar.

Vector: Una lista ordenada de variables del mismo tipo (nombres, números, etc.).

Ventanas: Es una porción encuadrada de la pantalla que es manejada independientemente de la información desplegada en el

fondo de la pantalla. Las pantallas se sobreponen ("overlay") y pueden desplazarse dentro de la pantalla o cambiar el tamaño y forma.

BILIOGRAFIA

APPLE, M.. Maestros y Textos. Barcelona, España, Paidós MEC, 1989.

BECKER, Henry J. Using Computer For Instruction. Byte (USA): 149-162 Febrero 1987.

BOONET, A., y otros. An ICAI System For Teaching Derivates in Mathematics. Computer in Education. Lausana, Suiza: 135-142 Julio 1981.

BRUGUÉS, Amadeu. Desarrollo A Toda Máquina. Byte (Madrid, España) 8 : 54-56 Junio 1995.

CABRALES VERGEL, Gustavo. Metodología Para La Elaboración de Anteproyectos y Proyectos de Investigación. Corporación UNICOSTA. Barranquilla. 1995. 2 Ed.

CUESTA, L, Gil, PADILLA, A. Schaum: Electrónica Digital. Álgebra de Boole, Circuitos Combinacionales y Secuenciales, Automatismos, Memorias. Mexico: McGraw Hill. 1993.

DIEUDONNÉ, Jean. La Abstracción En Matemática Y La Evolución Del Algebra. En Piaget, J., y otros. La Enseñanza de las Matemáticas. Madrid, España, Aguilar, 1971. p42-46.

FEDELMAN, Robert S. Psicología Con Aplicaciones Para Ibero América. Mexico: McGraw Hill. 1996.

GARCÍA, D[24] GUTIÉRREZ, J., y otros. Tratamiento De Las Interacciones Del Alumno En Un Sistema Tutor Inteligente. IV Congreso Iberoamericano de Inteligencia Artifical. Chile, 1994.

GOOD, Thomas L. BROPHY, Jere. Psicología Educativa Contemporánea. 5 Ed. México: McGraw Hill. 1996.

HERRERA S., René, y otros. La Informática En La Educación Cubana. CID. Electrónica y Proceso de Datos en Cuba. Habana, Cuba, No. 2, 1992.

KENDALL. Keneth. Análisis y Diseño De Sistemas. México: Prentice Hall. 1995.

LERENA, C.. Materiales De Sociología De La Educación Y La Cultura. Madrid, España, Grupo Cultural Zero, 1985.

LOWD, Beth. The Right Ways To Use Computer. The Computing Teacher (USA) : 4-5 Octubre 1986.

MORRIS, Charles. Introducción a la Psicología. 7 Ed. México: Prentice Hall. 1992.

NERICE, Imideo. Hacia Una Didáctica General Dinámica. Buenos Aires: Kapelusz. 1973.

OSCER, Dan, GROBMAN, Steve. Aprendiendo Delphi 3 En 14 Días. México: Prentice Hall. 1998.

SCHILDT, Herbert. C++, Guía De Autoenseñanza. 3 Ed. Madrid: McGraw Hill. 1995.

SENN, Jmes A. Análisis y Diseño De Sistemas De Información. México: McGraw Hill. 1995.

TOCCI. Ronald. Sistemas Digitales, Principios y Aplicaciones. México: Prentice Hall. 1996.

UNESCO. Hacia Una Nueva Etapa De Desarrollo Educativo. Boletin. Quinta Reunión del Proyecto Principal de Educación para América Latina y el Caribe. Santiago de Chile, Chile, 1993.

VERGEL CABRALES, Gustavo. Metodología Para La Elaboración De Anteproyectos y Proyectos De Investigación. 2 Ed. Barranquilla: Publicaciones Corporación UNICOSTA. 1995.

INTRODUCCION

El Software Didáctico Aprenda Álgebra de Boole es flexible y por eso ofrece la oportunidad a la persona encargada de su mantenimiento de realizar modificaciones según las necesidades que surjan y así generar nuevas aplicaciones, para esto se le recomienda seguir las sugerencias de este manual para cumplir con el propósito inicial de su creación.

En este manual de sistema se presentan cuatro capítulos que sirven de ayuda para entender el código fuente de este Software. El primer capítulo hace referencia a los requerimientos para el mantenimiento y ejecución de este Software.

El segundo capítulo describe las estructuras de datos, sus campos y el flujo de datos detallado.

El tercer capítulo describe los procedimientos principales y muestra el código fuente de éstos.

El cuarto capítulo muestra el árbol de directorios del programa fuente, instaladores y archivos de ayuda.

1. REQUERIMIENTOS

1.1 HARDWARE.

Para editar el programa se requiere un equipo compatible con Windows, con memoria RAM de 32 MB. Un procesador de 133 Mhz o más, Mouse, impresora, unidad de CD-ROM, espacio en disco para su ejecución de 200 Mb. ó más y tarjeta de video.

1.2 SOFTWARE.

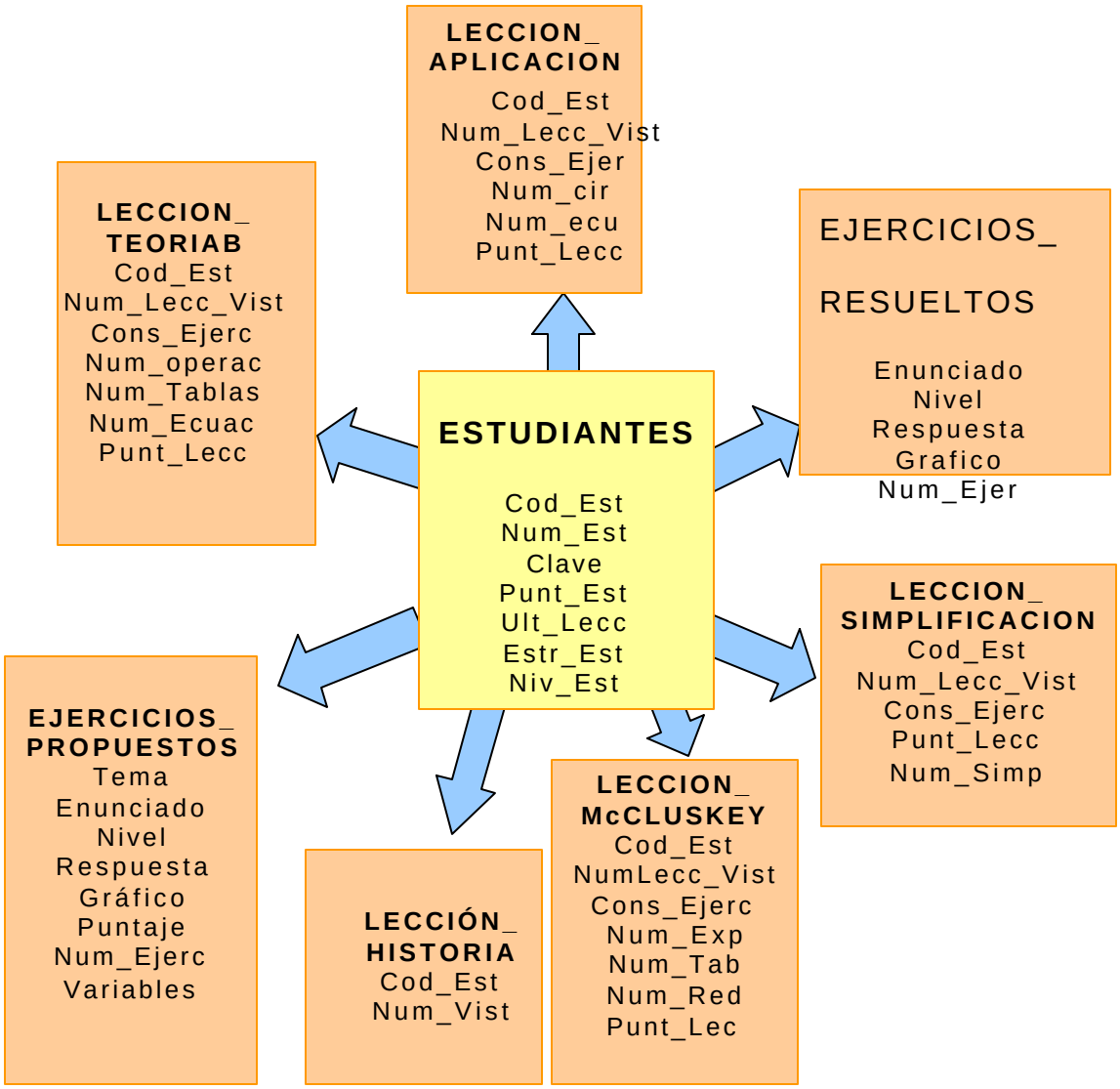
El sofware Didáctico fue creado en Delphi 5 así que para sus actualizaciones se debe utilizar este paquete. Los gráficos se diseñaron en Corel Draw y un ilustrador cómico.

1.3 DISEÑO EDUCATIVO.

Las modificaciones a realizar deben basarse en la fundamentación educativa aplicada para el diseño del software, la información sobre ésta se encuentra en el documento impreso que acompaña al software.

2. ESTRUCTURAS DE DATOS

2.1 MODELO ENTIDAD RELACION.



2.2 DICCIONARIO DE DATOS

2.2.1 Tabla Estudiantes

Definición de los campos.

Campo 1.

Nombre. Cod_Est.

Descripción. Identifica al estudiante mediante un número único.

Tipo. Alfabético.

Campo 2.

Nombre. Nom_est.

Descripción. Nombre del estudiante.

Tipo. Alfabético.

Campo 3.

Nombre. Clave secreta del usuario.

Descripción. Guarda la clave del usuario.

Tipo. Alfabético.

Campo 4.

Nombre. Punt_Est.

Descripción. Puntaje acumulado del usuario en las evaluaciones realizadas.

Tipo. Real.

Campo 5.

Nombre. Ult_lecc.

Descripción. Nombre de la última lección visitada.

Tipo. Alfabético.

Campo 6.

Nombre. Niv_Est.

Descripción. Nivel de complejidad en el que se encuentra el estudiante.

Tipo. Entero corto.

2.2.2 Tabla Ejercicios Resueltos

Campo 1.

Nombre. Tema.

Descripción. Tema relacionado con el ejercicio.

Tipo. Alfabético.

Campo 2.

Nombre. Enunciado.

Descripción. Enunciado del ejercicio.

Tipo. Memo.

Campo 3.

Nombre. Nivel.

Descripción. Nivel de dificultad del ejercicio.

Tipo. Entero Corto.

Campo 4.

Nombre. Respuesta.

Descripción. Respuesta del ejercicio.

Tipo. Memo.

Campo 5.

Nombre. Grafico.

Descripción. Grafico del ejercicio si este lo tiene.

Tipo. Gráfico.

4.3.3 Tabla Ejercicios Propuestos

Campo 1.

Nombre. Tema.

Descripción. Tema relacionado con el ejercicio.

Tipo. alfabético.

Campo 2.

Nombre. Enunciado.

Descripción. Enunciado del ejercicio.

Tipo. Memo.

Campo 3.

Nombre. Nivel.

Descripción. Nivel de dificultad del ejercicio.

Tipo. Entero Corto.

Campo 4.

Nombre. Respuesta.

Descripción. Respuesta del ejercicio.

Tipo. Memo.

Campo 5.

Nombre. Gráfico.

Descripción. Grafico del ejercicio si este lo tiene.

Tipo. Grafico.

Campo 6

Nombre. Puntaje.

Descripción. Puntaje que del ejercicio.

Tipo. Real.

Campo 7.

Nombre. Num_Ejer.

Descripción. Asigna a cada ejercicio un número.

Tipo. Entero Corto.

Campo 8.

Nombre. Variables.

Descripción. Número de variables que tiene el ejercicio.

Tipo. Entero Corto.

2.2.3 Tabla Lección Historia.

Campo 1.

Nombre. Cod_Est.

Descripción. Código del usuario.

Tipo. Alfabético.

Campo 2.

Nombre. Num_Vist.

Descripción. Número de veces que el usuario ha visitado la lección Historia del Álgebra Booleana.

Tipo. Entero Corto.

2.2.4 Lección_TeoriaB.

Campo 1.

Nombre. Cod_Est.

Descripción. Código del usuario.

Tipo. Alfabético.

Campo 2.

Nombre. NumLecc_Vist.

Descripción. Número de veces que el usuario ha visitado la lección Teoría Básica del Álgebra Booleana.

Tipo. Entero Corto.

Campo 3.

Nombre. Cons_Ejer.

Descripción. Número de veces que ha consultado los ejercicios resueltos de la lección Teoría Básica del Álgebra Booleana.

Tipo. Entero Corto.

Campo 4.

Nombre. Num_Operac.

Descripción. Controla los ejercicios que ha resuelto el usuario del módulo Operaciones del Álgebra Booleana de la lección Teoría Básica.

Tipo. Alfabético.

Campo 5.

Nombre. Num_Tab.

Descripción. Controla los ejercicios que ha resuelto el usuario módulo Tablas de Verdad lección Teoría Básica.

Tipo. Alfabético.

Campo 6.

Nombre. Num_Ecuac.

Descripción. Controla los ejercicios que ha resueltos el usuario módulo Ecuaciones Booleanas de la lección Teoría Básica.

Tipo. Alfabético.

Campo 7.

Nombre. Punt_Lecc.

Descripción. Puntaje obtenido en la lección Teoría Básica.

Tipo. Real.

2.2.5 Tabla Lección McCluskey.

Campo 1.

Nombre. Cod_Est.

Descripción. Código del usuario.

Tipo. Alfabético.

Campo 2.

Nombre. NumLecc_Vist.

Descripción. Numero de veces que el usuario ha visitado la lección Método de Simplificación de Quine McCluskey.

Tipo. Entero Corto.

Campo 3.

Nombre. Cons_Ejer.

Descripción. Numero de veces que ha consultado los ejercicios resueltos.

Tipo. Entero Corto.

Campo 4.

Nombre. Num_Exp.

Descripción. Controla los ejercicios que ha resuelto el usuario del Módulo Expresiones Minterms.

Tipo. Alfabético..

Campo 5.

Nombre. Num_Tab.

Descripción. Controla los ejercicios que ha resuelto el usuario del Módulo Tablas de Agrupamientos de la lección Simplificación de McCluskey.

Tipo. Alfabético..

Campo 6.

Nombre. Num_Red.

Descripción. Controla los ejercicios que ha resuelto el usuario del Módulo reducción del Método de simplificación de McCluskey.

Tipo. Alfabético..

Campo 7.

Nombre. Punt_Lecc.

Descripción. Puntaje obtenido en el módulo.

Tipo. Real.

2.2.6 Lección_Aplicación

Campo 1.

Nombre. Cod_Est.

Descripción. Código del usuario.

Tipo. Alfabético.

Campo 2.

Nombre. NumLecc_Vist.

Descripción. Numero de veces que el usuario ha visitado la lección Simplificación Algebraica.

Tipo. Entero Corto.

Campo 3.

Nombre. Cons_Ejer.

Descripción. Numero de veces que ha consultado el estudiante los ejercicios resueltos de la lección.

Tipo. Entero Corto.

Campo 4.

Nombre. Num_cir.

Descripción. Controla los ejercicios que ha resuelto el usuario del Módulo Circuitos digitales.

Tipo. Alfabético..

Campo 5.

Nombre. Num_ecu.

Descripción. Controla los ejercicios que ha resueltos el usuario del Módulo deducción de Ecuaciones booleanas.

Tipo. Alfabético..

Campo 6.

Nombre. Punt_Lecc.

Descripción. Puntaje obtenido en la lección.

Tipo. Real.

2.2.7 Tabla Lección Simplificación

Campo 1.

Nombre. Cod_Est.

Descripción. Código del usuario.

Tipo. Alfabético.

Campo 2.

Nombre. NumLecc_Vist.

Descripción. Numero de veces que el usuario ha visitado la lección.

Tipo. Entero Corto.

Campo 3.

Nombre. Cons_Ejer.

Descripción. Numero de veces que ha consultado los ejercicios resueltos.

Tipo. Entero Corto.

Campo 4.

Nombre. Punt_Lecc.

Descripción. Puntaje obtenido en el módulo.

Tipo. Real.

Campo 5.

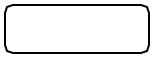
Nombre. Num_ecu.

Descripción. Controla los ejercicios que ha resuelto el usuario de la lección Simplificación Algebraica.

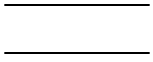
Tipo. Alfabético..

2.3 FLUJO DE DATOS DETALLADO.

CONVENCIONES: En la definición del diagrama de flujo de datos detallados utilizamos la siguiente convención:

 : Este simboliza un proceso.

 : Este simboliza un flujo de datos.

 : Este simboliza un almacenamiento de datos.

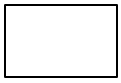
 : Este simboliza una fuente o destino de datos.

DIAGRAMA USUARIO

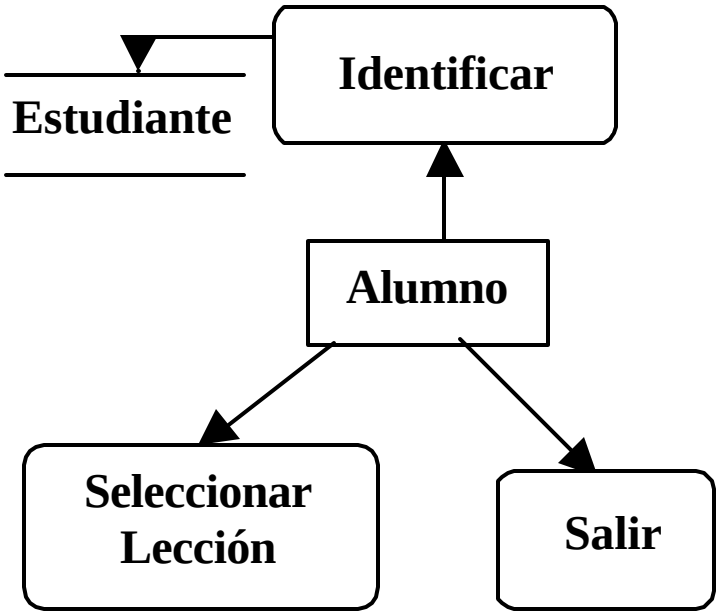


DIAGRAMA SELECCIÓN

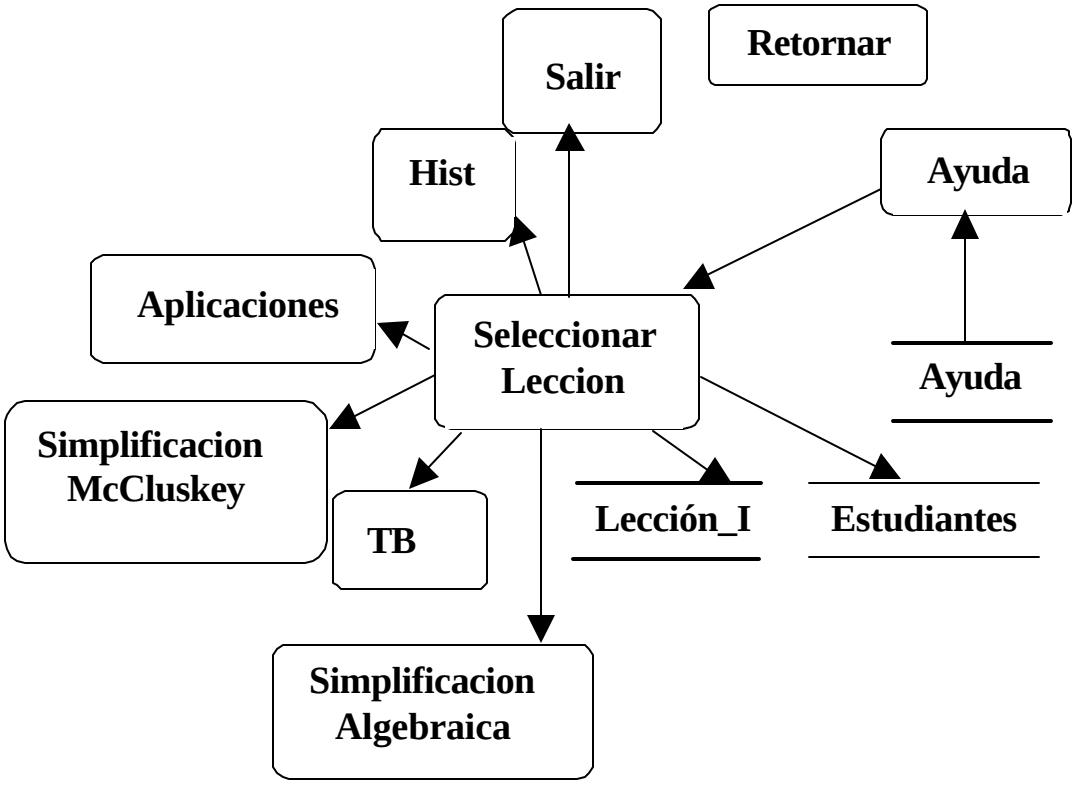


DIAGRAMA HISTORIA

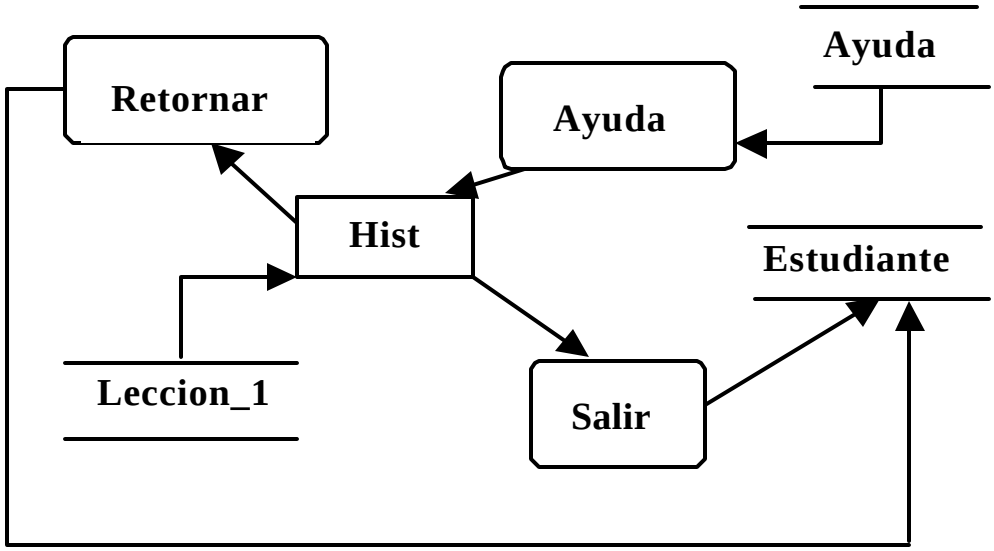
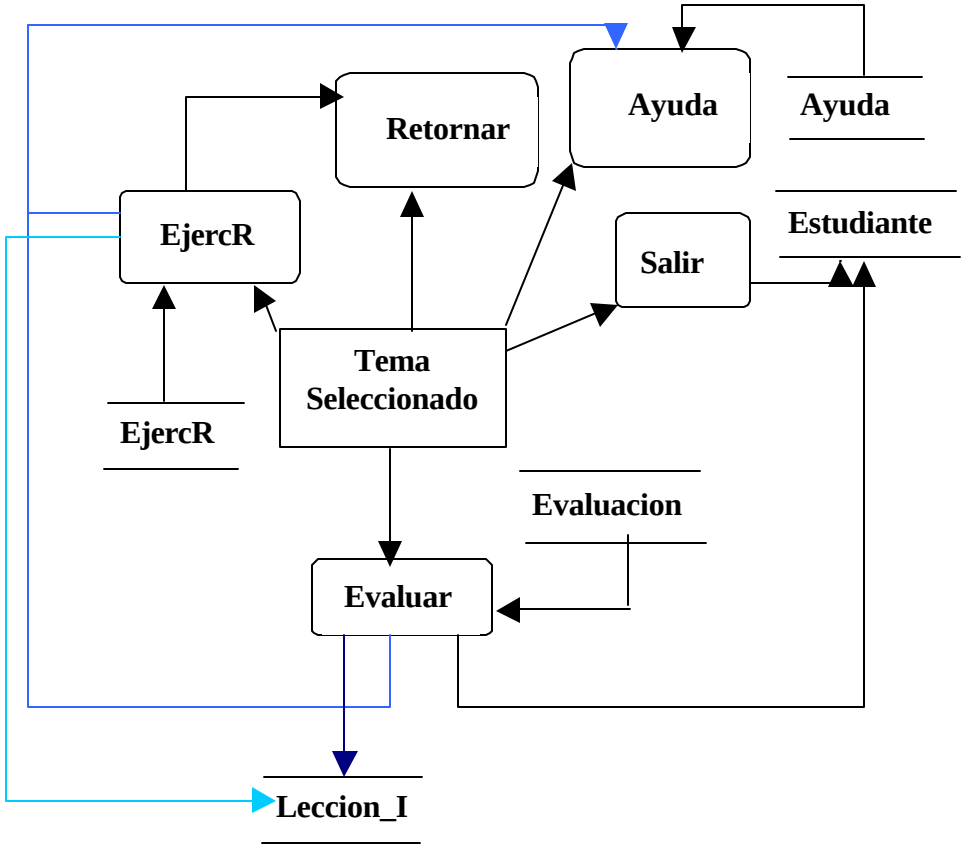


DIAGRAMA SELECCIONAR TEMA



En el diagrama Lección_I se refiere a las tablas: Teoría Básica, Simplificación, Aplicación, McCluskey, según el tema seleccionado.

3. PROCEDIMIENTOS PRINCIPALES

3.1 MODULO IDENTIFICAR.

PROCEDIMIENTO VALIDAR_NUM. Valida que los caracteres de una cadena sean números.

```
procedure ValidarNum(Numero:String;var sw0:Boolean);

var
 Num0:Numeros;
 LongNum,i:Integer;

begin
 sw0:=False;
 Num0:=['0','1','2','3','4','5','6','7','8','9'];
 LongNum:=Length(Numero);
 if (Numero='') then
  sw0:=true
 else
 begin
  i:=1;
  while ((i<=LongNum) and (sw0=False)) do
   if (Numero[i] in Num0) then
    i:=i+1
   else
   begin
    numero:=' ';
    sw0:=True;
   end;
 end;
end;
```

PROCEDIMIENTO VALIDAR_LETRAS. Verifica que los caracteres de una cadena sean solo letras.

```
procedure ValidarLet(Cadena:String;var sw0:Boolean);

var
 Let0:Letras;
 LongLet,i:Integer;
begin
 sw0:=False;

Let0:=['á','a','b','c','d','e','é','f','g','h','i','í','j','k','l','m','n','ñ','o','ó','p','q','r'
,'s','t','u','ú','v','w','x','y','z','A','Á','B','C','D','E','É','F','G','H','I','Í','J','K','L'
,'M','N','Ñ','O','Ó','P','Q','R','S','T','U','Ú','V','W','X','Y','Z',' ','.'];
 LongLet:=Length(Cadena);

 if (Cadena=' ') then
  showmessage('Por favor digite los datos completos')
 else
 begin
  i:=1;

  while ((i<=LongLet) and (sw0=False)) do

   if (Cadena[i] in Let0) then
    i:=i+1
   else

   begin
    showmessage('Por favor solo digite letras');
    cadena:=' ';
    sw0:=True;
   end;
 end;
end;
```

PROCEDIMIENTO NUEVO_USUARIO. Constata que el nuevo usuario diligencie su datos correctamente.

```
procedure TForm7.SpeedButton1NClick(Sender: TObject);

var
  LongN,LongC,i,sw1,sw2:Integer;

begin
 Num:=['0','1','2','3','4','5','6','7','8','9'];
Let:=['a','b','c','d','e','f','g','h','i','j','k','l','m','n','ñ','o','p','q','r','s','t','u','v','
w','x','y','z','A','B','C','D','E','F','G','H','I','J','K','L','M','N','Ñ','O','P','Q','R',
'S','T','U','V','W','X','Y','Z',' ','.'];

 Nombre:=Edit1.Text;
 Codigo:=Edit2.Text;
 LongN:=Length(Nombre);
 LongC:=Length(Codigo);
 Estudiantes.Open;
 sw1:=0;
 sw2:=0;
 if (Nombre=' ') or (Codigo=' ') then
  showmessage('Por favor,digite los datos completos')
 else
 begin
  i:=1;
  sw1:=0;
  while (i<=LongC)and(sw1=0) do
   if (Codigo[i] in Num) then
    i:=i+1
   else
   begin
    Showmessage('El codigo solo debe contener números');
    Codigo:=' ';
    Edit2.Clear;
    Edit2.SetFocus;
    sw1:=1;
   end;
  i:=1;
  sw2:=0;
  while (i<=LongN)and(sw2=0) do
   if (Nombre[i] in Let) then
    i:=i+1
   else
   begin
    Showmessage('El nombre solo debe contener letras');
    Nombre:=' ';
```

```
    Edit1.Clear;
    Edit1.SetFocus;
    sw2:=1;
   end;
 end;
 if (sw1=0)and(sw2=0) then
 begin
  Estudiantes.SetKey;
  Estudiantes.FieldByName('Cod_Est').AsString:=Codigo;
  Estudiantes.GotoKey;
  if (Estudiantes.GotoKey=True) then
  begin
   showmessage('Código ya existe');
   Edit2.Clear;
   Edit2.SetFocus;
  end
  else
  begin
   panel1.visible:=True;
   Edit3.SetFocus;
   SpeedButton1N.Enabled:=False;
   SpeedButton3N.Enabled:=False;
  end;
 end;
end;
```

PROCEDIMIENTO ADICIONAR_USUARIO. Inicializa los datos del nuevo usuario en la tabla Estudiantes de la base de datos.

```
procedure TForm7.SpeedButton2NClick(Sender: TObject);
var
 Clave1,Clave2:String[10];
begin
 Clave1:=Edit3.Text;
 Clave2:=Edit4.Text;
 If(Clave1<>Clave2) Then
 begin
  ShowMessage('Rectifique su clave');
  Edit4.Clear;
  Edit4.SetFocus;
 end
 else
 begin
  Estudiantes.DisableControls;
  Estudiantes.Last;
```

```
  Estudiantes.Insert;
  Estudiantes.Edit;
  Estudiantes.Fields[0].AsString:=Codigo;
  Estudiantes.Fields[1].AsString:=Nombre;
  Estudiantes.Fields[2].AsString:=Clave1;
  Estudiantes.Fields[3].AsInteger:=0;
  Estudiantes.Fields[4].AsString:=' ';
  Estudiantes.Fields[5].AsInteger:=0;
  Estudiantes.Fields[6].AsInteger:=1;
  Estudiantes.post;
  Estudiantes.EnableControls;
  panel1.visible:=false;
  SpeedButton3N.Enabled:=True;
  PageControl1.ActivePage:=PInformacion;
  label6I.caption:=Estudiantes.FieldByName('Nom_Est').AsString;
  label7I.caption:=Codigo;
  Codigo1:=Codigo;
  label8I.caption:=Estudiantes.FieldByName('Ult_Lecc').AsString;
  Punt:=Estudiantes.FieldByName('Punt_Est').AsFloat;
  FormatFloat('0.00',Punt);
  label9I.caption:=PuntS;
  Estr:=Estudiantes.FieldByName('Estr_Est').As Integer;
  EstrS:=IntToStr(Estr);
  label10I.caption:=EstrS;
  Niv:=Estudiantes.FieldByName('Niv_Est').AsInteger;
  NivS:=IntToStr(Niv);
  label11I.caption:=NivS;
  Estudiantes.Refresh;
  Estudiantes.Close;
 end;
end;
```

PROCEDIMIENTO ANTIGUO_USUARIO. Controla la entrada de un antiguo usuario, verificando sus datos en la base de datos.

```
procedure TForm7.SpeedButton1AClick(Sender: TObject);
var
  Clave1,Clave2:String[10];
begin
 Num:=['0','1','2','3','4','5','6','7','8','9'];
 Codigo1:=Edit1A.Text;
 Clave1:=Edit2A.Text;
 LongC:=Length(Codigo1);
 Cont:=0;
 if (Clave1=' ') or (Codigo1=' ') then
```

```
 showmessage('Por favor,digite los datos completos')
else
begín
 i:=1;
 sw1:=0;
 while (i<=LongC)and(sw1=0) do
  if (Codigo1[i] in Num) then
   i:=i+1
  else
  begin
   Showmessage('El código solo debe contener números');
   Codigo1:=' ';
   Edit2A.Clear;
   Edit2A.SetFocus;
   sw1:=1;
  end;
end;
if (sw1=0) then
begin
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=(Codigo1);
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
  Clave2:=Estudiantes.FieldByName('Clave').AsString;
 if(Clave1<>Clave2)then
  begin
    showmessage('Clave errada');
    cont:=cont+1;
    edit2A.Clear;
    Edit2A.SetFocus;
  end;
  if (Cont>3) then
  begin
   showmessage('Ya no puedes entrar');
   Cont:=0;
   PageControl1.ActivePage:=POpciones;
  end;
  if (clave1=clave2) then
  begin
   PageControl1.ActivePage:=PInformacion;
   label6I.caption:=Estudiantes.FieldByName('Nom_Est').AsString;
   label7I.caption:=Codigo1;
   label8I.caption:=Estudiantes.FieldByName('Ult_Lecc').AsString;
   Punt:=Estudiantes.FieldByName('Punt_Est').AsFloat;
   PuntS:=FormatFloat('0.00',Punt);
   label9I.caption:=PuntS;
   Estr:=Estudiantes.FieldByName('Estr_Est').AsInteger;
   EstrS:=IntToStr(Estr);
   label10I.caption:=EstrS;
```

```
    Niv:=Estudiantes.FieldByName('Niv_Est').AsInteger;
    NivS:=IntToStr(Niv);
    label11I.caption:=NivS;
   end;
  end
  else

  begin
   showmessage('Código no existe');
   edit1A.Clear;
   Edit1A.SetFocus;
   edit2A.Clear;
  end;
 end;
 Estudiantes.Close;
end;
```

3.2 MODULO LECCIONES.

LECCIÓN HISTORIA

PROCEDIMIENTO INICIAR_HISTORIA. Cuando un usuario visita la lección Historia, actualiza los datos de éste en las tablas Estudiantes e Historia.

```
procedure TForm7.BHistoriaClick(Sender: TObject);
begin
 form7.hide;
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
  Estudiantes.Edit;
  Estudiantes.Fields[4].AsString:='Historia';
  Estudiantes.Post;
  Estudiantes.EnableControls;
  Estudiantes.Close;
```

```
 end;
 Leccion_Historia.Open;
 Leccion_Historia.SetKey;
 Leccion_Historia.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_Historia.GotoKey;
 if (Leccion_Historia.GotoKey=True) then
 begin
  Leccion_Historia.Edit;

Leccion_Historia.Fields[1].AsInteger:=Leccion_Historia.Fields[1].AsInteg
er+1;
  Leccion_Historia.Post;
  Leccion_Historia.EnableControls;
  Leccion_Historia.Close;
 end
 else
  begin
   Leccion_Historia.DisableControls;
   Leccion_Historia.Last;
   Leccion_Historia.Insert;
   Leccion_Historia.Edit;
   Leccion_Historia.Fields[0].AsString:=Codigo1;
   Leccion_Historia.Fields[1].AsInteger:=1;
   Leccion_Historia.post;
   Leccion_Historia.EnableControls;
   Leccion_Historia.Close;
  end;
 FEntrada.Show;
end;
```

LECCIÓN TEORIA BASICA.

PROCEDIMIENTO INICIAR_TEORIA_BASICA. Cuando un usuario visita la lección Teoría Básica, actualiza los datos de éste en las tablas Estudiantes y LecciónB.

```
procedure TForm7.BTeoriaBClick(Sender: TObject);
begin
 form7.hide;
 FEscogTeoria.show;
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
```

```
  Estudiantes.Edit;
  Estudiantes.Fields[4].AsString:='Teoría Básica';
  Estudiantes.Post;
  Estudiantes.EnableControls;
  Estudiantes.Close;
 end;
 Leccion_TeoriaB.Open;
 Leccion_TeoriaB.SetKey;
 Leccion_TeoriaB.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_TeoriaB.GotoKey;
 if (Leccion_TeoriaB.GotoKey=True) then
 begin
  Leccion_TeoriaB.Edit;

Leccion_TeoriaB.Fields[1].AsInteger:=Leccion_TeoriaB.Fields[1].AsInteg
er+1;
  Leccion_TeoriaB.Post;
  Leccion_TeoriaB.EnableControls;
  Leccion_TeoriaB.Close;
 end
 else
  begin
   Leccion_TeoriaB.DisableControls;
   Leccion_TeoriaB.Last;
   Leccion_TeoriaB.Insert;
   Leccion_TeoriaB.Edit;
   Leccion_TeoriaB.Fields[0].AsString:=Codigo1;
   Leccion_TeoriaB.Fields[1].AsInteger:=1;
   Leccion_TeoriaB.Fields[2].AsFloat:=0;
   Leccion_TeoriaB.Fields[3].AsInteger:=0;

Leccion_TeoriaB.Fields[4].AsString:='0000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000';

Leccion_TeoriaB.Fields[5].AsString:='0000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000';

Leccion_TeoriaB.Fields[6].AsString:='0000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000';

Leccion_TeoriaB.Fields[7].AsString:='0000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
```

```
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000';
  Leccion_TeoriaB.post;
G   Leccion_TeoriaB.EnableControls;
  Leccion_TeoriaB.Close;
end;
end;
```

PROCEDIMIENTO TABLA_OR. Controlan que el usuario termine de llenar correctamente la tabla de verdad para la operación Or del Álgebra Booleana que se le propone en ese módulo.

```
procedure TFOperacAB2.B0Click(Sender: TObject);
begin
if (NTerm0<=8)then
begin
 with StringGrid1 do
 begin
  Cells[3,NTerm0]:='0';
  if ((Cells[0,NTerm0]='1') or (Cells[1,NTerm0]='1') or
(Cells[2,NTerm0]='1')) then
  begin
showmessage('Las reglas de la operación Or indican que cuando una de
las variables de entrada tiene de valor uno (1),la salida es uno (1)');
   Cells[3,NTerm0]:='';
  end
  else
   NTerm0:=NTerm0+1;
  if (NTerm0=9) then
  begin
  label1.Visible:=False;
  label2.Visible:=False;
  label3.Visible:=False;
  label4.Visible:=False;
  B0.visible:=False;
  B1.visible:=False;
  Panel1.Visible:=True;
  label6.Visible:=True;
  label7.Visible:=True;
  label8.Visible:=True;
  BRegresar.Enabled:=True;
  Cerrar1.Enabled:=True;
 end;
```

```
 end;
end;
end;



procedure TFOperacAB2.B1Click(Sender: TObject);
begin
if (NTerm0<=8)then
begin
 with StringGrid1 do
 begin
  Cells[3,NTerm0]:='1';
  if ((Cells[0,NTerm0]='0') and (Cells[1,NTerm0]='0') and
(Cells[2,NTerm0]='0')) then
  begin
   showmessage('Las reglas de la operación Or indican que cuando todas
las variables de entrada tiene de valor cero (0),la salida es cero (0)');
   Cells[3,NTerm0]:='';
  end
  else
   NTerm0:=NTerm0+1;
  if (NTerm0=9) then
  begin
  label1.Visible:=False;
  label2.Visible:=False;
  label3.Visible:=False;
  label4.Visible:=False;
  B0.visible:=False;
  B1.visible:=False;
  Panel1.Visible:=True;
  label6.Visible:=True;
  label7.Visible:=True;
  label8.Visible:=True;
  Cerrar1.Enabled:=True;
  BRegresar.Enabled:=True;
 end;
 end;
end;
end;
```

PROCEDIMIENTO TABLA_AND. Controla que el usuario termine de llenar correctamente la tabla de verdad para la operación And del Álgebra Booleana que se le propone en ese módulo.

```
procedure TFOperacABAnd2.B1Click(Sender: TObject);
begin
if (NTerm1<=8)then
begin
 with StringGrid1 do
 begin
  Cells[3,NTerm1]:='1';
  if ((Cells[0,NTerm1]='0') or (Cells[1,NTerm1]='0') or
(Cells[2,NTerm1]='0')) then
  begin
   showmessage('Las reglas de la operación And indican que cuando una
de las variables de entrada tiene de valor cero (0),la salida es cero (0)');
   Cells[3,NTerm1]:='';
  end
  else
   NTerm1:=NTerm1+1;
  if (NTerm1=9) then
  begin
  label3.visible:=false;
  label4.Visible:=False;
  B0.visible:=False;
  B1.visible:=False;
  Panel1.Visible:=True;
  label6.Visible:=True;
  label7.Visible:=True;
  label8.Visible:=True;
  Cerrar1.Enabled:=True;
  BRegresar.Visible:=True;
 end;
 end;
 end;
end;

procedure TFOperacABAnd2.B0Click(Sender: TObject);
begin
if (NTerm1<=8)then
begin
 with StringGrid1 do
 begin
  Cells[3,NTerm1]:='0';
  if ((Cells[0,NTerm1]='1') and (Cells[1,NTerm1]='1') and
(Cells[2,NTerm1]='1')) then
  begin
   showmessage('Las reglas de la operación And indican que cuando
todas las variables de entrada tiene de valor uno (1),la salida es uno
(1)');
   Cells[3,NTerm1]:='';
  end
  else
   NTerm1:=NTerm1+1;
  if (NTerm1=9) then
```

```
  begin
  label1.Visible:=False;
  label2.Visible:=False;
  label3.Visible:=False;
  label4.Visible:=False;
  B0.visible:=False;
  B1.visible:=False;
  Panel1.Visible:=True;
  Cerrar1.Enabled:=True;
  BRegresar.Visible:=True;
  label6.Visible:=True;
  label7.Visible:=True;
  label8.Visible:=True;
 end;
 end;
end;
end;
```

PROCEDIMIENTO PROPUESTOS_OPERACIONES. Asigna al usuario un ejercicio para evaluarlo en el modulo Operaciones del Álgebra Booleana de la lección Teoría Básica, de acuerdo al nivel en el que se encuentre éste.

```
procedure TFOperacABAnd.N1Click(Sender: TObject);
var
 sw:boolean;
begin
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
   Estudiantes.Edit;
   niv:=Estudiantes.Fields[6].AsInteger;
 end;
 Estudiantes.Close;
 Leccion_TeoriaB.Open;
 Leccion_TeoriaB.SetKey;
 Leccion_TeoriaB.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_TeoriaB.GotoKey;
 Leccion_TeoriaB.Edit;
 EjerSiNo:=Leccion_TeoriaB.Fields[5].AsString;
 Leccion_TeoriaB.Close;
 Ejercicios_Propuestos.Open;
```

```
 Ejercicios_Propuestos.DisableControls;
 sw:=False;
 Ejercicios_Propuestos.First;
 while ((not(Ejercicios_Propuestos.EOF)) and (sw=False)) do
 begin
  Ejercicios_Propuestos.Edit;
              if  (Ejercicios_Propuestos.Fields[0].AsString=  'Operaciones
Booleanas')and (Ejercicios_Propuestos.Fields[2].Asinteger=niv) then
   if (EjerSiNo[Ejercicios_Propuestos.Fields[7].AsInteger]='0') then
    sw:=True;
  if sw=false then
   Ejercicios_Propuestos.Next;
 end;
 if (sw) then
 begin
  PropBas3:=PropBas3+1;
  Ecuac:=Ejercicios_Propuestos.Fields[1].asstring;
  FEvaluarOperac.Label2.Caption:='X= '+Ecuac;
  FEvaluarOperac.Show;
  FOperacABAnd.Hide;
 end
 else
  showmessage('No hay más ejercicios sobre este tema en este nivel.');
end;
```

PROCEDIMINTO AND_OR. Permite determinar si la evaluación es con la operación tipo Or ó con la And y cuantas variables se van a utilizar.

```
procedure AndOr(Ecuac:string;var Tipo:string;var CantV:integer);
var
 i:integer;
 sw:boolean;
begin
 sw:=False;
 CantV:=0;
 for i:=1 to length(Ecuac) do
  if (Ecuac[i]='+') then
   sw:=True
  else
   CantV:=CantV+1;
 if (sw) then
  Tipo:='Or'
```

```
 else
  Tipo:='And';
end;
```

PROCEDIMIENTO NAND_NOR. Permite determinar si la evaluación es con la operación tipo Or ó con la And y cuantas variables se van a utilizar.

```
procedure NandNor (Ecuac :string ;var Tipo: string; var CantV: integer);
var
 i,j:integer;
 sw1:boolean;
 Expres:string;
begin
 sw1:=False;
 CantV:=0;
 Expres:='';
 for i:=1 to length(Ecuac) do
  if ((Ecuac[i]='-') and (Ecuac[i+1]='(')) then
  begin
   Expres:=Expres+Ecuac[i]+Ecuac[i+1];
   j:=i+2;
   while ((Ecuac[j]<>')') and (j<=length(Ecuac))) do
   begin
    Expres:=Expres+Ecuac[j];
    if (Ecuac[j]='+') then
     sw1:=True
    else
     CantV:=CantV+1;
    j:=j+1;
   end;
  end;
 if (sw1) then
  Tipo:='Nor'
 else
  Tipo:='Nand';
end;
```

PROCEDIMIENTO EVALUAR_OPERAC1. Este procedimiento hace el seguimiento a la evaluación en el módulo Operaciones del

Álgebra Booleana, cuando al hacer el llenado de la tabla que se le solicita al usuario con ceros y unos, él selecciona 1.

```
procedure TFEvaluarOperac.B1Click(Sender: TObject);
var
  Comp,i,j,ceros,num0:integer;
  sw,sw0:Boolean;
begin
 panel1.Visible:=False;
 panel2.Visible:=False;
 panel3.Visible:=False;
 if (sw_t=false) then
 begin
  if (NTerm<StringGrid1.RowCount) then
  begin
   StringGrid1.Cells[NCol,NTerm]:='1';
   if (NCol<stringgrid1.ColCount-1) then
    Acumula:=Acumula+'1';
   NCol:=NCol+1;
   if (NCol=stringgrid1.ColCount-1) then
   begin
    if (NTerm=1)then
    begin
     VTerm[NTerm-1]:=Acumula;
     acumula:='';
    end
    else
    begin
     sw:=False;
     i:=0;
     while ((i<=NTerm) and (sw=False))do
     begin
      Comp:=CompareStr(VTerm[i],Acumula);
      if (Comp=0) then
      begin
       panel3.Visible:=True;
       label14.Caption:=Acumula;
       NError:=NError+1;
       NCol:=0;
       sw:=True;
       for j:=0 to StringGrid1.ColCount-1 do
        StringGrid1.Cells[j,NTerm]:='';
       acumula:='';
      end
      else
       i:=i+1;
     end;//while
```

```
   if (sw=False) then
   begin
    VTerm[NTerm-1]:=Acumula;
    Acumula:='';
   end;
  end;
 end;
 if (NCol=stringgrid1.ColCount-1) then
 begin
  NTerm:=NTerm+1;
  NCol:=0;
 end;
 if (NTerm=stringgrid1.RowCount) then
  if (Cantv<=4) then
  begin
   sw_t:=True;
   NCol:=stringgrid1.ColCount-1;
   NTerm:=1;
  end
  else
  begin
   NCol:=0;
   NTerm2:=1;
  end;
end
else
 if (NTerm2<StringGrid2.RowCount) then
 begin
  StringGrid2.Cells[NCol,NTerm2]:='1';
  if (NCol<stringgrid2.ColCount-1) then
   Acumula:=Acumula+'1';
  NCol:=NCol+1;
  if (NCol=stringgrid2.ColCount-1) then
  begin
   sw:=False;
   i:=0;
   while ((i<=NTerm+NTerm2-2) and (sw=False))do
   begin
    Comp:=CompareStr(VTerm[i],Acumula);
    if (Comp=0) then
    begin
     panel3.Visible:=True;
     label14.Caption:=Acumula;
     NError:=NError+1;
     NCol:=0;
     sw:=True;
     for j:=0 to StringGrid2.ColCount-1 do
      StringGrid2.Cells[j,NTerm2]:='';
     acumula:='';
    end
    else
```

```
      i:=i+1;
    end;//while
    if (sw=False) then
    begin
     VTerm[NTerm+NTerm2-2]:=Acumula;
     Acumula:='';
    end;
   end;//
   if (NCol=stringgrid2.ColCount-1) then
   begin
    NTerm2:=NTerm2+1;
    NCol:=0;
    if (NTerm2=stringgrid2.RowCount) then
    begin
     sw_t:=True;
     NCol:=stringgrid2.ColCount-1;
     NTerm:=1;
     NTerm2:=20;
    end;
   end;
  end;
 end
 else
  if (sw_t) then
 begin
  if (NTerm<StringGrid1.RowCount) then
  begin
   StringGrid1.Cells[NCol,NTerm]:='1';
   if (Tipo='And') then
   begin
    Term:=VTerm[NTerm-1];
    sw0:=False;
    for j:=1 to length(Term) do
     if (Term[j]='0') then
      sw0:=True;
    if (sw0=False) then
    begin
     if (StringGrid1.Cells[NCol,NTerm]='1') then
     begin
      panel1.Visible:=True;
      label6.Caption:=StringGrid1.Cells[NCol,NTerm];
     end
     else
     begin
      panel2.Visible:=True;
      label10.Caption:='1';
      NError:=NError+1;
      StringGrid1.Cells[NCol,NTerm]:='1';
     end;
    end
    else
```

```
  if (sw0) then
  begin
   if (StringGrid1.Cells[NCol,NTerm]='1') then
   begin
    panel2.Visible:=True;
    label10.Caption:='0';
    NError:=NError+1;
    StringGrid1.Cells[NCol,NTerm]:='0';
   end
   else
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid1.Cells[NCol,NTerm];
   end;
  end;
end
else
 if (Tipo='Or') then
 begin
  ceros:=StringGrid1.ColCount-1;
  num0:=0;
  Term:=VTerm[NTerm-1];
  for j:=1 to length(Term) do
   if (Term[j]='0') then
    num0:=num0+1;
  if (num0=ceros) then
  begin
   if (StringGrid1.Cells[NCol,NTerm]='0') then
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid1.Cells[NCol,NTerm];
   end
   else
   begin
    panel2.Visible:=True;
    label10.Caption:='0';
    NError:=NError+1;
    StringGrid1.Cells[NCol,NTerm]:='0';
   end;
  end
  else
  begin
   if (StringGrid1.Cells[NCol,NTerm]='0') then
   begin
    panel2.Visible:=True;
    label10.Caption:='1';
    NError:=NError+1;
    StringGrid1.Cells[NCol,NTerm]:='1';
   end
   else
   begin
```

```
   panel1.Visible:=True;
   label6.Caption:=StringGrid1.Cells[NCol,NTerm];
  end;
 end;
end
else
 if (Tipo='Nand') then
 begin
  Term:=VTerm[NTerm-1];
  sw0:=False;
  for j:=1 to length(Term) do
   if (Term[j]='0') then
    sw0:=True;
  if (sw0=False) then
  begin
   if (StringGrid1.Cells[NCol,NTerm]='1') then
   begin
    panel2.Visible:=True;
    label10.Caption:='0';
    NError:=NError+1;
    StringGrid1.Cells[NCol,NTerm]:='0';
   end
   else
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid1.Cells[NCol,NTerm];
   end;
  end
  else
   if (sw0) then
   begin
    if (StringGrid1.Cells[NCol,NTerm]='1') then
    begin
     panel1.Visible:=True;
     label6.Caption:=StringGrid1.Cells[NCol,NTerm];
    end
    else
    begin
     panel2.Visible:=True;
     label10.Caption:='1';
     NError:=NError+1;
     StringGrid1.Cells[NCol,NTerm]:='1';
    end;
   end;
 end
 else
  if (Tipo='Nor') then
  begin
   ceros:=StringGrid1.ColCount-1;
   num0:=0;
   Term:=VTerm[NTerm-1];
```

```
    for j:=1 to length(Term) do
     if (Term[j]='0') then
      num0:=num0+1;
    if (num0=ceros) then
    begin
     if (StringGrid1.Cells[NCol,NTerm]='0') then
     begin
      panel2.Visible:=True;
      label10.Caption:='1';
      NError:=NError+1;
      StringGrid1.Cells[NCol,NTerm]:='1';
     end
     else
     begin
      panel1.Visible:=True;
      label6.Caption:=StringGrid1.Cells[NCol,NTerm];
     end;
    end
    else
    begin
     if (StringGrid1.Cells[NCol,NTerm]='0') then
     begin
      panel1.Visible:=True;
      label6.Caption:=StringGrid1.Cells[NCol,NTerm];
     end
     else
     begin
      panel2.Visible:=True;
      label10.Caption:='0';
      NError:=NError+1;
      StringGrid1.Cells[NCol,NTerm]:='0';
     end;
    end;
   end;
   NTerm:=NTerm+1;
   if (NTerm=stringgrid1.RowCount) then
    if (cantv<=4) then
    begin
     BRegresar.Visible:=true;
     B0.Visible:=false;
     B1.Visible:=false;
    end
    else
     NTerm2:=1;
end
else
if (NTerm2<StringGrid2.RowCount) then
begin
 StringGrid2.Cells[NCol,NTerm2]:='1';
 if (Tipo='And') then
 begin
```

```
 Term:=VTerm[NTerm+NTerm2-2];
 sw0:=False;
 for j:=1 to length(Term) do
  if (Term[j]='0') then
   sw0:=True;
 if (sw0=False) then
 begin
  if (StringGrid2.Cells[NCol,NTerm2]='1') then
  begin
   panel1.Visible:=True;
   label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
  end
  else
  begin
   panel2.Visible:=True;
   label10.Caption:='1';
   NError:=NError+1;
   StringGrid2.Cells[NCol,NTerm2]:='1';
  end;
 end
 else
  if (sw0) then
  begin
   if (StringGrid2.Cells[NCol,NTerm2]='1') then
   begin
    panel2.Visible:=True;
    label10.Caption:='0';
    NError:=NError+1;
    StringGrid2.Cells[NCol,NTerm2]:='0';
   end
   else
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
   end;
  end;
end
else
 if (Tipo='Or') then
 begin
  ceros:=StringGrid2.ColCount-1;
  num0:=0;
  Term:=VTerm[NTerm+NTerm2-2];
  for j:=1 to length(Term) do
   if (Term[j]='0') then
    num0:=num0+1;
  if (num0=ceros) then
  begin
   if (StringGrid2.Cells[NCol,NTerm2]='0') then
   begin
    panel1.Visible:=True;
```

```
   label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
  end
  else
  begin
   panel2.Visible:=True;
   label10.Caption:='0';
   NError:=NError+1;
   StringGrid2.Cells[NCol,NTerm2]:='0';
  end;
 end
 else
 begin
  if (StringGrid2.Cells[NCol,NTerm2]='0') then
  begin
   panel2.Visible:=True;
   label10.Caption:='1';
   NError:=NError+1;
   StringGrid2.Cells[NCol,NTerm2]:='1';
  end
  else
  begin
   panel1.Visible:=True;
   label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
  end;
 end;
end
else
 if (Tipo='Nand') then
 begin
  Term:=VTerm[NTerm+NTerm2-2];
  sw0:=False;
  for j:=1 to length(Term) do
   if (Term[j]='0') then
    sw0:=True;
  if (sw0=False) then
  begin
   if (StringGrid2.Cells[NCol,NTerm2]='1') then
   begin
    panel2.Visible:=True;
    label10.Caption:='0';
    NError:=NError+1;
    StringGrid2.Cells[NCol,NTerm2]:='0';
   end
   else
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
   end;
  end
  else
   if (sw0) then
```

```
  begin
   if (StringGrid2.Cells[NCol,NTerm2]='1') then
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
   end
   else
   begin
    panel2.Visible:=True;
    label10.Caption:='1';
    NError:=NError+1;
    StringGrid2.Cells[NCol,NTerm2]:='1';
   end;
  end;
end
else
 if (Tipo='Nor') then
 begin
  ceros:=StringGrid2.ColCount-1;
  num0:=0;
  Term:=VTerm[NTerm+NTerm2-2];
  for j:=1 to length(Term) do
   if (Term[j]='0') then
    num0:=num0+1;
  if (num0=ceros) then
  begin
   if (StringGrid2.Cells[NCol,NTerm2]='0') then
   begin
    panel2.Visible:=True;
    label10.Caption:='1';
    NError:=NError+1;
    StringGrid2.Cells[NCol,NTerm2]:='1';
   end
   else
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
   end;
  end
  else
  begin
   if (StringGrid2.Cells[NCol,NTerm2]='0') then
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
   end
   else
   begin
    panel2.Visible:=True;
    label10.Caption:='0';
    NError:=NError+1;
```

```
        StringGrid2.Cells[NCol,NTerm2]:='0';
       end;
      end;
     end;
     NTerm2:=NTerm2+1;
     if (NTerm2=stringgrid2.RowCount) then
     begin
      BRegresar.Visible:=true;
      B0.Visible:=false;
      B1.Visible:=false;
     end;
  end;
 end;
end;
```

PROCEDIMIENTO EVALUAR_OPERAC2 Este procedimiento hace el seguimiento a la evaluación en el módulo Operaciones del Álgebra Booleana, cuando al hacer el llenado de la tabla que se le solicita al usuario con ceros y unos, él selecciona 0.

```
procedure TFEvaluarOperac.B0Click(Sender: TObject);
var
  Comp,i,j,ceros,num0:integer;
  sw,sw0:Boolean;
begin
 panel1.Visible:=False;
 panel2.Visible:=False;
 panel3.Visible:=False;
 if (sw_t=false) then
 begin
  if (NTerm<StringGrid1.RowCount) then
  begin
   StringGrid1.Cells[NCol,NTerm]:='0';
   if (NCol<stringgrid1.ColCount-1) then
    Acumula:=Acumula+'0';
   NCol:=NCol+1;
   if (NCol=stringgrid1.ColCount-1) then
   begin
    if (NTerm=1)then
    begin
     VTerm[NTerm-1]:=Acumula;
     acumula:='';
    end
    else
```

```
 begin
  sw:=False;
  i:=0;
  while ((i<=NTerm) and (sw=False))do
  begin
   Comp:=CompareStr(VTerm[i],Acumula);
   if (Comp=0) then
   begin
    panel3.Visible:=True;
    label14.Caption:=Acumula;
    NError:=NError+1;
    NCol:=0;
    sw:=True;
    for j:=0 to StringGrid1.ColCount-1 do
     StringGrid1.Cells[j,NTerm]:='';
    acumula:='';
   end
   else
    i:=i+1;
  end;//while
  if (sw=False) then
  begin
   VTerm[NTerm-1]:=Acumula;
   Acumula:='';
  end;
 end;// NTerm
end;//NCol
if (NCol=stringgrid1.ColCount-1) then
begin
 NTerm:=NTerm+1;
 NCol:=0;
end;
if (NTerm=stringgrid1.RowCount) then
 if (Cantv<=4) then
 begin
  sw_t:=True;
  NCol:=stringgrid1.ColCount-1;
  NTerm:=1;
 end
 else
 begin
  NCol:=0;
  NTerm2:=1;
 end;
end
else
 if (NTerm2<StringGrid2.RowCount) then
 begin
  StringGrid2.Cells[NCol,NTerm2]:='0';
  if (NCol<stringgrid2.ColCount-1) then
   Acumula:=Acumula+'0';
```

```
  NCol:=NCol+1;
  if (NCol=stringgrid2.ColCount-1) then
  begin
   sw:=False;
   i:=0;
   while ((i<=NTerm+NTerm2-2) and (sw=False))do
   begin
    Comp:=CompareStr(VTerm[i],Acumula);
    if (Comp=0) then
    begin
     panel3.Visible:=True;
     label14.Caption:=Acumula;
     NError:=NError+1;
     NCol:=0;
     sw:=True;
     for j:=0 to StringGrid2.ColCount-1 do
      StringGrid2.Cells[j,NTerm2]:='';
     acumula:='';
    end
    else
     i:=i+1;
   end;//while
   if (sw=False) then
   begin
    VTerm[NTerm+NTerm2-2]:=Acumula;
    Acumula:='';
   end;
  end;
  if (NCol=stringgrid2.ColCount-1) then
  begin
   NTerm2:=NTerm2+1;
   NCol:=0;
   if (NTerm2=stringgrid2.RowCount) then
   begin
    sw_t:=True;
    NCol:=stringgrid2.ColCount-1;
    NTerm:=1;
    NTerm2:=20;
   end;
  end;
 end;
end
else
 if (sw_t) then
 begin
  if (NTerm<StringGrid1.RowCount) then
  begin
   StringGrid1.Cells[NCol,NTerm]:='0';
   if (Tipo='And') then
   begin
    Term:=VTerm[NTerm-1];
```

```
 sw0:=False;
 for j:=1 to length(Term) do
  if (Term[j]='0') then
   sw0:=True;
 if (sw0=False) then
 begin
  if (StringGrid1.Cells[NCol,NTerm]='1') then
  begin
   panel1.Visible:=True;
   label6.Caption:=StringGrid1.Cells[NCol,NTerm];
  end
  else
  begin
   panel2.Visible:=True;
   label10.Caption:='1';
   NError:=NError+1;
   StringGrid1.Cells[NCol,NTerm]:='1';
  end;
 end
 else
  if (sw0) then
  begin
   if (StringGrid1.Cells[NCol,NTerm]='1') then
   begin
    panel2.Visible:=True;
    label10.Caption:='0';
    NError:=NError+1;
    StringGrid1.Cells[NCol,NTerm]:='0';
   end
   else
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid1.Cells[NCol,NTerm];
   end;
  end;
end
else
 if (Tipo='Or') then
 begin
  ceros:=StringGrid1.ColCount-1;
  num0:=0;
  Term:=VTerm[NTerm-1];
  for j:=1 to length(Term) do
   if (Term[j]='0') then
    num0:=num0+1;
  if (num0=ceros) then
  begin
   if (StringGrid1.Cells[NCol,NTerm]='0') then
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid1.Cells[NCol,NTerm];
```

```
 end
 else
 begin
  panel2.Visible:=True;
  label10.Caption:='0';
  NError:=NError+1;
  StringGrid1.Cells[NCol,NTerm]:='0';
 end;
end
else
begin
 if (StringGrid1.Cells[NCol,NTerm]='0') then
 begin
  panel2.Visible:=True;
  label10.Caption:='1';
  NError:=NError+1;
  StringGrid1.Cells[NCol,NTerm]:='1';
 end
 else
 begin
  panel1.Visible:=True;
  label6.Caption:=StringGrid1.Cells[NCol,NTerm];
 end;
end;
end
else
if (Tipo='Nand') then
begin
 Term:=VTerm[NTerm-1];
 sw0:=False;
 for j:=1 to length(Term) do
  if (Term[j]='0') then
   sw0:=True;
 if (sw0=False) then
 begin
  if (StringGrid1.Cells[NCol,NTerm]='1') then
  begin
   panel2.Visible:=True;
   label10.Caption:='0';
   NError:=NError+1;
   StringGrid1.Cells[NCol,NTerm]:='0';
  end
  else
  begin
   panel1.Visible:=True;
   label6.Caption:=StringGrid1.Cells[NCol,NTerm];
  end;
 end
 else
  if (sw0) then
  begin
```

```
   if (StringGrid1.Cells[NCol,NTerm]='1') then
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid1.Cells[NCol,NTerm];
   end
   else
   begin
    panel2.Visible:=True;
    label10.Caption:='1';
    NError:=NError+1;
    StringGrid1.Cells[NCol,NTerm]:='1';
   end;
  end;
end
else
 if (Tipo='Nor') then
 begin
  ceros:=StringGrid1.ColCount-1;
  num0:=0;
  Term:=VTerm[NTerm-1];
  for j:=1 to length(Term) do
   if (Term[j]='0') then
    num0:=num0+1;
  if (num0=ceros) then
  begin
   if (StringGrid1.Cells[NCol,NTerm]='0') then
   begin
    panel2.Visible:=True;
    label10.Caption:='1';
    NError:=NError+1;
    StringGrid1.Cells[NCol,NTerm]:='1';
   end
   else
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid1.Cells[NCol,NTerm];
   end;
  end
  else
  begin
   if (StringGrid1.Cells[NCol,NTerm]='0') then
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid1.Cells[NCol,NTerm];
   end
   else
   begin
    panel2.Visible:=True;
    label10.Caption:='0';
    NError:=NError+1;
    StringGrid1.Cells[NCol,NTerm]:='0';
```

```
     end;
    end;
   end;
   NTerm:=NTerm+1;
   if (NTerm=stringgrid1.RowCount) then
    if (cantv<=4) then
    begin
     BRegresar.Visible:=true;
     B0.Visible:=false;
     B1.Visible:=false;
    end
    else
     NTerm2:=1;
end
else
 if (NTerm2<StringGrid2.RowCount) then
 begin
  StringGrid2.Cells[NCol,NTerm2]:='0';
  if (Tipo='And') then
  begin
   Term:=VTerm[NTerm+NTerm2-2];
   sw0:=False;
   for j:=1 to length(Term) do
    if (Term[j]='0') then
     sw0:=True;
   if (sw0=False) then
   begin
    if (StringGrid2.Cells[NCol,NTerm2]='1') then
    begin
     panel1.Visible:=True;
     label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
    end
    else
    begin
     panel2.Visible:=True;
     label10.Caption:='1';
     NError:=NError+1;
     StringGrid2.Cells[NCol,NTerm2]:='1';
    end;
   end
   else
    if (sw0) then
    begin
     if (StringGrid2.Cells[NCol,NTerm2]='1') then
     begin
      panel2.Visible:=True;
      label10.Caption:='0';
      NError:=NError+1;
      StringGrid2.Cells[NCol,NTerm2]:='0';
     end
     else
```

```
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
   end;
  end;
end
else
 if (Tipo='Or') then
 begin
  ceros:=StringGrid2.ColCount-1;
  num0:=0;
  Term:=VTerm[NTerm+NTerm2-2];
  for j:=1 to length(Term) do
   if (Term[j]='0') then
    num0:=num0+1;
  if (num0=ceros) then
  begin
   if (StringGrid2.Cells[NCol,NTerm2]='0') then
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
   end
   else
   begin
    panel2.Visible:=True;
    label10.Caption:='0';
    NError:=NError+1;
    StringGrid2.Cells[NCol,NTerm2]:='0';
   end;
  end
  else
  begin
   if (StringGrid2.Cells[NCol,NTerm2]='0') then
   begin
    panel2.Visible:=True;
    label10.Caption:='1';
    NError:=NError+1;
    StringGrid2.Cells[NCol,NTerm2]:='1';
   end
   else
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
   end;
  end;
 end
 else
  if (Tipo='Nand') then
  begin
   Term:=VTerm[NTerm+NTerm2-2];
   sw0:=False;
```

```
 for j:=1 to length(Term) do
  if (Term[j]='0') then
   sw0:=True;
 if (sw0=False) then
 begin
  if (StringGrid2.Cells[NCol,NTerm2]='1') then
  begin
   panel2.Visible:=True;
   label10.Caption:='0';
   NError:=NError+1;
   StringGrid2.Cells[NCol,NTerm2]:='0';
  end
  else
  begin
   panel1.Visible:=True;
   label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
  end;
 end
 else
  if (sw0) then
  begin
   if (StringGrid2.Cells[NCol,NTerm2]='1') then
   begin
    panel1.Visible:=True;
    label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
   end
   else
   begin
    panel2.Visible:=True;
    label10.Caption:='1';
    NError:=NError+1;
    StringGrid2.Cells[NCol,NTerm2]:='1';
   end;
  end;
end
else
 if (Tipo='Nor') then
 begin
  ceros:=StringGrid2.ColCount-1;
  num0:=0;
  Term:=VTerm[NTerm+NTerm2-2];
  for j:=1 to length(Term) do
   if (Term[j]='0') then
    num0:=num0+1;
  if (num0=ceros) then
  begin
   if (StringGrid2.Cells[NCol,NTerm2]='0') then
   begin
    panel2.Visible:=True;
    label10.Caption:='1';
    NError:=NError+1;
```

```
          StringGrid2.Cells[NCol,NTerm2]:='1';
         end
         else
         begin
          panel1.Visible:=True;
          label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
         end;
        end
        else
        begin
         if (StringGrid2.Cells[NCol,NTerm2]='0') then
         begin
          panel1.Visible:=True;
          label6.Caption:=StringGrid2.Cells[NCol,NTerm2];
         end
         else
         begin
          panel2.Visible:=True;
          label10.Caption:='0';
          NError:=NError+1;
          StringGrid2.Cells[NCol,NTerm2]:='0';
         end;
        end;
       end;
       NTerm2:=NTerm2+1;
       if (NTerm2=stringgrid2.RowCount) then
       begin
        BRegresar.Visible:=true;
        B0.Visible:=false;
        B1.Visible:=false;
       end;
    end;
  end;
end;
```

PROCEDIMIENTO ACTUALIZAR_OPERACIONES. Una vez realizada la evaluación del módulo Operaciones del Álgebra Booleana este procedimiento procede a realizar las actualizaciones necesarias en la base de datos Tablas.

```
procedure TFEvaluarOperac.BRegresarClick(Sender: TObject);
var
 PorcE,Punt,Desc,Puntaje,PuntE:Real;
 NumEjer:integer;
begin
```

```
panel1.Visible:=False;
panel2.Visible:=False;
if ((Tipo='And') or (Tipo='Or')) then
begin
 NumEjer:=FTablasdeV.Ejercicios_Propuestos.Fields[7].AsInteger;
 EjerSiNo[NumEjer]:='1';
 Punt:=FTablasdeV.Ejercicios_Propuestos.Fields[5].AsFloat;
 PorcE:=(NError*100)/TError;
 FTablasdeV.Ejercicios_Propuestos.Close;
 Leccion_TeoriaB.open;
 Leccion_TeoriaB.DisableControls;
 Leccion_TeoriaB.SetKey;
 Leccion_TeoriaB.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_TeoriaB.GotoKey;
 if (Leccion_TeoriaB.GotoKey=True) then
 begin
  Leccion_TeoriaB.Edit;
  if (PorcE<=60) then
  begin
   Punt:=((100-((NError*100)/(2*TError)))*Punt)/100;
   if (ResBas3=0) then
   begin
    Puntaje:=Punt;
    Leccion_TeoriaB.Fields[2].AsFloat:=
    Leccion_TeoriaB.Fields[2].AsFloat+Punt;
   end
   else
  begin
   Desc:=Punt-(Punt*0.05);
   Puntaje:=Desc;
   Leccion_TeoriaB.Fields[2].AsFloat:=
   Leccion_TeoriaB.Fields[2].AsFloat+Desc;
   end;
   Leccion_TeoriaB.Fields[5].AsString:=EjerSiNo;
   Leccion_TeoriaB.Post;
   Leccion_TeoriaB.EnableControls;
   Leccion_TeoriaB.Close;
   Estudiantes.Open;
   Estudiantes.DisableControls;
   Estudiantes.Setkey;
   Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
   Estudiantes.GotoKey;
   if (estudiantes.gotokey=true) then
   begin
    Estudiantes.Edit;
    Estudiantes.fields[6].AsInteger:=Estudiantes.fields[6].AsInteger+1;
    Estudiantes.fields[3].Asfloat:=Estudiantes.fields[3].Asfloat+Punt;
    PunT:=((Estudiantes.fields[3].AsFloat)/20);
    Estrellas:=Trunc(PunT);
    Estudiantes.fields[5].AsInteger:=Estrellas;
    Estudiantes.Post;
```

```
    Estudiantes.EnableControls;
    Estudiantes.Close;
   end;
   FFelicit.Memo2.Text:=FormatFloat('0.00',Puntaje);
   FFelicit.Show;
   FEvaluarOperac.Hide;
  end
  else
  begin
   PuntE:=(Leccion_TeoriaB.Fields[2].AsFloat)*0.1;
   Leccion_TeoriaB.Fields[2].AsFloat:=
   Leccion_TeoriaB.Fields[2].AsFloat-PuntE;
   Leccion_TeoriaB.Fields[5].AsString:=EjerSiNo;
   Leccion_TeoriaB.Post;
   Leccion_TeoriaB.EnableControls;
   Leccion_TeoriaB.Close;
   Estudiantes.Open;
   Estudiantes.DisableControls;
   Estudiantes.Setkey;
   Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
   Estudiantes.GotoKey;
   if (estudiantes.gotokey=true) then
   begin
    Estudiantes.Edit;
    Estudiantes.fields[6].AsInteger:=Estudiantes.fields[6].AsInteger-1;
Estudiantes.fields[3].Asfloat:=Estudiantes.fields[3].Asfloat-PuntE;
    PunT:=((Estudiantes.fields[3].AsFloat)/20);
    Estrellas:=Trunc(PunT);
    Estudiantes.fields[5].AsInteger:=Estrellas;
    Estudiantes.Post;
    Estudiantes.EnableControls;
    Estudiantes.Close;
   end;
   FEquivoc.Show;
   FEvaluarOperac.Hide;
  end;
 end;
end
else
if ((Tipo='Nand') or (Tipo='Nor')) then
begin
 NumEjer:=FAplicNand3.Ejercicios_Propuestos.Fields[7].AsInteger;
 EjerSiNo[NumEjer]:='1';
 Punt:=FAplicNand3.Ejercicios_Propuestos.Fields[5].AsFloat;
 PorcE:=(NError*100)/TError;
 FAplicNand3.Ejercicios_Propuestos.Close;
 Leccion_Aplicaciones.open;
 Leccion_Aplicaciones.DisableControls;
 Leccion_Aplicaciones.SetKey;
 Leccion_Aplicaciones.FieldByName('Cod_Est').AsString:=Codigo1;
Leccion_Aplicaciones.GotoKey;
```

```
  if (Leccion_Aplicaciones.GotoKey=True) then
  begin
   Leccion_Aplicaciones.Edit;
   if (PorcE<=60) then
   begin
    Punt:=((100-((NError*100)/(2*TError)))*Punt)/100;
    if (ResBas3=0) then
   begin
    Puntaje:=Punt;
    Leccion_Aplicaciones.Fields[2].AsFloat:=
    Leccion_Aplicaciones.Fields[2].AsFloat+Punt;
    end
    else
    begin
     Desc:=Punt-(Punt*0.05);
     Puntaje:=Desc;
     Leccion_Aplicaciones.Fields[2].AsFloat:=
     Leccion_Aplicaciones.Fields[2].AsFloat+Desc;
    end;
    Leccion_Aplicaciones.Fields[4].AsString:=EjerSiNo;
    Leccion_Aplicaciones.Post;
    Leccion_Aplicaciones.EnableControls;
    Leccion_Aplicaciones.Close;
    Estudiantes.Open;
    Estudiantes.DisableControls;
    Estudiantes.Setkey;
    Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
    Estudiantes.GotoKey;
    if (estudiantes.gotokey=true) then
    begin
     Estudiantes.Edit;
     Estudiantes.fields[6].AsInteger:=Estudiantes.fields[6].AsInteger+1;
Estudiantes.fields[3].Asfloat:=Estudiantes.fields[3].Asfloat+Punt;
     PunT:=((Estudiantes.fields[3].AsFloat)/20);
     Estrellas:=Trunc(PunT);
     Estudiantes.fields[5].AsInteger:=Estrellas;
     Estudiantes.Post;
     Estudiantes.EnableControls;
     Estudiantes.Close;
    end;
    FFelicit.Memo2.Text:=FormatFloat('0.00',Puntaje);
    FFelicit.Show;
    FEvaluarOperac.Hide;
   end
   else
   begin
    PuntE:=(Leccion_Aplicaciones.Fields[2].AsFloat)*0.1;
    Leccion_Aplicaciones.Fields[2].AsFloat:=
    Leccion_Aplicaciones.Fields[2].AsFloat-PuntE;
    Leccion_Aplicaciones.Fields[4].AsString:=EjerSiNo;
    Leccion_Aplicaciones.Post;
```

```
   Leccion_Aplicaciones.EnableControls;
   Leccion_Aplicaciones.Close;
   Estudiantes.Open;
   Estudiantes.DisableControls;
   Estudiantes.Setkey;
   Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
   Estudiantes.GotoKey;
   if (estudiantes.gotokey=true) then
   begin
    Estudiantes.Edit;
    Estudiantes.fields[6].AsInteger:=Estudiantes.fields[6].AsInteger-1;
Estudiantes.fields[3].Asfloat:=Estudiantes.fields[3].Asfloat-PuntE;
    PunT:=((Estudiantes.fields[3].AsFloat)/20);
    Estrellas:=Trunc(PunT);
    Estudiantes.fields[5].AsInteger:=Estrellas;
    Estudiantes.Post;
    Estudiantes.EnableControls;
    Estudiantes.Close;
   end;
   FEquivoc.Show;
   FEvaluarOperac.Hide;
  end;
 end;
end;
```

PROCEDIMIENTO PROPUESTOS_TABLA_VERDAD. Selecciona un ejercicio para evaluar al usuario en el módulo tablas de verdad de la lección Teoria Básica, de acuerdo a las características de éste.

```
procedure TFTablasdeV.N1Click(Sender: TObject);
var
 sw:boolean;
begin
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
   Estudiantes.Edit;
   niv:=Estudiantes.Fields[6].AsInteger;
 end;
 Estudiantes.Close;
```

```
 Leccion_TeoriaB.Open;
 Leccion_TeoriaB.SetKey;
 Leccion_TeoriaB.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_TeoriaB.GotoKey;
 Leccion_TeoriaB.Edit;
 EjerSiNo:=Leccion_TeoriaB.Fields[5].AsString;
 Leccion_TeoriaB.Close;
 Ejercicios_Propuestos.Open;
 Ejercicios_Propuestos.DisableControls;
 sw:=False;
 Ejercicios_Propuestos.First;
 while ((not(Ejercicios_Propuestos.EOF)) and (sw=False)) do
 begin
  Ejercicios_Propuestos.Edit;
  if (Ejercicios_Propuestos.Fields[0].AsString='Tablas de Verdad')and
(Ejercicios_Propuestos.Fields[2].Asinteger=niv) then
   if (EjerSiNo[Ejercicios_Propuestos.Fields[7].AsInteger]='0') then
    sw:=True;
  if sw=false then
   Ejercicios_Propuestos.Next;
 end;
 if (sw) then
 begin
  PropBas2:=PropBas2+1;
  Ecuac:=Ejercicios_Propuestos.Fields[1].asstring;
  FEvaluarTabladeV.Memo1.Text:='X= '+Ecuac;
  FEvaluarTabladeV.Show;
  FTablasdeV.Hide;
 end
 else
  showmessage('No hay más ejercicios sobre este tema en este nivel.');
end;
```

PROCEDIMIENTO RESUELTOS_TABLA_VERDAD. Selecciona un ejercicio resuelto para mostrar al usuario del tema Tablas de Verdad.

```
procedure TFTablasdeV.Acercade1Click(Sender: TObject);
var
  sw:boolean;
begin
Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
```

```
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
  Estudiantes.Edit;
  niv:=Estudiantes.Fields[6].AsInteger;
 end;
 Estudiantes.Close;
 Leccion_TeoriaB.Open;
 Leccion_TeoriaB.SetKey;
 Leccion_TeoriaB.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_TeoriaB.GotoKey;
 Leccion_TeoriaB.Edit;
 Leccion_TeoriaB.Close;
 Ejercicios_Resueltos.Open;
 Ejercicios_Resueltos.DisableControls;
 sw:=False;
 Ejercicios_Resueltos.First;
 while ((not(Ejercicios_Resueltos.EOF)) and (sw=False)) do
 begin
   Ejercicios_Resueltos.Edit;
   if          (Ejercicios_Resueltos.Fields[0].AsString='Tablas          de
Verdad')and(Ejercicios_Resueltos.Fields[1].Asinteger=niv) then
   begin
    if (Ejercicios_Resueltos.Fields[2].Asinteger=Num_Res) then
    begin
     sw:=True;
     Num_Res:=Num_Res+1;
    end
    else
     Ejercicios_Resueltos.Next;
   end
   else
    if (sw=false) then
     Ejercicios_Resueltos.Next;
 end;
 if (sw) then
 begin
  Ecuac:=Ejercicios_Resueltos.Fields[3].asstring;
  FEjerTVRes.Memo1.Text:='X=
'+Ejercicios_Resueltos.Fields[3].asString;
  FEjerTVRes.Show;
  FTablasdeV.Hide;
 end
 else
  showmessage('No hay más ejercicios sobre este tema en este nivel.');
end;
```

PROCEDIMIENTO EXPRESIÓN_TABLA. Permite deducir la expresión que resulta de la tabla de verdad que llena el usuario en el módulo Tablas de verdad de la lección Teoría Básica.

```
procedure EcuacTab(var Ecuac:string);
var
 i,long:integer;
 Expresion:string;
begin
 with FTablasdeV2.StringGrid1 do
 begin
  Expresion:='';
  for i:=1 to 8 do
  begin
   if (Cells[3,i]='1') then
   begin
    if (Cells[0,i]='1') then
     Expresion:=Expresion+'A'
    else
     Expresion:=Expresion+'-A';
    if (Cells[1,i]='1') then
     Expresion:=Expresion+'B'
    else
     Expresion:=Expresion+'-B';
    if (Cells[2,i]='1') then
     Expresion:=Expresion+'C'
    else
     Expresion:=Expresion+'-C';
    Expresion:=Expresion+'+';
   end;
  end;
 end;
 long:=length(Expresion);
 for i:=1 to (long-1) do
  Ecuac:=Ecuac+Expresion[i];
end;
```

PROCEDIMIENTO COMBINACIONES. Controla que el usuario forme correctamente la tabla de verdad que se le propone llenar en el modulo tablas de verdad.

```
procedure TFTablasdeV2.B0Click(Sender: TObject);
var
  Comp,i:integer;
  sw:Boolean;
begin
 if (NTerm<9) then
 begin
  StringGrid1.Cells[NCol,NTerm]:='0';
  if (NCol<3) then
   Acumula:=Acumula+'0';
  NCol:=NCol+1;
  if (NCol=3) then
  begin
   if (NTerm=1)then
   begin
    VTerm[NTerm-1]:=Acumula;
    acumula:='';
   end
   else
   begin
   sw:=False;
   i:=0;
   while ((i<=NTerm) and (sw=False))do
   begin
    Comp:=CompareStr(VTerm[i],Acumula);
    if (Comp=0) then
    begin
     Showmessage('Esa combinación ya la realizaste');
     NCol:=0;
     sw:=True;
     StringGrid1.Cells[0,NTerm]:='';
     StringGrid1.Cells[1,NTerm]:='';
     StringGrid1.Cells[2,NTerm]:='';
     acumula:='';
    end
    else
     i:=i+1;
   end;//while
   if (sw=False) then
   begin
    VTerm[NTerm-1]:=Acumula;
    Acumula:='';
   end;
   end;// NTerm 1
  end;//NCol 3
  if (NCol=4) then
  begin
   NTerm:=NTerm+1;
   NCol:=0;
  end;
  if (NTerm=9) then
```

```
  begin
   Cerrar1.Enabled:=True;
   BRegresar.Visible:=True;
   B0.Visible:=False;
   B1.Visible:=False;
   label1.Visible:=False;
   label2.Visible:=False;
   label3.Visible:=False;
   label4.Visible:=False;
   label5.Visible:=False;
   label6.Visible:=True;
   label7.Visible:=True;
   label8.Visible:=True;
   panel1.Visible:=True;
   EcuacTab(Ecuac);
   label9.Visible:=True;
   label10.Visible:=True;
   label10.Caption:='X= '+Ecuac;
  end;
 end;
end;

procedure TFTablasdeV2.B1Click(Sender: TObject);
var
  sw:Boolean;
  Comp,i:integer;
begin
 if (NTerm<9) then
 begin
  StringGrid1.Cells[NCol,NTerm]:='1';
  if (NCol<3) then
   Acumula:=Acumula+'1';
  NCol:=NCol+1;
  if (NCol=3) then
  begin
  if (NTerm=1)then
  begin
   VTerm[NTerm-1]:=Acumula;
   acumula:='';
  end
  else
   begin
   sw:=False;
   i:=0;
   while ((i<=NTerm) and (sw=False))do
   begin
    Comp:=CompareStr(VTerm[i],Acumula);
    if (Comp=0) then
    begin
     Showmessage('Esa combinación ya la realizaste');
     NCol:=0;
```

```
    sw:=True;
    StringGrid1.Cells[0,NTerm]:='';
    StringGrid1.Cells[1,NTerm]:='';
    StringGrid1.Cells[2,NTerm]:='';
    acumula:='';
   end
   else
    i:=i+1;
  end;
  if (sw=False) then
  begin
   VTerm[NTerm-1]:=Acumula;
   Acumula:='';
  end;
  end;
 end;
 if (NCol=4) then
 begin
  NTerm:=NTerm+1;
  NCol:=0;
 end;
 if (NTerm=9) then
 begin
  Cerrar1.Enabled:=True;
  BRegresar.Visible:=True;
  B0.Visible:=False;
  B1.Visible:=False;
  label1.Visible:=False;
  label2.Visible:=False;
  label3.Visible:=False;
  label4.Visible:=False;
  label5.Visible:=False;
  label6.Visible:=True;
  label7.Visible:=True;
  label8.Visible:=True;
  panel1.Visible:=True;
  EcuacTab(Ecuac);
  label9.Visible:=True;
  label10.Visible:=True;
  label10.Caption:='X= '+Ecuac;
 end;
end;
end;
```

MODULO ECUACIONES BOOLEANAS

PROCEDIMIENTO PROPUESTOS_ECUACIÓN_MAX. Selecciona un ejercicio para evaluar al usuario en el módulo Ecuaciones booleanas de la lección Teoria Básica, de acuerdo a las características de éste.

```
procedure TFFormEcMax.N1Click(Sender: TObject);
var
 sw:boolean;
 i:integer;
begin
 TipoEc:='Ecuaciones Maxterms';
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
   Estudiantes.Edit;
   niv:=Estudiantes.Fields[6].AsInteger;
 end;
 Estudiantes.Close;
 Leccion_TeoriaB.Open;
 Leccion_TeoriaB.SetKey;
 Leccion_TeoriaB.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_TeoriaB.GotoKey;
 Leccion_TeoriaB.Edit;
 EjerSiNo:=Leccion_TeoriaB.Fields[5].AsString;
 Leccion_TeoriaB.Close;
 Ejercicios_Propuestos.Open;
 Ejercicios_Propuestos.DisableControls;
 sw:=False;
 Ejercicios_Propuestos.First;
 while ((not(Ejercicios_Propuestos.EOF)) and (sw=False)) do
 begin
  Ejercicios_Propuestos.Edit;
  if                (Ejercicios_Propuestos.Fields[0].AsString='Ecuaciones
Maxterms')and (Ejercicios_Propuestos.Fields[2].Asinteger=niv) then
   if (EjerSiNo[Ejercicios_Propuestos.Fields[7].AsInteger]='0') then
    sw:=True;
  if sw=false then
   Ejercicios_Propuestos.Next;
 end;
 if (sw) then
 begin
  PropBas5:=PropBas5+1;
```

```
 for i:=1 to 32 do
  VEcuac[i]:='X';
 Ecuacion:=Ejercicios_Propuestos.Fields[1].asstring;
 NumV:=Ejercicios_Propuestos.Fields[6].asInteger;
 NumTer:=Length(Ecuacion);
 for i:=1 to NumTer do
  VEcuac[i]:=Ecuacion[i];
 FEvaluarEc.Show;
 FFormEcMax.Hide;
end
else
 showmessage('No hay más ejercicios sobre este tema en este nivel.');
end;
```

PROCEDIMIENTO RESUELTOS_ECUACIONMAX. Selecciona un ejercicio resuelto para mostrar al usuario del tema Ecuaciones Booleanas en la forma maxterms .

```
procedure TFFormEcMax.Acercade1Click(Sender: TObject);
var
 sw:boolean;
 i:integer;
begin
// NSimpRes:=NSimpRes+1;
 TipoEc:='Ecuaciones Maxterms';
 NumTer:=0;
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
  Estudiantes.Edit;
  niv:=Estudiantes.Fields[6].AsInteger;
 end;
 Estudiantes.Close;
 Leccion_TeoriaB.Open;
 Leccion_TeoriaB.SetKey;
 Leccion_TeoriaB.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_TeoriaB.GotoKey;
 Leccion_TeoriaB.Edit;
 Leccion_TeoriaB.Close;
 Ejercicios_Resueltos.Open;
 Ejercicios_Resueltos.DisableControls;
 sw:=False;
 Ejercicios_Resueltos.First;
```

```
 while ((not(Ejercicios_Resueltos.EOF)) and (sw=False)) do
 begin
   Ejercicios_Resueltos.Edit;
   if                  (Ejercicios_Resueltos.Fields[0].AsString='Ecuaciones
Maxterms')and(Ejercicios_Resueltos.Fields[1].Asinteger=niv) then
   begin
    if (Ejercicios_Resueltos.Fields[2].Asinteger=Num_Res) then
    begin
     sw:=True;
     Num_Res:=Num_Res+1;
    end
    else
     Ejercicios_Resueltos.Next;
   end
   else
    if (sw=false) then
     Ejercicios_Resueltos.Next;
 end;
 if (sw) then
 begin
  Ecuacion:=Ejercicios_Resueltos.Fields[3].asstring;
  NumTer:=Length(Ejercicios_Resueltos.Fields[3].AsString);
  for i:=1 to 32 do
   VEcuac[i]:='X';
  for i:=1 to NumTer do
   VEcuac[i]:=Ecuacion[i];
  FEjerEBRes.Memo1.Text:=Ejercicios_Resueltos.Fields[4].asString;
  FEjerEBRes.Show;
  FFormEcMax.Hide;
 end
 else
  showmessage('No hay más ejercicios sobre este tema en este nivel.');
end;
```

PROCEDIMIENTO SELECCIONA_FILA. Lleva el control de las filas que se han seleccionado de la tabla de verdad propuesta para que el usuario forme la ecuación maxterms correspondiente.

```
procedure TFFormEcMax1.BAceptarClick(Sender: TObject);
var
  i,Numero:integer;
  sw0:boolean;
begin
 if (Num1<5) then
 begin
  Numero:=StrToInt(Edit1.Text);
```

```
  if (Edit1.Text<>'') then
  begin
   if (( Numero<1) or ( Numero>8)) then
   begin
    showmessage('El número de filas se encuentra entre uno (1) y ocho
(8)');
    edit1.Clear;
   end
   else
   begin
    sw0:=False;
    for i:=0 to 3 do
     if (Edit1.Text=VecST[i])then
      sw0:=True;
    if (sw0) then
    begin
     showmessage('Ese número de fila ya lo digitaste. Prueba con otro
por favor');
     Edit1.Clear;
    end
    else
    begin
     NT:=StrToInt(Edit1.Text);
     if (StringGrid1.Cells[3,NT]='1') then
     begin
      showmessage('Para obtener una función maxterms solo se tienen en
cuenta las variables de entrada que hacen cero (0) la función');
      Edit1.Clear;
     end
     else
     begin
      VecST[Num1-1]:=Edit1.Text;
      Num1:=Num1+1;
      NumV:=0;
      Ecuac:='';
      label3.Visible:=True;
      label4.Visible:=True;
      label8.visible:=true;
      label10.Visible:=True;
      label7.Visible:=True;
      BA.visible:=True;
      BA1.visible:=True;
      BB.visible:=True;
      BB1.visible:=True;
      BC.visible:=True;
      BC1.visible:=True;
      BA.enabled:=true;
      BA1.enabled:=true;
      Bb.enabled:=true;
      Bb1.enabled:=true;
      Bc.enabled:=true;
```

```
     Bc1.enabled:=true;
     Label1.Visible:=False;
     Label9.Visible:=False;
     Label5.Visible:=False;
     Label6.Visible:=False;
     label7.caption:=Edit1.Text;
     Edit1.Clear;
     Edit1.Visible:=False;
     BAceptar.visible:=False;
     label2.Visible:=true;
    end;
   end;//else
  end;//else
 end
 else
  showmessage('Por favor digite el número de la fila que hace cero (0)
la función');
end;
end;
```

PROCEDIMIENTO FORMAR_A. Cuando el usuario está formando la ecuación en forma maxterms resultante de la tabla propuesta en en el modulo ecuaciones booleanas, al seleccionar A este procedimiento revisa que sea la selección correcta.

```
procedure TFFormEcMax1.BAClick(Sender: TObject);
begin
 with StringGrid1 do
 begin
  if (Cells[0,NT]='0') then
  begin
   if (Ecuac='') then
    Ecuac:='('+BA.Caption
   else
    Ecuac:=Ecuac+'+'+BA.Caption;
   label2.Caption:=Func+Ecuac;
   NumV:=NumV+1;
   if ((Num1=5)and(NumV=3)) then
   begin
    Label4.Visible:=False;
    Label11.Visible:=True;
    Cerrar1.Enabled:=True;
    BRegresar.Visible:=True;
```

```
   end;
   if (NumV=3) then
   begin
    Ecuac:=Ecuac+')';
    Func:=Func+Ecuac;
   end;
   BA.Enabled:=False;
   BA1.Enabled:=False;
   if ((Num1<5) and (NumV=3)) then
   begin
    Label5.Visible:=True;
    Label6.Visible:=True;
    Edit1.Visible:=True;
    BAceptar.Visible:=True;
    BAceptar.Enabled:=True;
   end;
   label2.Caption:=func;
  end
  else
   Showmessage('Debes digitar -A');
 end;
end;
```

PROCEDIMIENTO FORMAR_-A. Cuando el usuario está formando la ecuación en forma maxterms resultante de la tabla propuesta en en el modulo ecuaciones booleanas, al seleccionar -A este procedimiento revisa que sea la selección correcta.

```
procedure TFFormEcMax1.BA1Click(Sender: TObject);
begin
  with StringGrid1 do
 begin
  if (Cells[0,NT]='1') then
  begin
   if (Ecuac='') then
    Ecuac:='('+BA1.Caption
   else
    Ecuac:=Ecuac+'+'+BA1.Caption;
   label2.Caption:=Func+Ecuac;
   NumV:=NumV+1;
   if ((Num1=5)and(NumV=3)) then
   begin
    Label4.Visible:=False;
```

```
    Label11.Visible:=True;
    Cerrar1.Enabled:=True;
    BRegresar.Visible:=True;
   end;
   if (NumV=3) then        ///# de terminos
   begin
    Ecuac:=Ecuac+')';
    Func:=Func+Ecuac;
   end;
   BA.Enabled:=False;
   BA1.Enabled:=False;
   if ((Num1<5) and (NumV=3)) then
   begin
    Label5.Visible:=True;
    Label6.Visible:=True;
    Edit1.Visible:=True;
    BAceptar.Visible:=True;
    BAceptar.Enabled:=True;
   end;
   label2.Caption:=func;
  end
  else
   Showmessage('Debes digitar A');
 end;
end;
```

PROCEDIMIENTO FORMAR_B. Cuando el usuario está formando la ecuación en forma maxterms resultante de la tabla propuesta en en el modulo ecuaciones booleanas, al seleccionar B este procedimiento revisa que sea la selección correcta.

```
procedure TFFormEcMax1.BBClick(Sender: TObject);
begin
 with StringGrid1 do
 begin
  if (Cells[1,NT]='0') then
  begin
   if (Ecuac='') then
    Ecuac:='('+BB.Caption
   else
    Ecuac:=Ecuac+'+'+BB.Caption;
   label2.Caption:=Func+Ecuac;
   NumV:=NumV+1;
```

```
   if ((Num1=5)and(NumV=3)) then
   begin
    Label4.Visible:=False;
    Label11.Visible:=True;
    Cerrar1.Enabled:=True;
    BRegresar.Visible:=True;
   end;
   if (NumV=3) then       ///# de terminos
   begin
    Ecuac:=Ecuac+')';
    Func:=Func+Ecuac;
   end;
   BB.Enabled:=False;
   BB1.Enabled:=False;
   if ((Num1<5) and (NumV=3)) then
   begin
    Label5.Visible:=True;
    Label6.Visible:=True;
    Edit1.Visible:=True;
    BAceptar.Visible:=True;
    BAceptar.Enabled:=True;
   end;
   label2.Caption:=Func;
  end
  else
   Showmessage('Debes digitar -B');
 end;
end;
```

PROCEDIMIENTO FORMAR_-B. Cuando el usuario está formando la ecuación en forma maxterms resultante de la tabla propuesta en en el modulo ecuaciones booleanas, al seleccionar -B este procedimiento revisa que sea la selección correcta.

```
procedure TFFormEcMax1.BB1Click(Sender: TObject);
begin
 with StringGrid1 do
 begin
  if (Cells[1,NT]='1') then
  begin
   if (Ecuac='') then
    Ecuac:='('+BB1.Caption
   else
```

```
   Ecuac:=Ecuac+'+'+BB1.Caption;
  label2.Caption:=Func+Ecuac;
  NumV:=NumV+1;
  if ((Num1=5)and(NumV=3)) then
  begin
   Label4.Visible:=False;
   Label11.Visible:=True;
   Cerrar1.Enabled:=True;
   BRegresar.Visible:=True;
  end;
  if (NumV=3) then       ///# de terminos
  begin
   Ecuac:=Ecuac+')';
   Func:=Func+Ecuac;
  end;
  BB.Enabled:=False;
  BB1.Enabled:=False;
  if ((Num1<5) and (NumV=3)) then
  begin
   Label5.Visible:=True;
   Label6.Visible:=True;
   Edit1.Visible:=True;
   BAceptar.Visible:=True;
   BAceptar.Enabled:=True;
  end;
  label2.Caption:=Func;
 end
 else
  Showmessage('Debes digitar B');
end;
end;
```

PROCEDIMIENTO FORMAR_C. Cuando el usuario está formando la ecuación en forma maxterms resultante de la tabla propuesta en en el modulo ecuaciones booleanas, al seleccionar C este procedimiento revisa que sea la selección correcta.

```
procedure TFFormEcMax1.BCClick(Sender: TObject);
begin
 with StringGrid1 do
 begin
  if (Cells[2,NT]='0') then
  begin
```

```
  if (Ecuac='') then
   Ecuac:='('+BC.Caption
  else
   Ecuac:=Ecuac+'+'+BC.Caption;
  label2.Caption:=Func+Ecuac;
  NumV:=NumV+1;
  if ((Num1=5)and(NumV=3)) then
  begin
   Label4.Visible:=False;
   Label11.Visible:=True;
   Cerrar1.Enabled:=True;
   BRegresar.Visible:=True;
  end;
  if (NumV=3) then       ///# de terminos
  begin
   Ecuac:=Ecuac+')';
   Func:=Func+Ecuac;
  end;
  BC.Enabled:=False;
  BC1.Enabled:=False;
  if ((Num1<5) and (NumV=3)) then
  begin
   Label5.Visible:=True;
   Label6.Visible:=True;
   Edit1.Visible:=True;
   BAceptar.Visible:=True;
   BAceptar.Enabled:=True;
  end;
  label2.Caption:=Func;
 end
 else
  Showmessage('Debes digitar -C');
end;
end;
```

PROCEDIMIENTO FORMAR_-C. Cuando el usuario está formando la ecuación en forma maxterms resultante de la tabla propuesta en en el modulo ecuaciones booleanas, al seleccionar -C este procedimiento revisa que sea la selección correcta.

```
procedure TFFormEcMax1.BC1Click(Sender: TObject);
begin
 with StringGrid1 do
```

```
 begin
  if (Cells[2,NT]='1') then
  begin
   if (Ecuac='') then
    Ecuac:='('+BC1.Caption
   else
    Ecuac:=Ecuac+'+'+BC1.Caption;
   label2.Caption:=Func+Ecuac;
   NumV:=NumV+1;
   if ((Num1=5)and(NumV=3)) then
   begin
    Label4.Visible:=False;
    Label11.Visible:=True;
    Cerrar1.Enabled:=True;
    BRegresar.Visible:=True;
   end;
   if (NumV=3) then
   begin
    Ecuac:=Ecuac+')';
    Func:=Func+Ecuac;
   end;
   BC.Enabled:=False;
   BC1.Enabled:=False;
   if ((Num1<5) and (NumV=3)) then
   begin
    Label5.Visible:=True;
    Label6.Visible:=True;
    Edit1.Visible:=True;
    BAceptar.Visible:=True;
    BAceptar.Enabled:=True;
   end;
   label2.Caption:=Func;
  end
  else
   Showmessage('Debes digitar C');
 end;
end;
```

PROCEDIMIENTO SELECCIONA_FILA. Lleva el control de las filas que se han seleccionado de la tabla de verdad propuesta para que el usuario forme la ecuación minterms correspondiente.

```
procedure TFFormEcMin1.BAceptarClick(Sender: TObject);
var
```

```
  i,Numero:integer;
  sw0:boolean;
begin
 if (Num1<5) then
 begin
  if (Edit1.Text<>'') then
  begin
   Numero:=StrToInt(Edit1.Text);
   if ((Numero<1) or (Numero>8)) then
   begin
    showmessage('El número de filas se encuentra entre uno (1) y ocho
(8)');
    edit1.Clear;
   end
   else
   begin
    sw0:=False;
    for i:=0 to 3 do
     if (Edit1.Text=VecST[i])then
      sw0:=True;
    if (sw0) then
    begin
     showmessage('Ese número de fila ya lo digitaste. Prueba con otro
por favor');
     Edit1.Clear;
    end
    else
    begin
     NT:=StrToInt(Edit1.Text);
     if (StringGrid1.Cells[3,NT]='0') then
     begin
      showmessage('Para obtener una función minterms solo se tienen en
cuenta las variables de entrada que hacen uno (1) la función');
      Edit1.Clear;
     end
     else
     begin
      VecST[Num1-1]:=Edit1.Text;
      Num1:=Num1+1;
      NumV:=0;
      Ecuac:='';
      label3.Visible:=True;
      label4.Visible:=True;
      label8.visible:=true;
      label7.Visible:=True;
      label10.Visible:=True;
      BA.visible:=True;
      BA1.visible:=True;
      BB.visible:=True;
      BB1.visible:=True;
      BC.visible:=True;
```

```
      BC1.visible:=True;
      BA.enabled:=true;
      BA1.enabled:=true;
      Bb.enabled:=true;
      Bb1.enabled:=true;
      Bc.enabled:=true;
      Bc1.enabled:=true;
      Label5.Visible:=False;
      Label6.Visible:=False;
      label7.caption:=Edit1.Text;
      Edit1.Clear;
      Edit1.Visible:=False;
      BAceptar.visible:=False;
      label2.Visible:=true;
     end;
    end;//else
   end;//else
  end
  else
   showmessage('Por favor digite el número de la fila que hace cero (0)
la función');
 end;
end;
```

PROCEDIMIENTO FORMAR_A. Cuando el usuario está formando la ecuación en forma minterms resultante de la tabla propuesta en en el modulo ecuaciones booleanas, al seleccionar A este procedimiento revisa que sea la selección correcta.

```
procedure TFFormEcMin1.BAClick(Sender: TObject);
begin
 with StringGrid1 do
 begin
  if (Cells[0,NT]='1') then
  begin
   ecuac:=ecuac+BA.Caption;
   label2.caption:=func+ecuac;
   NumV:=NumV+1;
   if ((Num1=5)and(NumV=3)) then
   begin
    Label4.Visible:=False;
    Label11.Visible:=True;
    Cerrar1.Enabled:=True;
```

```
   BRegresar.Visible:=True;
  end;
  if (NumV=3) then
  begin
   if (Num1<=4) then
   begin
    Ecuac:=Ecuac+'+';
    Func:=Func+Ecuac;
    label3.visible:=false;
    label10.visible:=false;
    label7.visible:=false;
    BA.VISIBLE:=false;
    Bb.VISIBLE:=false;
    Bc.VISIBLE:=false;
    Ba1.VISIBLE:=false;
    Bb1.VISIBLE:=false;
    Bc1.VISIBLE:=false;
   end
   else
    Func:=Func+Ecuac;
  label2.caption:=func;
  end;
  BA.Enabled:=False;
  BA1.Enabled:=False;
  if ((Num1<5) and (NumV=3)) then
  begin
   Label5.Visible:=True;
   Label6.Visible:=True;
   Edit1.Visible:=True;
   BAceptar.visible:=True;
  end;
 end
 else
  Showmessage('Debes digitar -A');
end;
end;
```

PROCEDIMIENTO FORMAR_-A. Cuando el usuario está formando la ecuación en forma minterms resultante de la tabla propuesta en en el modulo ecuaciones booleanas, al seleccionar -A este procedimiento revisa que sea la selección correcta.

```
procedure TFFormEcMin1.BA1Click(Sender: TObject);
```

```
begin
 with StringGrid1 do
 begin
  if (Cells[0,NT]='0') then
  begin
   ecuac:=ecuac+BA1.Caption;
   label2.caption:=func+ecuac;
   NumV:=NumV+1;
   if ((Num1=5)and(NumV=3)) then
   begin
    Label4.Visible:=False;
    Label11.Visible:=True;
    Cerrar1.Enabled:=True;
    BRegresar.Visible:=True;
   end;
   if (NumV=3) then
   begin
    if (Num1<=4) then
    begin
     Ecuac:=Ecuac+'+';
     Func:=Func+Ecuac;
     label3.visible:=false;
     label10.visible:=false;
     label7.visible:=false;
     BA.VISIBLE:=false;
     Bb.VISIBLE:=false;
     Bc.VISIBLE:=false;
     Ba1.VISIBLE:=false;
     Bb1.VISIBLE:=false;
     Bc1.VISIBLE:=false;
    end
    else
     Func:=Func+Ecuac;
   label2.caption:=func;
   end;
   BA.Enabled:=False;
   BA1.Enabled:=False;
   if ((Num1<5) and (NumV=3)) then
   begin
    Label5.Visible:=True;
    Label6.Visible:=True;
    Edit1.Visible:=True;
    BAceptar.visible:=True;
   end;
  end
  else
   Showmessage('Debes digitar A');
 end;
end;
```

PROCEDIMIENTO FORMAR_B. Cuando el usuario está formando la ecuación en forma minterms resultante de la tabla propuesta en en el modulo ecuaciones booleanas, al seleccionar B este procedimiento revisa que sea la selección correcta.

```
procedure TFFormEcMin1.BBClick(Sender: TObject);
begin
 with StringGrid1 do
 begin
  if (Cells[1,NT]='1') then
  begin
   ecuac:=ecuac+Bb.Caption;
   label2.caption:=func+ecuac;
   NumV:=NumV+1;
   if ((Num1=5)and(NumV=3)) then
   begin
    Label4.Visible:=False;
    Label11.Visible:=True;
    Cerrar1.Enabled:=True;
    BRegresar.Visible:=True;
   end;
   if (NumV=3) then
   begin
    if (Num1<=4) then
    begin
     Ecuac:=Ecuac+'+';
     Func:=Func+Ecuac;
     label3.visible:=false;
     label10.visible:=false;
     label7.visible:=false;
     BA.VISIBLE:=false;
     Bb.VISIBLE:=false;
     Bc.VISIBLE:=false;
     Ba1.VISIBLE:=false;
     Bb1.VISIBLE:=false;
     Bc1.VISIBLE:=false;
    end
    else
     Func:=Func+Ecuac;
   label2.caption:=func;
   end;
   Bb.Enabled:=False;
   Bb1.Enabled:=False;
   if ((Num1<5) and (NumV=3)) then
   begin
```

```
    Label5.Visible:=True;
    Label6.Visible:=True;
    Edit1.Visible:=True;
    BAceptar.visible:=True;
   end;
  end
  else
   Showmessage('Debes digitar -B');
 end;
end;
```

PROCEDIMIENTO FORMAR_-B. Cuando el usuario está formando la ecuación en forma minterms resultante de la tabla propuesta en en el modulo ecuaciones booleanas, al seleccionar -B este procedimiento revisa que sea la selección correcta.

```
procedure TFFormEcMin1.BB1Click(Sender: TObject);
begin
 with StringGrid1 do
 begin
  if (Cells[1,NT]='0') then
  begin
   ecuac:=ecuac+Bb1.Caption;
   label2.caption:=func+ecuac;
   NumV:=NumV+1;
   if ((Num1=5)and(NumV=3)) then
   begin
    Label4.Visible:=False;
    Label11.Visible:=True;
    Cerrar1.Enabled:=True;
    BRegresar.Visible:=True;
   end;
   if (NumV=3) then
   begin
    if (Num1<=4) then
    begin
     Ecuac:=Ecuac+'+';
     Func:=Func+Ecuac;
     label3.visible:=false;
     label10.visible:=false;
     label7.visible:=false;
     BA.VISIBLE:=false;
     Bb.VISIBLE:=false;
     Bc.VISIBLE:=false;
```

```
    Ba1.VISIBLE:=false;
    Bb1.VISIBLE:=false;
    Bc1.VISIBLE:=false;
   end
   else
    Func:=Func+Ecuac;
  label2.caption:=func;
  end;
  Bb.Enabled:=False;
  Bb1.Enabled:=False;
  if ((Num1<5) and (NumV=3)) then
  begin
   Label5.Visible:=True;
   Label6.Visible:=True;
   Edit1.Visible:=True;
   BAceptar.visible:=True;
  end;
 end
 else
  Showmessage('Debes digitar B');
end;
end;
```

PROCEDIMIENTO FORMAR_C. Cuando el usuario está formando la ecuación en forma minterms resultante de la tabla propuesta en en el modulo ecuaciones booleanas, al seleccionar C este procedimiento revisa que sea la selección correcta.

```
procedure TFFormEcMin1.BCClick(Sender: TObject);
begin
 with StringGrid1 do
 begin
  if (Cells[2,NT]='1') then
  begin
   ecuac:=ecuac+Bc.Caption;
   label2.caption:=func+ecuac;
   NumV:=NumV+1;
   if ((Num1=5)and(NumV=3)) then
   begin
    Label4.Visible:=False;
    Label11.Visible:=True;
    Cerrar1.Enabled:=True;
    BRegresar.Visible:=True;
```

```
  end;
  if (NumV=3) then
  begin
   if (Num1<=4) then
   begin
    Ecuac:=Ecuac+'+';
    Func:=Func+Ecuac;
    label3.visible:=false;
    label10.visible:=false;
    label7.visible:=false;
    BA.VISIBLE:=false;
    Bb.VISIBLE:=false;
    Bc.VISIBLE:=false;
    Ba1.VISIBLE:=false;
    Bb1.VISIBLE:=false;
    Bc1.VISIBLE:=false;
   end
   else
    Func:=Func+Ecuac;
  label2.caption:=func;
  end;
  Bc.Enabled:=False;
  Bc1.Enabled:=False;
  if ((Num1<5) and (NumV=3)) then
  begin
   Label5.Visible:=True;
   Label6.Visible:=True;
   Edit1.Visible:=True;
   BAceptar.visible:=True;
  end;
 end
 else
  Showmessage('Debes digitar -C');
end;
end;
```

PROCEDIMIENTO FORMAR_-C. Cuando el usuario está formando la ecuación en forma minterms resultante de la tabla propuesta en en el modulo ecuaciones booleanas, al seleccionar -C este procedimiento revisa que sea la selección correcta.

```
procedure TFFormEcMin1.BC1Click(Sender: TObject);
begin
```

```
 with StringGrid1 do
 begin
  if (Cells[2,NT]='0') then
  begin
   ecuac:=ecuac+Bc1.Caption;
   label2.caption:=func+ecuac;
   NumV:=NumV+1;
   if ((Num1=5)and(NumV=3)) then
   begin
    Label4.Visible:=False;
    Label11.Visible:=True;
    Cerrar1.Enabled:=True;
    BRegresar.Visible:=True;
   end;
   if (NumV=3) then
   begin
    if (Num1<=4) then
    begin
     Ecuac:=Ecuac+'+';
     Func:=Func+Ecuac;
     label3.visible:=false;
     label10.visible:=false;
     label7.visible:=false;
     BA.VISIBLE:=false;
     Bb.VISIBLE:=false;
     Bc.VISIBLE:=false;
     Ba1.VISIBLE:=false;
     Bb1.VISIBLE:=false;
     Bc1.VISIBLE:=false;
    end
    else
     Func:=Func+Ecuac;
   label2.caption:=func;
   end;
   Bc.Enabled:=False;
   Bc1.Enabled:=False;
   if ((Num1<5) and (NumV=3)) then
   begin
    Label5.Visible:=True;
    Label6.Visible:=True;
    Edit1.Visible:=True;
    BAceptar.visible:=True;
   end;
  end
  else
   Showmessage('Debes digitar C');
 end;
end;
```

PROCEDIMIENTO PROPUESTOS_ECUACIÓN_MIN. Selecciona un ejercicio para evaluar al usuario en el módulo Ecuaciones booleanas de la lección Teoria Básica, de acuerdo a las características de éste.

```
procedure TFFormEcMin.N1Click(Sender: TObject);
var
 sw:boolean;
 i:integer;
begin
 TipoEc:='Ecuaciones Minterms';
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
   Estudiantes.Edit;
   niv:=Estudiantes.Fields[6].AsInteger;
 end;
 Estudiantes.Close;
 Leccion_TeoriaB.Open;
 Leccion_TeoriaB.SetKey;
 Leccion_TeoriaB.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_TeoriaB.GotoKey;
 Leccion_TeoriaB.Edit;
 EjerSiNo:=Leccion_TeoriaB.Fields[5].AsString;
 Leccion_TeoriaB.Close;
 Ejercicios_Propuestos.Open;
 Ejercicios_Propuestos.DisableControls;
 sw:=False;
 Ejercicios_Propuestos.First;
 while ((not(Ejercicios_Propuestos.EOF)) and (sw=False)) do
 begin
  Ejercicios_Propuestos.Edit;
  if (Ejercicios_Propuestos.Fields[0].AsString= 'Ecuaciones Minterms')
and (Ejercicios_Propuestos.Fields[2].Asinteger=niv) then
   if (EjerSiNo[Ejercicios_Propuestos.Fields[7].AsInteger]='0') then
    sw:=True;
  if sw=false then
   Ejercicios_Propuestos.Next;
 end;
 if (sw) then
 begin
  PropBas4:=PropBas4+1;
  for i:=1 to 32 do
```

```
  VEcuac[i]:='X';
 Ecuacion:=Ejercicios_Propuestos.Fields[1].asstring;
 NumV:=Ejercicios_Propuestos.Fields[6].asInteger;
 NumTer:=Length(Ecuacion);
 for i:=1 to NumTer do
  VEcuac[i]:=Ecuacion[i];
 FEvaluarEc.Show;
 FFormEcMin.Hide;
end
else
 showmessage('No hay más ejercicios sobre este tema en este nivel.');
end;
```

PROCEDIMIENTO RESUELTOS_ECUACIONMIN. Selecciona un ejercicio resuelto para mostrar al usuario del tema Ecuaciones Booleanas en la forma minterms .

```
procedure TFFormEcMin.Acercade1Click(Sender: TObject);
var
 sw:boolean;
 i:integer;
begin
// NSimpRes:=NSimpRes+1;
 TipoEc:='Ecuaciones Minterms';
 NumTer:=0;
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
  Estudiantes.Edit;
  niv:=Estudiantes.Fields[6].AsInteger;
 end;
 Estudiantes.Close;
 Leccion_TeoriaB.Open;
 Leccion_TeoriaB.SetKey;
 Leccion_TeoriaB.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_TeoriaB.GotoKey;
 Leccion_TeoriaB.Edit;
 Leccion_TeoriaB.Close;
 Ejercicios_Resueltos.Open;
 Ejercicios_Resueltos.DisableControls;
 sw:=False;
 Ejercicios_Resueltos.First;
```

```
 while ((not(Ejercicios_Resueltos.EOF)) and (sw=False)) do
 begin
   Ejercicios_Resueltos.Edit;
   if                  (Ejercicios_Resueltos.Fields[0].AsString='Ecuaciones
Minterms')and(Ejercicios_Resueltos.Fields[1].Asinteger=niv) then
   begin
    if (Ejercicios_Resueltos.Fields[2].Asinteger=Num_Res) then
    begin
     sw:=True;
     NumTer:=Length(Ejercicios_Resueltos.Fields[3].AsString);
     Num_Res:=Num_Res+1;
    end
    else
     Ejercicios_Resueltos.Next;
   end
   else
    if (sw=false) then
     Ejercicios_Resueltos.Next;
 end;
 if (sw) then
 begin
  Ecuacion:=Ejercicios_Resueltos.Fields[3].asstring;
  NumTer:=Length(Ejercicios_Resueltos.Fields[3].AsString);
  for i:=1 to 32 do
   VEcuac[i]:='X';
  for i:=1 to NumTer do
   VEcuac[i]:=Ecuacion[i];
  FEjerEBRes.Memo1.Text:=Ejercicios_Resueltos.Fields[4].asString;
  FEjerEBRes.Show;
  FFormEcMin.Hide;
 end
 else
  showmessage('No hay más ejercicios sobre este tema en este nivel.');
end;
```

LECCIÓN SIMPLIFICACIÓN McCLUSKEY.

PROCEDIMIENTO INICIAR_McCLUSKEY. Cuando un usuario visita la lección Método de simplificación de Quine McCluskey, actualiza los datos de éste en las tablas Estudiantes e Lección_McCluskey.

```
procedure TForm7.BMcCluskeyClick(Sender: TObject);
begin
 form7.hide;
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
  Estudiantes.Edit;
  Estudiantes.Fields[4].AsString:='Simplificación de McCluskey';
  Estudiantes.Post;
 end;
 Estudiantes.EnableControls;
 Estudiantes.Close;
 Leccion_McCluskey.Open;
 Leccion_McCluskey.SetKey;
 Leccion_McCluskey.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_McCluskey.GotoKey;
 if (Leccion_McCluskey.GotoKey=True) then
 begin
  Leccion_McCluskey.Edit;

Leccion_McCluskey.Fields[1].AsInteger:=Leccion_McCluskey.Fields[1].A
sInteger+1;
  Leccion_McCluskey.Post;
  Leccion_McCluskey.EnableControls;
  Leccion_McCluskey.Close;
 end
 else
  begin
   Leccion_McCluskey.DisableControls;
   Leccion_McCluskey.Last;
   Leccion_McCluskey.Insert;
   Leccion_McCluskey.Edit;
   Leccion_McCluskey.Fields[0].AsString:=Codigo1;
   Leccion_McCluskey.Fields[1].AsInteger:=1;
   Leccion_McCluskey.Fields[2].AsFloat:=0;
   Leccion_McCluskey.Fields[3].AsInteger:=0;

Leccion_McCluskey.Fields[4].AsString:='000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000';

Leccion_McCluskey.Fields[5].AsString:='000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000';
```

```
Leccion_McCluskey.Fields[6].AsString:='0000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000';
   Leccion_McCluskey.post;
   Leccion_McCluskey.EnableControls;
   Leccion_McCluskey.Close;
  end;
 form9.show;
end;
```

PROCEDIMIENTO EVALUAR_MINTERMS. Asigna un ejercicio para evaluar al estudiante en el modulo Expresiones Minterms de la lección Simplificación de McCluskey, de acuerdo al nivel en que se encuentre él.

```
procedure TForm10.Propuestos1Click(Sender: TObject);
var
  niv:integer;
  sw:boolean;
begin
 PropExp:=PropExp+1;
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 Estudiantes.Edit;
 niv:=Estudiantes.Fields[6].AsInteger;
 Estudiantes.Close;
 Leccion_McCluskey.Open;
 Leccion_McCluskey.SetKey;
 Leccion_McCluskey.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_McCluskey.GotoKey;
 Leccion_McCluskey.Edit;
 EjerSiNo:=Leccion_McCluskey.Fields[4].AsString;
 Leccion_McCluskey.Close;
 Ejercicios_Propuestos.Open;
 Ejercicios_Propuestos.DisableControls;
 sw:=False;
 Ejercicios_Propuestos.First;
 while ((not(Ejercicios_Propuestos.EOF)) and (sw=False)) do
```

```
 begin
  Ejercicios_Propuestos.Edit;
  if (Ejercicios_Propuestos.Fields[0].AsString='Expresiones
Minterms')and (Ejercicios_Propuestos.Fields[2].Asinteger=niv) then
  if (EjerSiNo[Ejercicios_Propuestos.Fields[7].AsInteger]='0') then
    sw:=True;
  if sw=false then
    Ejercicios_Propuestos.Next;
 end;
  if (sw) then
  begin

FEvaluarMinterms.MEnunciado.Text:=Ejercicios_Propuestos.Fields[1].as
string;
   form10.hide;
   FEvaluarMinterms.Show;
  end
  else
   showmessage('No existen más ejercicios de este nivel');
end;
```

PROCEDIMIENTO BINARIO_DECIMAL. Toma una cadena de unos y ceros y la convierte en una que sea el número decimal correspondiente.

```
procedure binadec(var binario:string);
var
acumbin,bin,k,j: integer;
begin
 acumbin:=0;
 for k:=1 to length(binario) do
  if (binario[k]='1') then
  begin
   bin:=1;
   for j:=1 to length(binario)-k do
    bin:=bin*2;
    acumbin:=acumbin+bin;
   end;
 binario:=inttostr(acumbin);
end;
```

PROCEDIMIENTO ORDENAR_TERMINOS. Permite ordenar alfabéticamente de menor a mayor, un termino de una ecuación booleana.

```
procedure ordenterm(var guarda:string);
var
i,sw,sw1,sw2,j,l,long: integer;
aux: char;
begin
sw:=0;

while(sw=0)do{mq 1}
begin
  i:=1;
  j:=1;
  sw:=1;
  long:=length(guarda);
  while(j<=long-1)do {mq 2}
  begin
    sw1:=0;
    sw2:=0;
    if(guarda[i]='-')then
    begin
      i:=i+1;
      sw1:=1;
    end;
    if(guarda[i+1]='-')then
    begin
      j:=i+2;
      sw2:=1;
    end
    else
      j:=i+1;
    l:=comparestr(guarda[i],guarda[j]);
    if (l>0)then
    begin
      sw:=0;

      if ((sw1=0) and (sw2=0))or((sw1=1) and (sw2=1)) then
      begin
        aux:=guarda[i];
        guarda[i]:=guarda[j];
        guarda[j]:=aux;
      end
      else
      if (sw1=1) and (sw2=0) then
```

```
     begin
       guarda[i-1]:=guarda[j];
       aux:=guarda[i];
       guarda[i]:='-';
       guarda[j]:=aux;
     end
     else
     if (sw1=0) and (sw2=1) then
     begin
       aux:=guarda[i];
       guarda[i]:='-';
       guarda[j-1]:=guarda[j];
       guarda[j]:=aux;
     end;
   end;
   if ((sw=0)and(sw1=0)and(sw2=1)) or ((sw=0)and(sw1=0)and(sw2=0))
or((sw=1)and(sw1=0)and(sw2=0)) or ((sw=1)and(sw1=1)and(sw2=0))
then
     i:=j
   else
   i:=j-1;
  end;
end;
end;
```

PROCEDIMIENTO LLENAR_AGRUPAMIENTO. Controla que el usuario llene correctamente la columna 2 de la tabla de agrupamiento base del método de simplificación Quine McCluskey.

```
procedure TForm12.AceptarClick(Sender: TObject);
var
   i,k,longitud,comp:integer;
   sw:boolean;
begin
 if (cuentavar<>cuentavar1) then
  showmessage('Faltan variables en el término')
 else
 begin
   if (stringGrid1.colcount=1)then
   begin
   termagrup:='';
   termagrup:=stringGrid1.cells[0,cuentaterm];
   ordenterm(termagrup);
   k:=0;
```

```
  sw:=False;
  while (k<CuentaTerm) and (sw=False) do
  begin
   comp:=CompareStr(stringGrid1.cells[0,k],Termagrup);
   if (comp=0) then
    sw:=True;
   k:=k+1;
  end;
  if (sw) then
  begin
   showmessage('Término ya existe');
   stringGrid1.Cells[0,CuentaTerm]:='';
   cuentavar1:=0;
   if (form13.BD.Enabled=False)then
   begin
    BD.Enabled:=True;
    BD1.Enabled:=True;
   end;
   if ((form13.BB1.Enabled=False) or (form13.BB2.Enabled=False) or
(form13.BB3.Enabled=False))then
   begin
    BB.Enabled:=True;
    BB1.Enabled:=True;
   end;
   if ((form13.BC1.Enabled=False) or (form13.BC2.Enabled=False) or
(form13.BC3.Enabled=False))then
   begin
    BC.Enabled:=True;
    BC1.Enabled:=True;
   end;
   if ((form13.BE1.Enabled=False) or (form13.BE2.Enabled=False) or
(form13.BE3.Enabled=False))then
   begin
    BE.Enabled:=True;
    BE1.Enabled:=True;
   end;
   if ((form13.BA1.Enabled=False) or (form13.BA2.Enabled=False)) then
   begin
    BA.Enabled:=True;
    BA1.Enabled:=True;
   end;
   if ((form13.BF1.Enabled=False) or (form13.BF2.Enabled=False)) then
   begin
    BF.Enabled:=True;
    BF1.Enabled:=True;
   end;
   end
  else
  begin
  stringGrid1.Cells[0,cuentaterm]:=termagrup;
  cuentavar1:=0;
```

```
  CuentaTerm:=CuentaTerm+1;
  if (CuentaTerm=NumTerminos) then
   begin
   label3.visible:=False;
   label4.visible:=False;
   edit1.visible:=False;
   Aceptar4.visible:=False;
   memo2.visible:=True;
   stringGrid1.colcount:=2;
   cuentavar1:=0;
   cuentaterm:=0;
  end
  else
  begin
  if (form13.BD.Enabled=False)then
  begin
   BD.Enabled:=True;
   BD1.Enabled:=True;
  end;

  if ((form13.BB1.Enabled=False) or (form13.BB2.Enabled=False) or
(form13.BB3.Enabled=False))then
  begin
   BB.Enabled:=True;
   BB1.Enabled:=True;
  end;

  if ((form13.BC1.Enabled=False) or (form13.BC2.Enabled=False) or
(form13.BC3.Enabled=False))then
  begin
   BC.Enabled:=True;
   BC1.Enabled:=True;
  end;

  if ((form13.BE1.Enabled=False) or (form13.BE2.Enabled=False) or
(form13.BE3.Enabled=False))then
  begin
   BE.Enabled:=True;
   BE1.Enabled:=True;
  end;

  if ((form13.BA1.Enabled=False) or (form13.BA2.Enabled=False))   then
  begin
   BA.Enabled:=True;
   BA1.Enabled:=True;
  end;

  if ((form13.BF1.Enabled=False) or (form13.BF2.Enabled=False)) then
  begin
   BF.Enabled:=True;
   BF1.Enabled:=True;
```

```
 end;
end;
end;
end;

if (stringGrid1.colcount=2)then
begin   //5
  label3.visible:=false;
  label4.visible:=false;
  label1.visible:=false;
  label2.visible:=false;
  strinGgrid1.width:=192;
  B0.enabled:=true;
  B1.enabled:=true;
  BA.visible:=false;
  BA1.visible:=false;
  BB.visible:=false;
  BB1.visible:=false;
  BC.visible:=false;
  BC1.visible:=false;
  BD.visible:=false;
  BD1.visible:=false;
  BE.visible:=false;
  BE1.visible:=false;
  BF.visible:=false;
  BF1.visible:=false;
  B0.visible:=true;
  B1.visible:=true;
  term2:=' ';
  if (cuentaterm>0)then
  begin//6
    term0:=stringGrid1.cells[0,cuentaterm-1];
    i:=1;
    longitud:=length(term0);
    term1:='';
    While(i<=longitud)do
    begin
    if (term0[i]='-') then
    begin
     term1:=term1+'0';
     i:=i+2;
    end
    else
     if (term0[i]='A')or
(term0[i]='B')or(term0[i]='C')or(term0[i]='D')or(term0[i]='E')OR
(term0[i]='F')then
      begin
       term1:=term1+'1';
       i:=i+1;
      end
      ELSE
```

```
    I:=I+1;
  end;
  term2:=stringGrid1.cells[1,cuentaterm-1];
if (term1<>term2)then
begin
 panel1.visible:=true;
 label14.caption:=term1;
 label19.caption:=term2;
 stringGrid1.cells[1,cuentaterm-1]:=term1;
end;
end;
if (CuentaTerm=NumTerminos) then
begin
 stringGrid1.colcount:=3;
 stringGrid1.Width:=288;
 cuentavar1:=0;
 cuentaterm:=0;
 memo2.visible:=false;
 B0.Enabled:=false;
 B1.Enabled:=false;
 memo2.visible:=false;
 memo1.visible:=true;
 memo3.visible:=true;
 edit2.visible:=true;
 edit2.Focused;
 edit2.enabled:=true;
 aceptar2.visible:=true;
 aceptar.visible:=false;
 b1.visible:=false;
 b0.visible:=false;
 ba.visible:=false;
 bb.visible:=false;
 bc.visible:=false;
 bd.visible:=false;
 be.visible:=false;
 bf.visible:=false;
 ba1.visible:=false;
 bb1.visible:=false;
 bc1.visible:=false;
 bd1.visible:=false;
 be1.visible:=false;
 bf1.visible:=false;
 cuentaterm:=0;
 panel1.visible:=false;
 ejemplo.visible:=true;
end
else
begin
cuentavar1:=0;
cuentaterm:=cuentaterm+1;
end;
```

```
   end;
  end;
end;
```

PROCEDIMIENTO LLENAR_AGRUPAMIENTO2. Controla que el usuario llene correctamente la columna 3 de la tabla de agrupamiento base del método de simplificación Quine McCluskey.

```
procedure TForm14.SpeedButton1Click(Sender: TObject);
var
num,bina:string;
sw:boolean;
long,i,cuenta,j,k:integer;
begin
 if (cuentaterm<=NumTerminos-1)then
 begin
  num:=edit1.text;
  validarnum(num,sw);

  if (sw=true)then
    showmessage('Digite solo números')
  else
  begin

    if (strtoint(num)<6) or (strtoint(num)>=0)then
    begin
       bina:=stringGrid1.cells[1,cuentaterm];
       long:=length(bina);
       i:=1;
       cuenta:=0;
       while(i<=long)do
       begin
         if (bina[i]='1') then
           cuenta:=cuenta+1;
       i:=i+1;
       end;
       if (inttostr(cuenta)<>edit1.Text)then
       begin
        panel3.Visible:=True;
        label56.Caption:=inttostr(cuenta);
        label58.Caption:=edit1.Text+'.';
        label56.Visible:=True;
        label58.Visible:=True;
       end;
       edit1.clear;
```

```
      stringGrid1.Cells[3,cuentaterm]:='indice '+inttostr(cuenta);
      cuentaterm:=cuentaterm+1;
    end
    else
showmessage('El número mínimo de unos es 0, y el máximo es 6');
  end;
end;
if (cuentaterm=numterminos)then
begin
label8.visible:=true;
Memo1.visible:=true;
BotonOk.visible:=true;
Memo2.visible:=false;
edit1.visible:=false;
speedbutton1.visible:=false;
with stringGrid1 do
   for j:=0 to colcount-1 do
     for k:=0 to rowcount-1 do
      mat[j,k]:=cells[j,k];
ordenmat(mat);
with stringGrid1 do
   for j:=0 to colcount-1 do
     for k:=0 to rowcount-1 do
      cells[j,k]:=mat[j,k+1];
end;
end;
```

PROCEDIMIENTO ESCOGER_VARIABLES. Controla que el número de variables no esté en conflicto con el número de términos previamente escogidos por el usuario.

```
procedure TForm13.SpeedButton15Click(Sender: TObject);
begin
if (cuentavar<2) then
  showmessage('Debe seleccionar mas de una variable')
else
  if (numterminos=5) and (cuentavar<3)then
   showmessage('Para una expresión de cinco términos, debes escoger
mínimo tres variables')
  else
  if (numterminos=6) and (cuentavar<3)then
   showmessage('Para una expresión de seis términos, debes escoger
mínimo tres variables')
  else
  if (numterminos=7) and (cuentavar<3)then
```

```
   showmessage('Para una expresión de siete términos, debes escoger
mínimo tres variables')
  else
  if (numterminos=8) and (cuentavar<3)then
   showmessage('Para una expresión de ocho términos, debes escoger
mínimo tres variables')
  else
  if (numterminos=9) and (cuentavar<4)then
   showmessage('Para una expresión de nueve términos, debes escoger
mínimo cuatro variables')
  else
  if (numterminos=10) and (cuentavar<4)then
   showmessage('Para una expresión de diez términos, debes escoger
mínimo cuatro variables')
  else
  begin
   Form13.Hide;
   Form12.Show;
   if (form13.BD.Enabled=False)then
   begin
    form12.BD.Visible:=True;
    form12.BD1.Visible:=True;
    form12.BD.Enabled:=True;
    form12.BD1.Enabled:=True;
   end
   else
   begin
    form12.BD.Enabled:=False;
    form12.BD1.Enabled:=False;
   end;

   if ((form13.BB1.Enabled=False) or (form13.BB2.Enabled=False) or
(form13.BB3.Enabled=False))then
   begin
    form12.BB.Visible:=True;
    form12.BB1.Visible:=True;
    form12.BB.Enabled:=True;
    form12.BB1.Enabled:=True;
   end
   else
   if ((form13.BB1.Enabled=True) or (form13.BB2.Enabled=True) or
(form13.BB3.Enabled=True))then
   begin
    form12.BB.Enabled:=False;
    form12.BB1.Enabled:=False;
   end;

   if ((form13.BC1.Enabled=False) or (form13.BC2.Enabled=False) or
(form13.BC3.Enabled=False))then
   begin
    form12.BC.Visible:=True;
```

```
   form12.BC1.Visible:=True;
   form12.BC.Enabled:=True;
   form12.BC1.Enabled:=True;
  end
  else
  if ((form13.BC1.Enabled=True) or (form13.BC2.Enabled=True) or
(form13.BC3.Enabled=True))then
  begin
   form12.BC.Enabled:=False;
   form12.BC1.Enabled:=False;
  end;

  if ((form13.BE1.Enabled=False) or (form13.BE2.Enabled=False) or
(form13.BE3.Enabled=False))then
  begin
   form12.BE.Visible:=True;
   form12.BE1.Visible:=True;
   form12.BE.Enabled:=True;
   form12.BE1.Enabled:=True;
  end
  else
  if ((form13.BE1.Enabled=True) or (form13.BE2.Enabled=True) or
(form13.BE3.Enabled=True))then
  begin
   form12.BE.Enabled:=False;
   form12.BE1.Enabled:=False;
  end;

  if ((form13.BA1.Enabled=False) or (form13.BA2.Enabled=False)) then
  begin
   form12.BA.Visible:=True;
   form12.BA1.Visible:=True;
   form12.BA.Enabled:=True;
   form12.BA1.Enabled:=True;
  end
  else
  if ((form13.BA1.Enabled=True) or (form13.BA2.Enabled=True))then
  begin
   form12.BA.Enabled:=False;
   form12.BA1.Enabled:=False;
  end;

  if ((form13.BF1.Enabled=False) or (form13.BF2.Enabled=False)) then
  begin
   form12.BF.Visible:=True;
   form12.BF1.Visible:=True;
   form12.BF.Enabled:=True;
   form12.BF1.Enabled:=True;
  end
  else
  if ((form13.BF1.Enabled=True) or (form13.BF2.Enabled=True))then
```

```
  begin
   form12.BF.Enabled:=False;
   form12.BF1.Enabled:=False;
  end;
  Form12.label1.Visible:=False;
  Form12.label2.Visible:=False;
  Form12.edit1.visible:=false;
  Form12.edit1.Enabled:=false;
  Form12.Aceptar4.Visible:=false;
  form12.aceptar.visible:=true;
  form12.aceptar.enabled:=true;
  Form12.label3.Visible:=True;
  Form12.label4.Visible:=True;
  Form12.StringGrid1.Visible:=True;
end;
ordenterm(TermGral);
end;
```

PROCEDIMIENTO ORDENAR_MATRIZ. Ordena una matriz cuyos datos son de tipo cadena, de menor a mayor según la columna número tres.

```
procedure ordenmat(var mat1:matriz1);
var
sw,i,c: integer;
aux1,aux2,aux3,aux4,a,b: string;
begin
sw:=0;
while(sw=0)do
begin
sw:=1;
  for i:=0 to numterminos-1 do
  begin
  a:=mat1[3,i];
  b:=mat1[3,i+1];
  c:=comparestr(a,b);
  if (c>0) then
  begin
    aux1:=mat1[0,i];
    aux2:=mat1[1,i];
    aux3:=mat1[2,i];
    aux4:=mat1[3,i];
    mat1[0,i]:=mat1[0,i+1];
    mat1[1,i]:=mat1[1,i+1];
    mat1[2,i]:=mat1[2,i+1];
```

```
    mat1[3,i]:=mat1[3,i+1];
    mat1[0,i+1]:=aux1;
    mat1[1,i+1]:=aux2;
    mat1[2,i+1]:=aux3;
    mat1[3,i+1]:=aux4;
    sw:=0;
  end;
  end;
end;
end;
```

PROCEDIMIENTO PROPUESTOS_McCluskey. Selecciona la evaluación que se utiliza en el modulo tabla de agrupamiento base de la lección de Quine McCluskey.

```
procedure TForm14.Propuestos1Click(Sender: TObject);
var
  i,j:integer;
  sw:boolean;
begin
 PropTab:=PropTab+1;
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Cod;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
   Estudiantes.Edit;
   niv:=Estudiantes.Fields[6].AsInteger;
 end;
 Estudiantes.Close;
 Leccion_McCluskey.Open;
 Leccion_McCluskey.SetKey;
 Leccion_McCluskey.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_McCluskey.GotoKey;
 Leccion_McCluskey.Edit;
 EjerSiNo:=Leccion_McCluskey.Fields[5].AsString;
 Leccion_McCluskey.Close;
 Ejercicios_Propuestos.Open;
 Ejercicios_Propuestos.DisableControls;
 sw:=False;
 Ejercicios_Propuestos.First;
 while ((not(Ejercicios_Propuestos.EOF)) and (sw=False)) do
 begin
   Ejercicios_Propuestos.Edit;
```

```
   if (Ejercicios_Propuestos.Fields[0].AsString='Tabla de Agrupamiento
Base')and (Ejercicios_Propuestos.Fields[2].Asinteger=niv) then
    if (EjerSiNo[Ejercicios_Propuestos.Fields[7].AsInteger]='0') then
     sw:=True;
   if sw=false then
    Ejercicios_Propuestos.Next;
 end;
 if (sw) then
 begin
  Funcion:=Ejercicios_Propuestos.Fields[1].asstring;
  FEvaluarTabla.Memo4.Lines.Text:='X='+Funcion;
  NumT:=0;
  ContarTerm(Funcion,NumT);
  FEvaluarTabla.StringGrid1.Height:=20*NumT;
//  FEvaluarTabla.StringGrid1.Height:=20*(NumT+1);
  with FEvaluarTabla.StringGrid1 do
  begin
   RowCount:=NumT;
   for i:=0 to 3 do
    for j:=0 to NumT-1 do
     Cells[i,j]:='';
   end;
   with FEvaluarTabla.StringGrid1 do
   for i:=0 to NumT-1 do
     Vec[i]:='';
   FuncVec(Funcion,Vec);
   with FEvaluarTabla.StringGrid1 do
   begin
    Height:=NumT*19;
    RowCount:=NumT;
    for i:=0 to NumT-1 do
      Cells[0,i]:=Vec[i];
   end;
  col:=1;
  FEvaluarTabla.B0.Enabled:=True;
  FEvaluarTabla.B1.Enabled:=True;
  FEvaluarTabla.B2.Enabled:=False;
  FEvaluarTabla.B3.Enabled:=False;
  FEvaluarTabla.B4.Enabled:=False;
  FEvaluarTabla.B5.Enabled:=False;
  FEvaluarTabla.B6.Enabled:=False;
  FEvaluarTabla.B7.Enabled:=False;
  FEvaluarTabla.B8.Enabled:=False;
  FEvaluarTabla.B9.Enabled:=False;
  FEvaluarTabla.BAceptar.Enabled:=true;
  FEvaluarTabla.Show;
  form14.Hide;
 end
 else
  showmessage('No hay más ejercicios sobre este tema en este nivel.');
end;
```

PROCEDIMIENTO RESUELTOS_McCLUSKEY. Selecciona un ejercicio resuelto para que sea consultado por el usuario del módulo tabla de agrupamiento base de la lección de Quine McCluskey.

```
procedure TForm14.Resueltos1Click(Sender: TObject);
var
 i,j,k:integer;
 sw:boolean;
begin
 ResTab:=ResTab+1;
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Cod;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
   Estudiantes.Edit;
   niv:=Estudiantes.Fields[6].AsInteger;
 end;
 Estudiantes.Close;
 Leccion_McCluskey.Open;
 Leccion_McCluskey.SetKey;
 Leccion_McCluskey.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_McCluskey.GotoKey;
 Leccion_McCluskey.Edit;
 EjerSiNo:='';
 for k:=1 to 255 do
  EjerSiNo:=EjerSiNo+VEjerR[k];
 Leccion_McCluskey.Close;
 Ejercicios_Resueltos.Open;
 Ejercicios_Resueltos.DisableControls;
 sw:=False;
 Ejercicios_Resueltos.First;
 while ((not(Ejercicios_Resueltos.EOF)) and (sw=False)) do
 begin
   Ejercicios_Resueltos.Edit;
   if (Ejercicios_Resueltos.Fields[0].AsString='Tabla de Agrupamiento
Base')and (Ejercicios_Resueltos.Fields[1].Asinteger=niv) then
    if (EjerSiNo[Ejercicios_Resueltos.Fields[2].AsInteger]='0') then
     sw:=True;
  if sw=false then
```

```
   Ejercicios_Resueltos.Next;
 end;
 if (sw) then
 begin
  Funcion:=Ejercicios_Resueltos.Fields[3].asstring;
  FEjerMCRes.Memo4.Lines.Text:='X='+Funcion;
  VEjerR[Ejercicios_Resueltos.Fields[2].AsInteger]:='1';
  NumT:=0;
  ContarTerm(Funcion,NumT);
  FEjerMCRes.StringGrid1.Height:=19*(NumT+1);
  with FEjerMCRes.StringGrid1 do
  begin
   RowCount:=NumT;
   for i:=0 to 3 do
    for j:=0 to NumT-1 do
     Cells[i,j]:='';
   end;
   with FEjerMCRes.StringGrid1 do
   for i:=0 to NumT-1 do
     Vec[i]:='';
   FuncVec(Funcion,Vec);
   with FEjerMCRes.StringGrid1 do
   begin
    Height:=NumT*19;
    RowCount:=NumT;
    for i:=0 to NumT-1 do
      Cells[0,i]:=Vec[i];
   end;
  col:=1;
  FEjerMCRes.PageControl1.ActivePage:=FEjerMCRes.PTablaAgr;
  FEjerMCRes.Image1.Visible:=True;
  FEjerMCRes.Show;
  form14.Hide;
 end
 else
  showmessage('No hay más ejercicios sobre este tema en este nivel.');
end;
```

PROCEDIMIENTO DECIMAL_BINARIO. Toma una cadena que lo contiene un número decimal y lo convierte en una cadena binaria de unos y ceros.

```
procedure decbin(longceld:integer;var dec:string);
var
 bin,num,long,i: integer;
```

```
 cad,num1:string;
begin
 num:=strtoint(dec);
 dec:='';
 repeat
  bin:=num mod 2;
  dec:=dec+inttostr(bin);
  num:=num div 2;
 until(num<2);
 num1:=IntToStr(num);
 dec:=dec+num1;
 long:=length(dec);
 cad:='';
 for i:=long downto 1 do
  cad:=cad+dec[i];
 if (longceld>long) then
  for i:=1 to (longceld-long) do
   cad:='0'+cad;
 dec:=cad;
end;
```

PROCEDIMIENTO COMPARE_CADENAS. Compara dos cadenas y genera una tercera cadena donde a los teminos diferentes se les asigna (-).

```
procedure comparecad(cadena1,cadena2:string; var cadena3:string; var
sw:boolean);
var
 long,i,c:integer;
begin
 long:=length(cadena1);
 c:=0;
 sw:=false;
 for i:=1 to long do
  if cadena1[i]=cadena2[i] then
   cadena3:=cadena3+cadena1[i]
  else
  begin
   cadena3:=cadena3+'-';
   c:=c+1;
  end;
 if c=1 then
  sw:=true;
end;
```

PROCEDIMIENTO AGRUPAMIENTO. Realiza la primera reducción sobre la tabla de agrupamiento base del método de simplificación de Quine McCluskey.

```
procedure agrupamiento(StringGrid1:
TStringGrid;mat1:agrupa;col1:integer;col2:integer;col3:integer;var mat2:
agrupa;var cont:integer);
var
 i,j,long,c,c1: integer;
 indice,indice2,cadena:string;
 sw,sw1:boolean;
begin
 j:=0;
 while(j<=stringGrid1.rowcount-1) do
 begin
  indice:=mat1[col1,j];
  i:=j+1;
  sw:=false;
  while(i<=stringGrid1.rowcount-1)and (sw=false)do
   if indice=mat1[col1,i] then
     i:=i+1
   else
   begin
     sw:=true;
     indice2:=mat1[col1,i];
   end;
  if (sw) then
   while(mat1[col1,i]=indice2)and(i<=stringGrid1.rowcount-1)do
   begin
    cadena:='';
    comparecad(mat1[col2,j],mat1[col2,i],cadena,sw1);
    if (sw1) then
    begin
     mat2[0,cont]:=cadena;
     mat2[1,cont]:=mat1[col3,j]+','+mat1[col3,i];
     long:=length(cadena);
     c1:=0;
     for c:=1 to long do
      if cadena[c]='1' then
       c1:=c1+1;
       mat2[2,cont]:='indice'+inttostr(c1);
       cont:=cont+1;
    end;
    i:=i+1;
```

```
   end;
  j:=j+1;
 end;
end;
```

PROCEDIMIENTO ORD_VECT. Ordena un vector

```
procedure ordenvect(tam:integer;var vect:vector);
var
 aux:integer;
 sw:boolean;
 i,j:integer;
begin
 sw:=true;
 i:=0;
 while(i<=tam+1) and (sw)do
 begin
  sw:=false;
  for j:=0 to tam-i-1 do
   if (vect[j]>vect[j+1]) then
   begin
    sw:=true;
    aux:=vect[j];
    vect[j]:=vect[j+1];
    vect[j+1]:=aux;
   end;
  i:=i+1;
 end;
end;
```

LECCIÓN SIMPLIFICACIÓN ALGEBRAICA.

PROCEDIMIENTO INICIAR_SIMPLIFICACION. Cuando un usuario visita la lección Simplificación Algebraica, actualiza los datos de éste en las tablas Estudiantes y Lección_Simplificación.

```
procedure TForm7.Bsimplifi
```

```
cClick(Sender: TObject);
begin
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
  Estudiantes.Edit;
  Estudiantes.Fields[4].AsString:='Simplificación Algebraica';
  Estudiantes.Post;
  Estudiantes.EnableControls;
  Estudiantes.Close;
 end;
 Leccion_Simplificacion.Open;
 Leccion_Simplificacion.SetKey;
 Leccion_Simplificacion.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_Simplificacion.GotoKey;
 if (Leccion_Simplificacion.GotoKey=True) then
 begin
  Leccion_Simplificacion.Edit;

Leccion_Simplificacion.Fields[1].AsInteger:=Leccion_Simplificacion.Field
s[1].AsInteger+1;
  Leccion_Simplificacion.Post;
  Leccion_Simplificacion.EnableControls;
  Leccion_Simplificacion.Close;
 end
 else
  begin
   Leccion_Simplificacion.DisableControls;
   Leccion_Simplificacion.Last;
   Leccion_Simplificacion.Insert;
   Leccion_Simplificacion.Edit;
   Leccion_Simplificacion.Fields[0].AsString:=Codigo1;
   Leccion_Simplificacion.Fields[1].AsInteger:=1;
   Leccion_Simplificacion.Fields[2].AsFloat:=0;
   Leccion_Simplificacion.Fields[3].AsInteger:=0;

Leccion_Simplificacion.Fields[4].AsString:='000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000';

Leccion_Simplificacion.Fields[5].AsString:='000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000';
```

```
Leccion_Simplificacion.Fields[6].AsString:='0000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000';

Leccion_Simplificacion.Fields[7].AsString:='0000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000';
   Leccion_Simplificacion.post;
   Leccion_Simplificacion.EnableControls;
   Leccion_Simplificacion.Close;
end;
 form7.hide;
 FSimpAlg0.Show;
end;
```

PROCEDIMIENTO TEOREMA_CUATRO. Determina si el teorema X-X=0 del Álgebra booleana es aplicable al ejercicio que se le ha propuesto resolver al usuario en la lección simplificación Algebraica.

```
procedure TFSimpAlg.B4Click(Sender: TObject);
begin
 if (Ejercicio=VEjerc[1]) then
 begin
  if (Cont_T=0) then
    MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[2]) then
 begin
  if (Cont_T=0) then
    MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[3]) then
 begin
  if (Cont_T=0) then
```

```
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[4]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[5]) then
 begin
  if (Cont_T=0) then
  begin
   MostOcul;
   showmessage('Esta ley no es aplicable en este caso');
  end
  else
   if (Cont_T=1) then
   begin
    label29.Visible:=True;
    Ecuacion:='X=Q-R+-QR';
showmessage('Hemos obtenido el resultado X=Q-R+-QR aplicando el
teorema 4');
    Dos.Visible:=True;
    Dos.Font.Color:=clWhite;
    Dos.Caption:=Ecuacion;
    Cont_T:=0;
    DeshBot(FSimpAlg);
{    FSimpAlg2.Show;
    FSimpAlg.Hide;}
   end
   else
    showmessage('Esta ley no es aplicable en este caso');
  end;
end;
```

PROCEDIMIENTO TEOREMA_DIECISIETE. Determina si el teorema –(XY)=-X+-Y del Álgebra booleana es aplicable al ejercicio que se le ha propuesto resolver al usuario en la lección simplificación Algebraica.

```
procedure TFSimpAlg.B17Click(Sender: TObject);
begin
 if (Ejercicio=VEjerc[1]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[2]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[3]) then
 begin
  if (Cont_T=0) then
  begin
   OculLB;
   Label29.Visible:=False;
   LEcuacion.Visible:=True;
   LEcuacion.Caption:=Ejercicio;
   Uno.Visible:=True;
   Ecuacion:='X=(-R+-S+-T)-(R+S+T)';
   Cont_T:=Cont_T+1;
   Uno.Caption:=Ecuacion;
   showmessage('Obtuvimos este resultado aplicando la Ley 17 al
término -(RST)');
  end
  else
   if ((Cont_T=1) and (Ecuacion='X=-(RST)-R-S-T')) then
   begin
    Dos.Visible:=True;
    Ecuacion:='X=(-R+-S+-T)-R-S-T';
    Dos.Caption:=Ecuacion;
    showmessage('Obtuvimos este resultado aplicando la Ley 17 al
término -(RST)');
    Cont_T:=Cont_T+1;
   end
   else
    showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[4]) then
 begin
  if (Cont_T=0) then
  begin
   OculLB;
   Label29.Visible:=False;
```

```
   LEcuacion.Visible:=True;
   LEcuacion.Caption:=Ejercicio;
   Uno.Visible:=True;
   Ecuacion:='X=ABC+A-B(--A+--C)';
   Cont_T:=Cont_T+1;
   Uno.Caption:=Ecuacion;
   showmessage('Obtuvimos este resultado aplicando la Ley 17 al término -(-A-C)');
  end
  else
   showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[5]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end;
end;
```

PROCEDIMIENTO TEOREMA_DIECIOCHO. Determina si el teorema –X=X del Álgebra booleana es aplicable al ejercicio que se le ha propuesto resolver al usuario en la lección simplificación Algebraica.

```
procedure TFSimpAlg.B18Click(Sender: TObject);
begin
 if (Ejercicio=VEjerc[1]) then
 begin
  if (Cont_T=0) then
   MostOcul;
   showmessage('En este caso no es posible cancelar las inversiones dobles');
 end
 else
 if (Ejercicio=VEjerc[2]) then
 begin
  if (Cont_T=0) then
   MostOcul;
   showmessage('En este caso no es posible cancelar las inversiones dobles');
 end
 else
```

```
 if (Ejercicio=VEjerc[3]) then
 begin
  if (Cont_T=0) then
   MostOcul;
   showmessage('En este caso no es posible cancelar las inversiones
dobles');
 end
 else
 if (Ejercicio=VEjerc[4]) then
 begin
  if (Cont_T=0) then
  begin
   MostOcul;
   showmessage('En este caso no es posible cancelar las inversiones
dobles');
  end
  else
   if (Cont_T=1) then
   begin
    Dos.Visible:=True;
    Ecuacion:='X=ABC+A-B(A+C)';
    Dos.Caption:=Ecuacion;
    Cont_T:=Cont_T+1;
    showmessage('Obtuvimos este resultado cancelando las inversiones
dobles en los términos --A y --C');
   end
   else
     showmessage('En este caso no es posible cancelar las inversiones
dobles');
 end
 else
 if (Ejercicio=VEjerc[5]) then
 begin
  if (Cont_T=0) then
   MostOcul;
   showmessage('En este caso no es posible cancelar las inversiones
dobles');
 end;
end;
```

PROCEDIMIENTO TEOREMA_TRECE. Determina si el teorema X(Y+Z)=XY+XZ del Álgebra booleana es aplicable al ejercicio que se le ha propuesto resolver al usuario en la lección simplificación Algebraica.

```
procedure TFSimpAlg.B13AClick(Sender: TObject);
begin
 if (Ejercicio=VEjerc[1]) then
 begin
  if (Cont_T=0) then
  begin
   OculLB;
   LEcuacion.Visible:=True;
   LEcuacion.Caption:=Ejercicio;
   Uno.Visible:=True;
   Ecuacion:='X=BC(-A+A)+-B-C(-A+A)+A-BC';
   Cont_T:=Cont_T+1;
   Uno.Caption:=Ecuacion;
   showmessage('Hemos agrupado los términos -ABC+ABC y A-B-C+-A-
B-C aplicando el teorema 13A');
  end
  else
   if (Cont_T=2) then
   begin
    Tres.Visible:=True;
    Ecuacion:='X=BC+-B(-C+AC)';
    Tres.Caption:=Ecuacion;
    Cont_T:=Cont_T+1;
    showmessage('Hemos agrupado los términos -B-C+A-BC aplicando el
teorema 13A');
   end
   else
   showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[2]) then
 begin
  if (Cont_T=0) then
  begin
   OculLB;
   LEcuacion.Visible:=True;
   LEcuacion.Caption:=Ejercicio;
   Uno.Visible:=True;
   Ecuacion:='X=-(C+D)+C-D(A+-A)+-B(A-C+-ACD)';
   Cont_T:=Cont_T+1;
   Uno.Caption:=Ecuacion;
Showmessage('Hemos agrupado los términos -ACD+AC-D aplicando el
teorema 13A');
  end
  else
   if ((Cont_T=1) and (Ecuacion='X=-C-D+-AC-D+A-B-C+-A-BCD+AC-D'))
then
   begin
    Dos.Visible:=True;
```

```
    Ecuacion:='X=-C-D+C-D(-A+A)+-B(A-C+-ACD)';
    Dos.Caption:=Ecuacion;
    Cont_T:=Cont_T+1;
    showmessage('Obtuvimos este resultado aplicando el Teorema 13A a
los términos -AC-D+AC-D y A-B-C+A-BCD');
   end
   else
    if (Cont_T=3) then
    begin
     Cuatro.Visible:=True;
     Ecuacion:='X=-D(-C+C)+-B(A-C+-ACD)';
     Cuatro.Caption:=Ecuacion;
     Cont_T:=Cont_T+1;
     showmessage('Obtuvimos este resultado aplicando el Teorema 13A a
los términos -C-D+C-D');
    end
    else
     showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[3]) then
 begin
  if (Cont_T=0) then
  begin
   MostOcul;
   showmessage('Esta ley no es aplicable en este caso');
  end
  else
   if (Cont_T=2) then
   begin
    Tres.Visible:=True;
    Ecuacion:='X=-R-R-S-T+-S-R-S-T+-T-R-S-T';
    Tres.Caption:=Ecuacion;
    showmessage('Obtuvimos este resultado aplicando el Teorema 13A a
los términos (-R+-S+-T)-R-S-T');
    Cont_T:=Cont_T+1;
   end
   else
    showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[4]) then
 begin
  if (Cont_T=0) then
  begin
   MostOcul;
   showmessage('Esta ley no es aplicable en este caso');
  end
  else
   if (Cont_T=2) then
   begin
```

```
    Tres.Visible:=True;
    Ecuacion:='X=ABC+A-BA+A-BC';
    Tres.Caption:=Ecuacion;
    Cont_T:=Cont_T+1;
    showmessage('Obtuvimos este resultado aplicando la Ley 13A al
término A-B(A+C)');
   end
   else
    if (Cont_T=4) then
    begin
     Cinco.Visible:=True;
     Ecuacion:='X=AC(B+-B)+A-B';
     cinco.Caption:=Ecuacion;
     Cont_T:=Cont_T+1;
     showmessage('Obtuvimos este resultado aplicando el Teorema 13A
agrupando los términos ABC+A-BC');
    end
    else
     if (Cont_T=6) then
     begin
      label29.Visible:=True;
      Siete.Visible:=True;
      Siete.Font.Color:=clWhite;
      Ecuacion:='X=A(-B+C)';
      Siete.Caption:=Ecuacion;
      Cont_T:=0;
       showmessage('Obtuvimos este resultado aplicando el Teorema 13A
a los términos AC+A-B');
      DeshBot(FSimpAlg);
{      FSimpAlg2.Show;
      FSimpAlg.Hide;}
     end
     else
      showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[5]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end;
end;
```

PROCEDIMIENTO TEOREMA_TRES. Determina si el teorema X· X=X del Álgebra booleana es aplicable al ejercicio que se le ha

propuesto resolver al usuario en la lección simplificación Algebraica.

```
procedure TFSimpAlg.B3Click(Sender: TObject);
begin
 if (Ejercicio=VEjerc[1]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[2]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[3]) then
 begin
  if (Cont_T=0) then
  begin
   MostOcul;
   showmessage('Esta ley no es aplicable en este caso');
  end
  else
   if (Cont_T=3) then
   begin
    Cuatro.Visible:=True;
    Ecuacion:='X=-R-S-T+-S-R-T+-T-R-S';
    Cuatro.Caption:=Ecuacion;
    showmessage('Obtuvimos este resultado aplicando el Teorema 3 a los
término -R-R-S-T+-S-S-R-T+-T-R-S-T');
    Cont_T:=Cont_T+1;
   end
   else
    showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[4]) then
 begin
  if (Cont_T=0) then
  begin
   MostOcul;
   showmessage('Esta ley no es aplicable en este caso');
  end
```

```
  else
   if (Cont_T=3) then
   begin
    Cuatro.Visible:=True;
    Ecuacion:='X=ABC+A-B+A-BC';
    Cuatro.Caption:=Ecuacion;
    Cont_T:=Cont_T+1;
     showmessage('Obtuvimos este resultado aplicando el Teorema 3 al
término A-BA');
   end
   else
    showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[5]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end;
end;
```

PROCEDIMIENTO TEOREMA_DIECISEIS. Determina si el teorema -(X+Y)=-X-Y del Álgebra booleana es aplicable al ejercicio que se le ha propuesto resolver al usuario en la lección simplificación Algebraica.

```
procedure TFSimpAlg.B16Click(Sender: TObject);
begin
 if (Ejercicio=VEjerc[1]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[2]) then
 begin
  if (Cont_T=0) then
  begin
   OculLB;
   LEcuacion.Visible:=True;
   LEcuacion.Caption:=Ejercicio;
   Uno.Visible:=True;
   Ecuacion:='X=-C-D+-AC-D+A-B-C+-A-BCD+AC-D';
```

```
   Cont_T:=Cont_T+1;
   Uno.Caption:=Ecuacion;
   showmessage('Obtuvimos este resultado aplicando la Ley 16 al
término -(C+D)');
  end
  else
   if ((Cont_T=2) and (Ecuacion='X=-(C+D)+C-D+-B(A-C+-ACD)')) then
   begin
    Tres.Visible:=True;
    Ecuacion:='X=-C-D+C-D+-B(A-C+-ACD)';
    Tres.Caption:=Ecuacion;
    Cont_T:=Cont_T+1;
    showmessage('Obtuvimos este resultado aplicando la Ley 16 al
término -(R+S+T)');
   end
   else
    if ((Cont_T=1) and (Ecuacion='X=-(C+D)+C-D(A+-A)+-B(A-C+-ACD)'))
then
    begin
     dos.Visible:=True;
     Ecuacion:='X=-C-D+C-D(A+-A)+-B(A-C+-ACD)';
     Dos.Caption:=Ecuacion;
     Cont_T:=Cont_T+1;
     showmessage('Obtuvimos este resultado aplicando la Ley 16 al
término -(C+D)');
    end
    else
     showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[3]) then
 begin
  if (Cont_T=0) then
  begin
   OculLB;
   LEcuacion.Visible:=True;
   LEcuacion.Caption:=Ejercicio;
   Uno.Visible:=True;
   Ecuacion:='X=-(RST)-R-S-T';
   Cont_T:=Cont_T+1;
   Uno.Caption:=Ecuacion;
   showmessage('Obtuvimos este resultado aplicando la Ley 16 al
término -(R+S+T)');
  end
  else
   if ((Cont_T=1) and (Ecuacion='X=(-R+-S+-T)-(R+S+T)')) then
   begin
    Dos.Visible:=True;
    Ecuacion:='X=(-R+-S+-T)-R-S-T';
    Dos.Caption:=Ecuacion;
    Cont_T:=Cont_T+1;
```

```
   showmessage('Obtuvimos este resultado aplicando la Ley 16 al término -(R+S+T)');
   end
   else
    showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[4]) then
 begin
  if (Cont_T=0) then
   MostOcul;
   showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[5]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end;
end;
```

PROCEDIMIENTO TEOREMA_13B. Determina si el teorema (W+X)(Y+Z)=WY+XY+WZ+XZ del Álgebra booleana es aplicable al ejercicio que se le ha propuesto resolver al usuario en la lección simplificación Algebraica.

```
procedure TFSimpAlg.B13BClick(Sender: TObject);
begin
 if (Ejercicio=VEjerc[1]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[2]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
```

```
 if (Ejercicio=VEjerc[3]) then
 begin
  if (Cont_T=0) then
    MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[4]) then
 begin
  if (Cont_T=0) then
    MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[5]) then
 begin
  if (Cont_T=0) then
  begin
    OculLB;
    LEcuacion.Visible:=True;
    LEcuacion.Caption:=Ejercicio;
    Uno.Visible:=True;
    Ecuacion:='X=Q-Q+Q-R+R-Q+R-R';
    Cont_T:=Cont_T+1;
    Uno.Caption:=Ecuacion;
    showmessage('Hemos obtenido este resultado aplicando el teorema
13B a los términos (Q+R)(-Q+-R))');
  end
  else
  showmessage('Esta ley no es aplicable en este caso');
 end;
end;
```

PROCEDIMIENTO TEOREMA_SIETE. Determina si el teorema X+X=X del Álgebra booleana es aplicable al ejercicio que se le ha propuesto resolver al usuario en la lección simplificación Algebraica.

```
procedure TFSimpAlg.B7Click(Sender: TObject);
begin
 if (Ejercicio=VEjerc[1]) then
 begin
  if (Cont_T=0) then
    MostOcul;
```

```
 showmessage('Esta ley no es aplicable en este caso');
end
else
if (Ejercicio=VEjerc[2]) then
begin
 if (Cont_T=0) then
  MostOcul;
 showmessage('Esta ley no es aplicable en este caso');
end
else
if (Ejercicio=VEjerc[3]) then
begin
 if (Cont_T=0) then
 begin
  MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
  if (Cont_T=4) then
  begin
   label29.Visible:=True;
   Cinco.Visible:=True;
   Cinco.Font.Color:=clWhite;
   Ecuacion:='X=-R-S-T';
   Cinco.Caption:=Ecuacion;
   showmessage('Obtuvimos este resultado aplicando el Teorema 7 a los
términos -R-S-T+-R-S-T+-R-S-T');
   Cont_T:=0;
   DeshBot(FSimpAlg);
{    FSimpAlg2.Show;
   FSimpAlg.Hide;}
  end
  else
   showmessage('Esta ley no es aplicable en este caso');
end
else
if (Ejercicio=VEjerc[4]) then
begin
 if (Cont_T=0) then
  MostOcul;
 showmessage('Esta ley no es aplicable en este caso');
end
else
if (Ejercicio=VEjerc[5]) then
begin
 if (Cont_T=0) then
  MostOcul;
 showmessage('Esta ley no es aplicable en este caso');
end;
end;
```

PROCEDIMIENTO TEOREMA_OCHO. Determina si el teorema X+-X=1 del Álgebra booleana es aplicable al ejercicio que se le ha propuesto resolver al usuario en la lección simplificación Algebraica.

```
procedure TFSimpAlg.B8Click(Sender: TObject);
begin
 if (Ejercicio=VEjerc[1]) then
 begin
  if (Cont_T=0) then
  begin
   MostOcul;
   showmessage('Esta ley no es aplicable en este caso');
  end
  else
   if (Cont_T=1) then
   begin
    Dos.Visible:=True;
    Ecuacion:='X=BC+-B-C+A-BC';
    Dos.Caption:=Ecuacion;
    Cont_T:=Cont_T+1;
    showmessage('Hemos agrupado los términos BC(A+-A) y -B-C(A+-A)
aplicando el teorema 8');
   end
   else
    showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[2]) then
 begin
  if (Cont_T=0) then
  begin
   MostOcul;
   showmessage('Esta ley no es aplicable en este caso');
  end
  else
   if ((Cont_T=2) and (Ecuacion='X=-C-D+C-D(-A+A)+-B(A-C+-ACD)'))
then
   begin
    Tres.Visible:=True;
    Ecuacion:='X=-C-D+C-D+-B(A-C+-ACD)';
    Tres.Caption:=Ecuacion;
    Cont_T:=Cont_T+1;
```

```
   showmessage('Hemos obtenido este resultado aplicando el Teorema
13A al término C-D(A+-A)');
   end
   else
    if ((Cont_T=4) and (Ecuacion='X=-D(-C+C)+-B(A-C+-ACD)'))then
    begin
     label29.Visible:=True;
     Cinco.Visible:=True;
     Cinco.Font.Color:=clWhite;
     Ecuacion:='X=-D+-B(A-C+-ACD)';
     Cinco.Caption:=Ecuacion;
     Cont_T:=0;
     showmessage('Hemos obtenido este resultado aplicando el Teorema
13A al término -D(C+-C)');
     DeshBot(FSimpAlg);
{     FSimpAlg2.Show;
     FSimpAlg.Hide;}
    end
    else
     if ((Cont_T=1) and (Ecuacion='X=-(C+D)+C-D(A+-A)+-B(A-C+-ACD)'))
then
     begin
      Dos.Visible:=True;
      Ecuacion:='X=-(C+D)+C-D+-B(A-C+-ACD)';
      Dos.Caption:=Ecuacion;
      Cont_T:=Cont_T+1;
      showmessage('Hemos obtenido este resultado aplicando el Teorema
13A al término C-D(A+-A)');
     end
     else
      showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[3]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[4]) then
 begin
  if (Cont_T=0) then
  begin
   MostOcul;
   showmessage('Esta ley no es aplicable en este caso');
  end
  else
   if (Cont_T=5) then
   begin
    Seis.Visible:=True;
```

```
    Ecuacion:='X=AC+A-B';
    seis.Caption:=Ecuacion;
    Cont_T:=Cont_T+1;
     showmessage('Obtuvimos este resultado aplicando el Teorema 8 al
término AC(B+-B)');
   end
   else
    showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[5]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end;
end;
```

PROCEDIMIENTO TEOREMA_QUINCE. Determina si el teorema X+-XY=X+Y del Álgebra booleana es aplicable al ejercicio que se le ha propuesto resolver al usuario en la lección simplificación Algebraica.

```
procedure TFSimpAlg.B15Click(Sender: TObject);
begin
 if (Ejercicio=VEjerc[1]) then
 begin
  if (Cont_T=0) then
  begin
   MostOcul;
   showmessage('Esta ley no es aplicable en este caso');
  end
  else
   if (Cont_T=3) then
   begin
    label29.Visible:=True;
    Cuatro.Visible:=True;
    Cuatro.Font.Color:=clWhite;
    Ecuacion:='X=BC+-B(-C+A)';
    Cuatro.Caption:=Ecuacion;
    showmessage('Hemos agrupado el término -B(-C+CA) aplicando el
teorema 15');
    Cont_T:=0;
```

```
   DeshBot(FSimpAlg);
{    FSimpAlg2.Show;
   FSimpAlg.Hide;}
  end
  else
   showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[2]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[3]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[4]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end
 else
 if (Ejercicio=VEjerc[5]) then
 begin
  if (Cont_T=0) then
   MostOcul;
  showmessage('Esta ley no es aplicable en este caso');
 end;
end;
```

PROCEDIMIENTO RESUELTOS_SIMPL_ALGEBRAICA. Selecciona un ejercicio resuelto del tema Simplificación Algebraica para mostrar al usuario.

```
procedure TFSimpAlg.Resueltos1Click(Sender: TObject);
var
  sw:boolean;
begin
```

```
 NSimpRes:=NSimpRes+1;
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
  Estudiantes.Edit;
  niv:=Estudiantes.Fields[6].AsInteger;
 end;
 Estudiantes.Close;
 Leccion_Simplificacion.Open;
 Leccion_Simplificacion.SetKey;
Leccion_Simplificacion.FieldByName('Cod_Est').AsString:=
Codigo1;
 Leccion_Simplificacion.GotoKey;
 Leccion_Simplificacion.Edit;
 EjerSiNo:=Leccion_Simplificacion.Fields[4].AsString;
 Leccion_Simplificacion.Close;
 Ejercicios_Resueltos.Open;
 Ejercicios_Resueltos.DisableControls;
 sw:=False;
 Ejercicios_Resueltos.First;
 while ((not(Ejercicios_Resueltos.EOF)) and (sw=False)) do
 begin
   Ejercicios_Resueltos.Edit;
if (Ejercicios_Resueltos.Fields[0].AsString = 'Simplificacion Algebraica')
and (Ejercicios_Resueltos.Fields[1].Asinteger=niv)  then
   begin
    if (Ejercicios_Resueltos.Fields[2].Asinteger=Num_Res) then
    begin
     sw:=True;
     Num_Res:=Num_Res+1;
    end
    else
     Ejercicios_Resueltos.Next;
   end
   else
    if (sw=false) then
     Ejercicios_Resueltos.Next;
 end;
 if (sw) then
 begin
FEjerSARes.Memo2.Text:=Ejercicios_Resueltos.Fields[4].asString;
FEjerSARes.Memo1.Text:=Ejercicios_Resueltos.Fields[3].asString;
FEjerSARes.Show;
FSimpAlg.Hide;
 end
 else
  showmessage('No hay más ejercicios sobre este tema en este nivel.');
end;
```

PROCEDIMIENTO PROPUESTOS_SIMPL_ALGEBRAICA. Selecciona un ejercicio para evaluar al estudiante en la lección simplificación algebraica.

```
procedure TFSimpAlg.Propuestos1Click(Sender: TObject);
var
 sw:boolean;
begin
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
   Estudiantes.Edit;
   niv:=Estudiantes.Fields[6].AsInteger;
 end;
 Estudiantes.Close;
 Leccion_Simplificacion.Open;
 Leccion_Simplificacion.SetKey;
 Leccion_Simplificacion.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_Simplificacion.GotoKey;
 Leccion_Simplificacion.Edit;
 case niv of
  1:EjerSiNo:=Leccion_Simplificacion.Fields[4].AsString;
  2:EjerSiNo:=Leccion_Simplificacion.Fields[5].AsString;
  3:EjerSiNo:=Leccion_Simplificacion.Fields[6].AsString;
  4:EjerSiNo:=Leccion_Simplificacion.Fields[7].AsString;
 end;
 Leccion_Simplificacion.Close;
 Ejercicios_Propuestos.Open;
 Ejercicios_Propuestos.DisableControls;
 sw:=False;
 Ejercicios_Propuestos.First;
 while ((not(Ejercicios_Propuestos.EOF)) and (sw=False)) do
 begin
  Ejercicios_Propuestos.Edit;
  if               (Ejercicios_Propuestos.Fields[0].AsString='Simplificación
Algebraica')and (Ejercicios_Propuestos.Fields[2].Asinteger=niv) then
    if (EjerSiNo[Ejercicios_Propuestos.Fields[7].AsInteger]='0') then
     sw:=True;
  if sw=false then
   Ejercicios_Propuestos.Next;
 end;
```

```
 if (sw) then
 begin
  PropSA:=PropSA+1;
  Ejercicio:=Ejercicios_Propuestos.Fields[1].asstring;
//Ejercicio:='X=-(A+B+-C+-D+E)+-A-B-C-E+ABCD-E+ABC+-A-BCD-E';
  FEvaluarSA.M28.Text:=Ejercicio;
  FEvaluarSA.Show;
  FSimpAlg.Hide;
 end
 else
  showmessage('No hay más ejercicios sobre este tema en este nivel.');
end;
```

PROCEDIMIENTO EVALUAR_SIMPLIFICA13A. Permite aplicar el teorema X(Y+Z)=XY+XZ del álgebra booleana en la evaluación sobre simplificación algebraica, una vez lo ha seleccionado el usuario.

```
procedure TFEvaluarSA.B13AClick(Sender: TObject);
begin
 if (Ejercicio='S=-P-Q-R+-P-QR+-PQ-R+-PQR+PQR') then
  if (Cont_T=0) then
  begin
   ContB:=ContB+1;
   OculL;
   LEcuacion.Visible:=True;
   LEcuacion.Text:=Ejercicio;
   Uno.Visible:=True;
   Ecuacion:='S=-P-Q(-R+R)+-PQ(-R+R)+PQR';
   Cont_T:=Cont_T+1;
   Uno.Text:=Ecuacion;
   Panel3.Visible:=True;
   Memo1.Text:='Obtuvimos este resultado aplicando el Teorema
   13A a los términos -P-Q-R con -P-QR y -PQ-R con -PQR';
  end
  else
   if ((Cont_T=2) and (Ecuacion='S=-P-Q+-PQ+PQR')) then
   begin
    ContB:=ContB+1;
    Tres.Visible:=True;
    Ecuacion:='S=-P(-Q+Q)+PQR';
    Tres.Text:=Ecuacion;
    Cont_T:=Cont_T+1;
    Panel3.Visible:=True;
```

```
   Memo1.Text:='Obtuvimos este resultado aplicando el Teorema
   13A al término -P-Q con -PQ';
  end
  else
  begin
   Panel4.Visible:=True;
   Memo2.Text:='El Teorema 13A no es aplicable en este caso';
   NError:=NError+1;
  end
else
 if (Ejercicio='X=ABC+-AC') then
  if (Cont_T=0) then
  begin
   ContB:=ContB+1;
   OculL;
   LEcuacion.Visible:=True;
   LEcuacion.Text:=Ejercicio;
   Uno.Visible:=True;
   Ecuacion:='X=C(AB+-A)';
   Cont_T:=Cont_T+1;
   Uno.Text:=Ecuacion;
   Panel3.Visible:=True;
  Memo1.Text:='Obtuvimos este resultado aplicando el Teorema
  13A a los términos ABC y -AC';
  end
  else
  begin
   Panel4.Visible:=True;
   Memo2.Text:='El Teorema 13A no es aplicable en este caso';
   NError:=NError+1;
  end
 else
  if (Ejercicio='X=ABC+A-BC+-A') then
   if (Cont_T=0) then
   begin
    ContB:=ContB+1;
    OculL;
    LEcuacion.Visible:=True;
    LEcuacion.Text:=Ejercicio;
    Uno.Visible:=True;
    Ecuacion:='X=AC(B+-B)+-A';
    Cont_T:=Cont_T+1;
    Uno.Text:=Ecuacion;
    Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el Teorema
    13A a los términos ABC con A-BC';
   end
   else
   begin
    Panel4.Visible:=True;
    Memo2.Text:='El Teorema 13A no es aplicable en este caso';
```

```
  NError:=NError+1;
 end
else
 if (Ejercicio='X=-A-B-C-D+-A-BC-D+-AB-C-D') then
  if (Cont_T=0) then
  begin
   ContB:=ContB+1;
   OculL;
   LEcuacion.Visible:=True;
   LEcuacion.Text:=Ejercicio;
   Uno.Visible:=True;
   Ecuacion:='X=-A-C-D(-B+B)+-A-BC-D';
   Cont_T:=Cont_T+1;
   Uno.Text:=Ecuacion;
   Panel3.Visible:=True;
  Memo1.Text:='Obtuvimos este resultado aplicando el Teorema
  13A a los términos -A-B-C-D con -AB-C-D';
  end
  else
   if ((Cont_T=2) and (Ecuacion='X=-A-C-D+-A-BC-D')) then
   begin
    ContB:=ContB+1;
    Tres.Visible:=True;
    Ecuacion:='X=-A-D(-C+-BC)';
    Tres.Text:=Ecuacion;
    Cont_T:=Cont_T+1;
    Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el Teorema
    13A al término -A-C-D con -A-BC-D';
   end
   else
  if ((Cont_T=1) and (Ecuacion='X=-A-C-D(-B+B)+-A-BC-D')) then
    begin
     ContB:=ContB+1;
     Dos.Visible:=True;
     Ecuacion:='X=-A-D(-C(-B+B)+-BC)';
     Dos.Text:=Ecuacion;
     Cont_T:=Cont_T+1;
     Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el Teorema
    13A al término -A-C-D(-B+B)con -A-BC-D';
    end
    else
    begin
     Panel4.Visible:=True;
     Memo2.Text:='El Teorema 13A no es aplicable en este caso';
     NError:=NError+1;
    end
 else
  if (Ejercicio='X=ABCDE+-A-BCDE+ABCD-E+-A-B-C-DE') then
   if (Cont_T=0) then
```

```
        begin
         ContB:=ContB+1;
         OculL;
         LEcuacion.Visible:=True;
         LEcuacion.Text:=Ejercicio;
         Uno.Visible:=True;
         Ecuacion:='X=ABCD(E+-E)+-A-BE(CD+-C-D)';
         Cont_T:=Cont_T+1;
         Uno.Text:=Ecuacion;
         Panel3.Visible:=True;
         Memo1.Text:='Obtuvimos este resultado aplicando el Teorema
        13A a los términos ABCDE con ABCD-E y -A-BCDE con -A-B-
        C-D-E';
        end
        else
        begin
         Panel4.Visible:=True;
         Memo2.Text:='El Teorema 13A no es aplicable en este caso';
         NError:=NError+1;
        end
       else
    if (Ejercicio='X= -A-B-C-D-E + -(A+B+C+D+E) + ABCDE + AB-CD-E') then
        if (Cont_T=0) then
        begin
         ContB:=ContB+1;
         OculL;
         LEcuacion.Visible:=True;
         LEcuacion.Text:=Ejercicio;
         Uno.Visible:=True;
         Ecuacion:='X=-A-B-C-D-E+-(A+B+C+D+E)+ABD(CE+-C-E)';
         Cont_T:=Cont_T+1;
         Uno.Text:=Ecuacion;
         Panel3.Visible:=True;
         Memo1.Text:='Obtuvimos este resultado aplicando el
         Teorema 13A a el término ABCDE con AB-CD-E';
        end
        else
if ((Cont_T=2) and (Ecuacion='X=-A-B-C-D-E+ABCDE+AB-CD-E')) then
         begin
          ContB:=ContB+1;
          Tres.Visible:=True;
          Ecuacion:='X=-A-B-C-D-E+ABD(CE+-C-E)';
          Tres.Font.Color:=clWhite;
          Tres.Text:=Ecuacion;
          Cont_T:=0;
          DeshBot(FEvaluarSA);
          Panel3.Visible:=True;
          Memo1.Text:='Obtuvimos este resultado aplicando el
          Teorema 13A a los términos ABCDE y AB-CD-E';
          Label29.Visible:=True;
          BRegresar.Visible:=True;
```

```
       BRegresar.Enabled:=True;
      end
      else
if ((Cont_T=1) and (Ecuacion='X=-A-B-C-D-E+-A-B-C-D-E+ABCDE+AB-
CD-E')) then

       begin
        ContB:=ContB+1;
        Dos.Visible:=True;
        Ecuacion:='X=-A-B-C-D-E+-A-B-C-D-E+ABD(CE+-C-E)';
        Dos.Text:=Ecuacion;
        Cont_T:=Cont_T+1;
        Panel3.Visible:=True;
        Memo1.Text:='Obtuvimos este resultado aplicando el
        Teorema 13A al término ABCDE con AB-CD-E';
       end
       else
       begin
      Panel4.Visible:=True;
      Memo2.Text:='El Teorema 13A no es aplicable en este caso';
      NError:=NError+1;
       end
     else
if  (Ejercicio='X=-A-BCD-EF+ABCDEF+AB-CD-EF+A-B-CD-EF+-A-B-C-D-
E-F') then
      if (Cont_T=0) then
      begin
       ContB:=ContB+1;
       OculL;
       LEcuacion.Visible:=True;
       LEcuacion.Text:=Ejercicio;
       Uno.Visible:=True;
      Ecuacion:='X=-A-B-E(CDF+-C-D-F)+ABCDEF+A-CD-EF(B+-B)';
      Cont_T:=Cont_T+1;
      Uno.Text:=Ecuacion;
      Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado aplicando el
      Teorema 13A a los términos -A-BCD-EF con -A-B-C-D-E-F y
      AB-CD-EF con A-B-CD-EF';
      end
      else
       if (Cont_T=2) then
       begin
        ContB:=ContB+1;
        Tres.Visible:=True;
        Ecuacion:='X=-A-B-E(CDF+-C-D-F)+ADF(BCE+-C-E)';
        Tres.Font.Color:=clWhite;
        Tres.Text:=Ecuacion;
        Cont_T:=0;
        DeshBot(FEvaluarSA);
```

```
        Panel3.Visible:=True;
        Memo1.Text:='Obtuvimos este resultado aplicando el
        Teorema 13A a los términos ABCDEF y A-CD-EF';
        Label29.Visible:=True;
        BRegresar.Visible:=True;
        BRegresar.Enabled:=True;
       end
       else
       begin
       Panel4.Visible:=True;
       Memo2.Text:='El Teorema 13A no es aplicable en este caso';
       NError:=NError+1;
       end
      else
if     (Ejercicio='X=-(A+B+-C+D+E+F)+A-B-CDEF+-ABCD-E-F+-(-A+-B+-C+-D+-E+-F)')
then
       if (Cont_T=0) then
       begin
        ContB:=ContB+1;
        OculL;
        LEcuacion.Visible:=True;
        LEcuacion.Text:=Ejercicio;
        Uno.Visible:=True;
        Ecuacion:='X=-(A+B+-C+D+E+F)+D(A-B-CEF+-ABC-E-F)+
        -(-A+-B+-C+-D+-E+-F)';
        Cont_T:=Cont_T+1;
        Uno.Text:=Ecuacion;
        Panel3.Visible:=True;
        Memo1.Text:='Obtuvimos este resultado aplicando el
        Teorema 13A al término A-B-CDEF con -ABCD-E-F';
       end
       else
if ((Cont_T=2) and (Ecuacion='X=-A-BC-D-E-F+A-B-CDEF+-ABCD-E-
F+ABCDEF')) then
        begin
         ContB:=ContB+1;
         Tres.Visible:=True;
         Ecuacion:='X=-AC-E-F(-B-D+BD)+ADEF(-B-C+BC)';
         Tres.Font.Color:=clWhite;
         Tres.Text:=Ecuacion;
         Cont_T:=0;
         DeshBot(FEvaluarSA);
         Panel3.Visible:=True;
         Memo1.Text:='Obtuvimos este resultado aplicando el
         Teorema 13A a los términos -A-BC-D-E-F con -ABCD-E-F y
         A-B-CDEF con ABCDEF';
         Label29.Visible:=True;
         BRegresar.Visible:=True;
         BRegresar.Enabled:=True;
        end
        else
```

```
 if (Cont_T=3) then
begin
ContB:=ContB+1;
Cuatro.Visible:=True;
Ecuacion:='X=-A-BC-D-E-F+D(A-B-CEF+-ABC-E-F+ABCEF)';
Cuatro.Text:=Ecuacion;
Cont_T:=Cont_T+1;
Panel3.Visible:=True;
Memo1.Text:='Obtuvimos este resultado aplicando el
Teorema 13A al término D(A-B-CEF+-ABC-E-F) con
ABCDEF';
 end
 else
  if (Cont_T=4) then
  begin
   ContB:=ContB+1;
   Cinco.Visible:=True;
   Ecuacion:='X=-A-BC-D-E-F+D(AEF(-B-C+BC)+-ABC-E-F)';
   Cinco.Text:=Ecuacion;
   Cont_T:=Cont_T+1;
   Panel3.Visible:=True;
   Memo1.Text:='Obtuvimos este resultado aplicando el
   Teorema 13A al término A-B-CEF con ABCEF';
  end
  else
   if (Cont_T=5) then
   begin
    ContB:=ContB+1;
   Seis.Visible:=True;
   Ecuacion:='X=-A-BC-D-E-F+ADEF(-B-C+BC)+-ABCD-E-F';
   Seis.Text:=Ecuacion;
   Cont_T:=Cont_T+1;
   Panel3.Visible:=True;
   Memo1.Text:='Obtuvimos este resultado aplicando el
   Teorema 13A al término D(AEF(-B-C+BC)+-ABC-E-F)';
   end
   else
    if (Cont_T=6) then
    begin
     ContB:=ContB+1;
     Siete.Visible:=True;
     Ecuacion:='X=-A-E-F(-BC-D+BCD)+ADEF(-B-C+BC)';
     Siete.Font.Color:=clWhite;
     Siete.Text:=Ecuacion;
     Cont_T:=0;
     DeshBot(FEvaluarSA);
     Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el
     Teorema 13A al término -A-BC-D-E-F con -ABCD-E-F';
     Label29.Visible:=True;
     BRegresar.Visible:=True;
```

```
            BRegresar.Enabled:=True;
           end
           else
        begin
        Panel4.Visible:=True;
        Memo2.Text:='El Teorema 13A no es aplicable en este caso';
        NError:=NError+1;
        end
       else
if(Ejercicio='X=-(-A+-B+-C+-D+-E+-F)+ABCDEF+-A-BCDE-F+A-BCD-E-F')
then
        if (Cont_T=0) then
        begin
         ContB:=ContB+1;
         OculL;
         LEcuacion.Visible:=True;
         LEcuacion.Text:=Ejercicio;
         Uno.Visible:=True;
         Ecuacion:='X=-(-A+-B+-C+-D+-E+-F)+ABCDEF+-BCD-F(-AE+A-E)';
         Cont_T:=Cont_T+1;
         Uno.Text:=Ecuacion;
         Panel3.Visible:=True;
         Memo1.Text:='Obtuvimos este resultado aplicando el
         Teorema 13A al término -A-BCDE-F con A-BCD-E-F';
        end
        else
if  ((Cont_T=3)  and  (Ecuacion='X=ABCDEF+-A-BCDE-F+A-BCD-E-F'))
then
         begin
          ContB:=ContB+1;
          Cuatro.Visible:=True;
          Ecuacion:='X=ABCDEF+-BCD-F(-AE+A-E)';
          Cont_T:=Cont_T+1;
          Cuatro.Text:=Ecuacion;
          Panel3.Visible:=True;
          Memo1.Text:='Obtuvimos este resultado aplicando el
          Teorema 13A al término -A-BCDE-F con A-BCD-E-F';
         end
         else
if (Cont_T=4) and (Ecuacion='X=ABCDEF+-BCD-F(-AE+A-E)')) then
          begin
           ContB:=ContB+1;
           Cinco.Visible:=True;
           Ecuacion:='X=CD(ABEF+-B-F(-AE+A-E))';
           Cinco.Font.Color:=clWhite;
           Cinco.Text:=Ecuacion;
           Cont_T:=0;
           DeshBot(FEvaluarSA);
           Panel3.Visible:=True;
           Memo1.Text:='Obtuvimos este resultado aplicando el
           Teorema 13A al término ABCDEF con -BCD-F(-AE+A-E)';
```

```
          Label29.Visible:=True;
          BRegresar.Visible:=True;
          BRegresar.Enabled:=True;
         end
         else
if((Cont_T=1) and (Ecuacion='X=-(-A+-B+-C+-D+-E+-F)+ABCDEF+-BCD-
F(-AE+A-E)')) then
          begin
           ContB:=ContB+1;
           Dos.Visible:=True;
           Ecuacion:= 'X=-(-A+-B+-C+-D+-E+-F)+CD(ABEF+-B-F(-AE+A-E))';
           Cont_T:=Cont_T+1;
           Dos.Text:=Ecuacion;
           Panel3.Visible:=True;
           Memo1.Text:='Obtuvimos este resultado aplicando el
           Teorema 13A al término ABCDEF con -BCD-F(-AE+A-E)';
          end
          else
if ((Cont_T=4) and (Ecuacion='X=ABCDEF+CD(ABEF+-B-F(-AE+A-E))'))
then
          begin
           ContB:=ContB+1;
           Cinco.Visible:=True;
           Ecuacion:='X=CD(ABEF+ABEF+-B-F(-AE+A-E))';
           Cinco.Text:=Ecuacion;
           Cont_T:=Cont_T+1;
           Panel3.Visible:=True;
           Memo1.Text:='Obtuvimos este resultado aplicando el
           Teorema 13A al término ABCDEF con CD(ABEF+
           -B-F(-AE+A-E)';
      end
      else
       begin
       Panel4.Visible:=True;
       Memo2.Text:='El Teorema 13A no es aplicable en este caso';
            NError:=NError+1;
           end
        else
if(Ejercicio=  'X=  -(A+B+C+D+E+F)  +  -A-B-C-D-E-F  +  -A-
BCDEF+ABCDEF+AB-CDE-F') then
         if (Cont_T=0) then
         begin
          ContB:=ContB+1;
          OculL;
          LEcuacion.Visible:=True;
          LEcuacion.Text:=Ejercicio;
          Uno.Visible:=True;
          Ecuacion:='X=-(-A+-B+-C+-D+-E+-F)+-C-F(-A-B-D-
          +ABDE)+CDEF(-A-B+AB)';
          Cont_T:=Cont_T+1;
          Uno.Text:=Ecuacion;
```

```
          Panel3.Visible:=True;
          Memo1.Text:='Obtuvimos este resultado aplicando el
          Teorema 13A al término -A-B-C-D-E-F con -A-BCDEF y
          -A-BCDEF con ABCDEF';
         end
         else
if ((Cont_T=2) and (Ecuacion='X=-A-B-C-D-E-F+-A-BCDEF + ABCDEF +
AB-CDE-F')) then
          begin
           ContB:=ContB+1;
           Tres.Visible:=True;
           Ecuacion:='X=-C-F(-A-B-D-E+ABDE)+CDEF(-A-B+AB)';
           Tres.Font.Color:=clWhite;
           Tres.Text:=Ecuacion;
           Cont_T:=0;
           DeshBot(FEvaluarSA);
           Panel3.Visible:=True;
           Memo1.Text:='Obtuvimos este resultado aplicando el
           Teorema 13A a los términos -A-B-C-D-E-F con AB-CDE-F
           y -A-BCDEF con ABCDEF';
           Label29.Visible:=True;
           BRegresar.Visible:=True;
           BRegresar.Enabled:=True;
          end
          else
if     ((Cont_T=2)     and     (Ecuacion='X=-A-B-C-D-E-F+-C-F(-A-B-D-
E+ABDE)+CDEF(-A-B+AB)')) then
           begin
            ContB:=ContB+1;
            Tres.Visible:=True;
            Ecuacion:='X=-C-F(-A-B-D-E+-A-B-D-E+ABDE)+
            CDEF(-A-B+AB)';
            Tres.Text:=Ecuacion;
            Cont_T:=Cont_T+1;
            Panel3.Visible:=True;
            Memo1.Text:='Obtuvimos este resultado aplicando el
            Teorema 13A a los términos -A-B-C-D-E-F con
           -C-F(-A-B-D-E+ABDE)';
           end
           else
if    ((Cont_T=1)    and    (Ecuacion='X=-A-B-C-D-E-F+-A-B-C-D-E-F+-A-
BCDEF+ABCDEF+AB-CDE-F')) then
            begin
             ContB:=ContB+1;
             Dos.Visible:=True;
             Ecuacion:='X=-A-B-C-D-E-F+-C-F(-A-B-D-E+ABDE)+
             CDEF(-A-B+AB)';
             Dos.Text:=Ecuacion;
             Cont_T:=Cont_T+1;
             Panel3.Visible:=True;
             Memo1.Text:='Obtuvimos este resultado aplicando el
```

```
            Teorema 13A a los términos -A-B-C-D-E-F con AB-CDE-F y
            -A-BCDEF con AB-CDE-F';
            end
            else
if      ((Cont_T=2)      and      (Ecuacion='X=-A-B-C-D-E-F+-C-F(-A-B-D-

E+ABDE)+CDEF(-A-B+AB)')) then

             begin
              ContB:=ContB+1;
              Tres.Visible:=True;
              Ecuacion:='X=-C-F(-A-B-D-E+-A-B-D-E+ABDE)+CDEF(-A-B+AB)';
              Tres.Text:=Ecuacion;
              Cont_T:=Cont_T+1;
              Panel3.Visible:=True;
              Memo1.Text:='Obtuvimos este resultado aplicando el
              Teorema 13A a los términos -A-B-C-D-E-F con
              -C-F(-A-B-D-E+ABDE)';
             end
             else
             begin
              Panel4.Visible:=True;
             Memo2.Text:='El Teorema 13A no es aplicable en este caso';
              NError:=NError+1;
             end
          else
           if (Ejercicio='X=A-BC-D+-(A+B+C+-D)+-A-BCD') then
            if (Cont_T=0) then
            begin
             ContB:=ContB+1;
             OculL;
             LEcuacion.Visible:=True;
             LEcuacion.Text:=Ejercicio;
             Uno.Visible:=True;
             Ecuacion:='X=-BC(A-D+-AD)+-(A+B+C+-D)';
             Cont_T:=Cont_T+1;
             Uno.Text:=Ecuacion;
             Panel3.Visible:=True;
             Memo1.Text:='Obtuvimos este resultado aplicando el
             Teorema 13A a los términos A-BC-D con -A-BCD';
            end
            else
             if ((Cont_T=2) and (Ecuacion='X=A-BC-D+-A-B-CD+-A-BCD')) then
             begin
              ContB:=ContB+1;
              Tres.Visible:=True;
              Ecuacion:='X=A-BC-D+-A-BD(-C+C)';
              Cont_T:=Cont_T+1;
              Tres.Text:=Ecuacion;
              Panel3.Visible:=True;
              Memo1.Text:='Obtuvimos este resultado aplicando el
```

```
   Teorema 13A a el término -A-B-CD con -A-BCD';
  end
  else
   if ((Cont_T=4) and (Ecuacion='X=A-BC-D+-A-BD')) then
   begin
    ContB:=ContB+1;
    Cinco.Visible:=True;
    Ecuacion:='X=-B(AC-D+-AD)';
    Cinco.Font.Color:=clWhite;
    Cinco.Text:=Ecuacion;
    Cont_T:=0;
    DeshBot(FEvaluarSA);
    Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el
    Teorema 13A a el término A-BC-D con -A-BD';
    Label29.Visible:=True;
    BRegresar.Visible:=True;
    BRegresar.Enabled:=True;
   end
   else
if ((Cont_T=3) and (Ecuacion='X=-BC(A-D+-AD)+-A-B-CD')) then
    begin
     ContB:=ContB+1;
     Cuatro.Visible:=True;
     Ecuacion:='X=-B(C(A-D+-AD)+-A-CD)';
     Cont_T:=Cont_T+1;
     Cuatro.Text:=Ecuacion;
     Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el
     Teorema 13A a el término -BC(A-D+-AD) con -A-B-CD';
    end
    else
if ((Cont_T=4) and (Ecuacion='X=-B(C(A-D+-AD)+-A-CD)')) then
     begin
      ContB:=ContB+1;
      Cinco.Visible:=True;
      Ecuacion:='X=-B(AC-D+-ACD+-A-CD)';
      Cont_T:=Cont_T+1;
      Cinco.Text:=Ecuacion;
      Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el
     Teorema 13A a el término -B(C(A-D+-AD)+-A-CD)';
     end
     else
if ((Cont_T=5) and (Ecuacion='X=-B(AC-D+-ACD+-A-CD)')) then
      begin
       ContB:=ContB+1;
       Seis.Visible:=True;
       Ecuacion:='X=-B(AC-D+-AD(C+-C))';
       Cont_T:=Cont_T+1;
       Seis.Text:=Ecuacion;
```

```
        Panel3.Visible:=True;
        Memo1.Text:='Obtuvimos este resultado aplicando el
        Teorema 13A a el término -ACD con -A-CD';
       end
       else
if ((Cont_T=1) and (Ecuacion='X=A-BC-D+-A-B-C--D+-A-BCD')) then
        begin
         ContB:=ContB+1;
         Dos.Visible:=True;
         Ecuacion:='X=-BC(A-D+-AD)+-A-B-C--D';
         Cont_T:=Cont_T+1;
         Dos.Text:=Ecuacion;
         Panel3.Visible:=True;
        Memo1.Text:='Obtuvimos este resultado aplicando el
        Teorema 13A a el término A-BC-D con -A-BCD';
        end
        else
        begin
         Panel4.Visible:=True;
        Memo2.Text:='El Teorema 13A no es aplicable en este caso';
         NError:=NError+1;
        end
 else
  if (Ejercicio='X=-(A+B+-C)+-A-BCD+-AB-CD') then
   if (Cont_T=0) then
   begin
    ContB:=ContB+1;
    OculL;
    LEcuacion.Visible:=True;
    LEcuacion.Text:=Ejercicio;
    Uno.Visible:=True;
    Ecuacion:='X=-(A+B+-C)+-AD(-BC+B-C)';
    Cont_T:=Cont_T+1;
    Uno.Text:=Ecuacion;
    Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el
    Teorema 13A a los términos A-BC-D con -A-BCD';
   end
   else
 if ((Cont_T=2) and (Ecuacion='X=-A-BC+-A-BCD+-AB-CD')) then
    begin
     ContB:=ContB+1;
     Tres.Visible:=True;
     Ecuacion:='X=-A-BC(1+D)+-AB-CD';
     Cont_T:=Cont_T+1;
     Tres.Text:=Ecuacion;
     Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el
     Teorema 13A a el término -A-BC con -A-BCD';
    end
    else
```

```
          if ((Cont_T=4) and (Ecuacion='X=-A-BC+-AB-CD')) then
          begin
           ContB:=ContB+1;
           Cinco.Visible:=True;
           Ecuacion:='X=-A(-BC+B-CD)';
           Cinco.Font.Color:=clWhite;
           Cinco.Text:=Ecuacion;
           Cont_T:=0;
           DeshBot(FEvaluarSA);
           Panel3.Visible:=True;
           Memo1.Text:='Obtuvimos este resultado aplicando el
           Teorema 13A a el término -A-BC con -AB-CD';
           Label29.Visible:=True;
           BRegresar.Visible:=True;
           BRegresar.Enabled:=True;
          end
          else
      if ((Cont_T=1) and (Ecuacion='X=-A-B--C+-A-BCD+-AB-CD')) then
           begin
            ContB:=ContB+1;
            Dos.Visible:=True;
            Ecuacion:='X=-A-B--C+-AD(-BC+B-C)';
            Cont_T:=Cont_T+1;
            Dos.Text:=Ecuacion;
            Panel3.Visible:=True;
             Memo1.Text:='Obtuvimos este resultado aplicando el
             Teorema 13A a el término -A-BCD con -AB-CD';
           end
           else
        if ((Cont_T=3) and (Ecuacion='X=-A-BC+-AD(-BC+B-C)')) then
           begin
            ContB:=ContB+1;
            Cuatro.Visible:=True;
            Ecuacion:='X=-A(-BC+D(-BC+B-C))';
            Cont_T:=Cont_T+1;
            Cuatro.Text:=Ecuacion;
            Panel3.Visible:=True;
            Memo1.Text:='Obtuvimos este resultado aplicando el
            Teorema 13A a el término -A-BC con -AD(-BC+B-C)';
           end
           else
if ((Cont_T=4) and (Ecuacion='X=-A(-BC+D(-BC+B-C))')) then
            begin
             ContB:=ContB+1;
             Cinco.Visible:=True;
             Ecuacion:='X=-A(-BC+-BCD+B-CD)';
             Cont_T:=Cont_T+1;
             Cinco.Text:=Ecuacion;
             Panel3.Visible:=True;
             Memo1.Text:='Obtuvimos este resultado aplicando el
             Teorema 13A a el término D(-BC+B-C)';
```

```
          end
          else
if ((Cont_T=5) and (Ecuacion='X=-A(-BC+-BCD+B-CD)')) then
           begin
            ContB:=ContB+1;
            Seis.Visible:=True;
            Ecuacion:='X=-A(-BC(1+D)+B-CD)';
            Cont_T:=Cont_T+1;
            Seis.Text:=Ecuacion;
            Panel3.Visible:=True;
            Memo1.Text:='Obtuvimos este resultado aplicando el
            Teorema 13A a el término -BC con -BCD';
           end
           else
           begin
            Panel4.Visible:=True;
            Memo2.Text:='El Teorema 13A no es aplicable en este caso';
            NError:=NError+1;
           end
      else
if(Ejercicio='X=-(A+B+-C+-D+E)+-ABC-DE+A-B-C-DE+ABCDE') then
        if (Cont_T=0) then
        begin
         ContB:=ContB+1;
         OculL;
         LEcuacion.Visible:=True;
         LEcuacion.Text:=Ejercicio;
         Uno.Visible:=True;
         Ecuacion:='X=-(A+B+-C+-D+E)+BCE(-A-D+AD)+A-B-C-DE';
         Cont_T:=Cont_T+1;
         Uno.Text:=Ecuacion;
         Panel3.Visible:=True;
         Memo1.Text:='Obtuvimos este resultado aplicando el
         Teorema 13A a los términos A-B-C-DE con ABCDE';
        end
        else
 if ((Cont_T=3) and (Ecuacion='X=-A-BCD-E+BCE(-A-D+AD)+A-B-C-DE'))
then
         begin
          ContB:=ContB+1;
          Cuatro.Visible:=True;
          Ecuacion:='X=-B(-ACD-E+A-C-DE)+BCE(-A-D+AD)';
          Cuatro.Font.Color:=clWhite;
          Cuatro.Text:=Ecuacion;
          Cont_T:=0;
          DeshBot(FEvaluarSA);
          Panel3.Visible:=True;
          Memo1.Text:='Obtuvimos este resultado aplicando el
          Teorema 13A a los términos -A-BCD-E con A-B-C-DE';
          Label29.Visible:=True;
          BRegresar.Visible:=True;
```

```
            BRegresar.Enabled:=True;
           end
           else
if ((Cont_T=2) and (Ecuacion='X=-A-BCD-E+-ABC-DE+A-B-C-DE+ABCDE')) then
            begin
             ContB:=ContB+1;
             Tres.Visible:=True;
             Ecuacion:='X=-B(-ACD-E+A-C-DE)+BCE(-A-D+AD)';
             Tres.Font.Color:=clWhite;
             Tres.Text:=Ecuacion;
             Cont_T:=0;
             DeshBot(FEvaluarSA);
             Panel3.Visible:=True;
             Memo1.Text:='Obtuvimos este resultado aplicando el
             Teorema 13A a los términos -A-BCD-E con A-B-C-DE y
             -ABC-DE con ABCDE';
             Label29.Visible:=True;
             BRegresar.Visible:=True;
             BRegresar.Enabled:=True;
            end
            else
if    ((Cont_T=1)    and    (Ecuacion='X=-A-B--C--D-E+-ABC-DE+A-B-C-
DE+ABCDE')) then
            begin
             ContB:=ContB+1;
             Dos.Visible:=True;
             Ecuacion:='X=-A-B--C--D-E+BCE(-A-D+AD)+A-B-C-DE';
             Cont_T:=Cont_T+1;
             Dos.Text:=Ecuacion;
             Panel3.Visible:=True;
             Memo1.Text:='Obtuvimos este resultado aplicando el
             Teorema 13A a los términos -ABC-DE con ABCDE';
            end
       else
       begin
       Panel4.Visible:=True;
       Memo2.Text:='El Teorema 13A no es aplicable en este caso';
        NError:=NError+1;
            end
          else
           if (Ejercicio='X=(-A+B)(A+B+C)-D') then
            if (Cont_T=0) then
            begin
             ContB:=ContB+1;
             OculL;
             LEcuacion.Visible:=True;
             LEcuacion.Text:=Ejercicio;
             Uno.Visible:=True;
             Ecuacion:='X=(-A+B)(A-D+B-D+C-D)';
             Cont_T:=Cont_T+1;
             Uno.Text:=Ecuacion;
```

```
            Panel3.Visible:=True;
            Memo1.Text:='Obtuvimos este resultado aplicando el
            Teorema 13A a los términos (A+B+C) con -D';
           end
           else
if ((Cont_T=4) and (Ecuacion='X=-AB-D+-AC-D+AB-D+B-D+BC-D')) then
            begin
             ContB:=ContB+1;
             Cinco.Visible:=True;
             Ecuacion:='X=B-D(-A+A)+C-D(-A+B)+B-D';
             Cont_T:=Cont_T+1;
             Cinco.Text:=Ecuacion;
             Panel3.Visible:=True;
            Memo1.Text:='Obtuvimos este resultado aplicando el
            Teorema 13A a los términos -AB-D con AB-D y
            -AC-D con BC-D';
            end
            else
if ((Cont_T=3) and (Ecuacion='X=(-AB+-AC+AB+B+BC)-D')) then
             begin
              ContB:=ContB+1;
              Cuatro.Visible:=True;
              Ecuacion:='X=-AB-D+-AC-D+AB-D+B-D+BC-D';
              Cont_T:=Cont_T+1;
              Cuatro.Text:=Ecuacion;
              Panel3.Visible:=True;
              Memo1.Text:='Obtuvimos este resultado aplicando el
              Teorema 13A a el término (-AB+-AC+AB+B+BC) con
              –D';
             end
             else
        begin
        Panel4.Visible:=True;
        Memo2.Text:='El Teorema 13A no es aplicable en este caso';
              NError:=NError+1;
             end
           else
            if (Ejercicio='X=(B+-C)(-B+C)+-(-A+B+-C)') then
             if ((Cont_T=4) and (Ecuacion='X=BC+-B-C+A-BC')) then
             begin
              ContB:=ContB+1;
              Cinco.Visible:=True;
              Ecuacion:='X=C(B+A-B)+-B-C';
              Cont_T:=Cont_T+1;
              Cinco.Text:=Ecuacion;
              Panel3.Visible:=True;
              Memo1.Text:='Obtuvimos este resultado aplicando el
              Teorema 13A a el término BC con A-BC';
             end
             else
if ((Cont_T=3) and (Ecuacion='X=-BB+BC+-B-C+-CC+A-BC')) then
```

```
            begin
             ContB:=ContB+1;
             Cuatro.Visible:=True;
             Ecuacion:='X=-BB+C(B+A-B)+-B-C+-CC';
             Cont_T:=Cont_T+1;
             Cuatro.Text:=Ecuacion;
             Panel3.Visible:=True;
             Memo1.Text:='Obtuvimos este resultado aplicando el
             Teorema 13A a el término BC con A-BC';
            end
            else
            begin
             Panel4.Visible:=True;
             Memo2.Text:='El Teorema 13A no es aplicable en este
             caso';
             NError:=NError+1;
            end
           else
      if(Ejercicio='X=-(A+B+-C+-D+E)+-A-B-C-E+ABCD-E+ABC+-A-BCD-E') then
            if (Cont_T=0) then
            begin
             ContB:=ContB+1;
             OculL;
             LEcuacion.Visible:=True;
             LEcuacion.Text:=Ejercicio;
             Uno.Visible:=True;
             Ecuacion:='X=-(A+B+-C+-D+E)+-A-B-E(-C+CD)+ABC(D-E+1)';
             Cont_T:=Cont_T+1;
             Uno.Text:=Ecuacion;
             Panel3.Visible:=True;
             Memo1.Text:='Obtuvimos este resultado aplicando el
             Teorema 13A a los términos -A-B-C-E con -A-BCD-E y
             ABCD-E con ABC';
             end
             else
         if  ((Cont_T=2)  and  (Ecuacion='X=-A-BCD-E+-A-B-C-E+ABCD-

E+ABC

        +-A-BCD-E')) then

              begin
               ContB:=ContB+1;
               Tres.Visible:=True;
               Ecuacion:='X=-A-B-E(CD+-C)+ABC(D-E+1)+-A-BCD-E';
               Cont_T:=Cont_T+1;
               Tres.Text:=Ecuacion;
               Panel3.Visible:=True;
               Memo1.Text:='Obtuvimos este resultado aplicando el
               Teorema 13A a los términos -A-BCD-E con -A-B-C-E
               y ABCD-E con ABC';
```

```
          end
          else
if ((Cont_T=5) and (Ecuacion='X=-A-B-E(-C+D)+ABC+-A-BCD-E')) then
           begin
            ContB:=ContB+1;
            Seis.Visible:=True;
            Ecuacion:='X=-A-B-E(-C+D+CD)+ABC';
            Cont_T:=Cont_T+1;
            Seis.Text:=Ecuacion;
            Panel3.Visible:=True;
            Memo1.Text:='Obtuvimos este resultado aplicando
            el Teorema 13A a el término -A-B-E(-C+D) con
            -A-BCD-E';
           end
           else
 if ((Cont_T=6) and (Ecuacion='X=-A-B-E(-C+D+CD)+ABC'))
 then
           begin
            ContB:=ContB+1;
            Siete.Visible:=True;
            Ecuacion:='X=-A-B-E(-C+D(1+C))+ABC';
            Cont_T:=Cont_T+1;
            Siete.Text:=Ecuacion;
            Panel3.Visible:=True;
            Memo1.Text:='Obtuvimos este resultado aplicando
            el Teorema 13A a el término D con CD';
           end
           else
 if ((Cont_T=3) and (Ecuacion='X=-A-BCD-E+-A-B-C-E+ABCD-E+ABC'))
then
            begin
             ContB:=ContB+1;
             Cuatro.Visible:=True;
             Ecuacion:='X=-A-B-E(CD+-C)+ABC(D-E+1)';
             Cont_T:=Cont_T+1;
             Cuatro.Text:=Ecuacion;
             Panel3.Visible:=True;
             Memo1.Text:='Obtuvimos este resultado aplicando
             el Teorema 13A a los términos -A-BCD-E con
             -A-B-C-E y ABCD-E con ABC';
            end
            else
 if ((Cont_T=5) and (Ecuacion='X=-A-BCD-E+-A-B-E(-C+D)+ABC')) then
             begin
              ContB:=ContB+1;
              Seis.Visible:=True;
              Ecuacion:='X=-A-B-E(CD+-C+D)+ABC';
              Cont_T:=Cont_T+1;
              Seis.Text:=Ecuacion;
              Panel3.Visible:=True;
              Memo1.Text:='Obtuvimos este resultado aplicando
```

```
                el Teorema 13A a los términos -A-BCD-E con
                -A-B-E(-C+D)';
               end
               else
 if ((Cont_T=6) and (Ecuacion='X=-A-B-E(CD+-C+D)+ABC'))
 then
                begin
                 ContB:=ContB+1;
                 Siete.Visible:=True;
                 Ecuacion:='X=-A-B-E(-C+D(1+C))+ABC';
                 Cont_T:=Cont_T+1;
                 Siete.Text:=Ecuacion;
                 Panel3.Visible:=True;
                 Memo1.Text:='Obtuvimos este resultado
                 aplicando el Teorema 13A a el término D con CD';
                end
                else
 if ((Cont_T=1) and (Ecuacion='X=-A-B--C--D-E+-A-B-C-E+
ABCD-E+ABC+-A-BCD-E')) then
               begin
                 ContB:=ContB+1;
                 Dos.Visible:=True;
                 Ecuacion:='X=-A-B--C--D-E+-A-B-E(-C+CD)+ABC(D-E+1)';
                 Cont_T:=Cont_T+1;
                 Dos.Text:=Ecuacion;
                 Panel3.Visible:=True;
                 Memo1.Text:='Obtuvimos este resultado aplicando
                 el Teorema 13A a los términos -A-B-C-E con
                 -A-BCD-E y ABCD-E con ABC';
               end
                  else
                  begin
                  Panel4.Visible:=True;
                  Memo2.Text:='El Teorema 13A no es aplicable en
                 este caso';
                  NError:=NError+1;
                  end
          else
          begin
           if (Cont_T=0) then
           MostL;
           Panel4.Visible:=True;
           Memo2.Text:='El Teorema 13A no es aplicable en este caso';
           NError:=NError+1;
          end;
end;
```

PROCEDIMIENTO EVALUAR_SIMPLIFICA16. Permite aplicar el teorema - -X=X del álgebra booleana en la evaluación sobre simplificación algebraica, una vez lo ha seleccionado el usuario.

```
procedure TFEvaluarSA.B16Click(Sender: TObject);
begin
 if (Ejercicio='X=-A-B-C-D-E+-(A+B+C+D+E)+ABCDE+AB-CD-E') then
  if (Cont_T=0) then
  begin
   ContB:=ContB+1;
   OculL;
   LEcuacion.Visible:=True;
   LEcuacion.Text:=Ejercicio;
   Uno.Visible:=True;
   Ecuacion:='X=-A-B-C-D-E+-A-B-C-D-E+ABCDE+AB-CD-E';
   Cont_T:=Cont_T+1;
   Uno.Text:=Ecuacion;
   Panel3.Visible:=True;
   Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al término
-(A+B+C+D+E)';
  end
  else
   if ((Cont_T=1) and (Ecuacion='X=-A-B-C-D-E+-
(A+B+C+D+E)+ABD(CE+-C-E)')) then
   begin
    ContB:=ContB+1;
    Dos.Visible:=True;
    Ecuacion:='X=-A-B-C-D-E+-A-B-C-D-E+ABD(CE+-C-E)';
    Cont_T:=Cont_T+1;
    Dos.Text:=Ecuacion;
    Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(A+B+C+D+E)';
   end
   else
   begin
    Panel4.Visible:=True;
    Memo2.Text:='La Ley 16 no es aplicable en este caso';
    NError:=NError+1;
   end
 else
  if (Ejercicio='X=-(A+B+-C+D+E+F)+A-B-CDEF+-ABCD-E-F+-(-A+-B+-
C+-D+-E+-F)') then
   if (Cont_T=0) then
   begin
    OculL;
```

```
   ContB:=ContB+1;
   LEcuacion.Visible:=True;
   LEcuacion.Text:=Ejercicio;
   Uno.Visible:=True;
   Ecuacion:='X=-A-B--C-D-E-F+A-B-CDEF+-ABCD-E-F+--A--B--C--D--E-
-F';
   Cont_T:=Cont_T+1;
   Uno.Text:=Ecuacion;
   Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 a los
términos -(A+B+-C+D+E+F) y -(-A+-B+-C+-D+-E+-F)';
  end
  else
   if ((Cont_T=1) and (Ecuacion='X=-(A+B+-C+D+E+F)+D(A-B-CEF+-
ABC-E-F)+-(-A+-B+-C+-D+-E+-F)')) then
   begin
    ContB:=ContB+1;
    Dos.Visible:=True;
    Ecuacion:='X=-A-B--C-D-E-F+D(A-B-CEF+-ABC-E-F)+--A--B--C--D--
E--F';
    Cont_T:=Cont_T+1;
    Dos.Text:=Ecuacion;
    Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(-A+-B+-C+-D+-E+-F)';
   end
   else
   begin
    Panel4.Visible:=True;
    Memo2.Text:='La Ley 16 no es aplicable en este caso';
    NError:=NError+1;
   end
 else
  if (Ejercicio='X=-(-A+-B+-C+-D+-E+-F)+ABCDEF+-A-BCDE-F+A-BCD-
E-F') then
   if (Cont_T=0) then
   begin
    ContB:=ContB+1;
    OculL;
    LEcuacion.Visible:=True;
    LEcuacion.Text:=Ejercicio;
    Uno.Visible:=True;
    Ecuacion:='X=--A--B--C--D--E--F+ABCDEF+-A-BCDE-F+A-BCD-E-F';
    Cont_T:=Cont_T+1;
    Uno.Text:=Ecuacion;
    Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(-A+-B+-C+-D+-E+-F)';
   end
   else
```

```
    if ((Cont_T=1) and (Ecuacion='X=-(-A+-B+-C+-D+-E+-F)+ABCDEF+-
BCD-F(-AE+A-E)')) then
    begin
     ContB:=ContB+1;
     Dos.Visible:=True;
     Ecuacion:='X=--A--B--C--D--E--F+ABCDEF+-BCD-F(-AE+A-E)';
     Cont_T:=Cont_T+1;
     Dos.Text:=Ecuacion;
     Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(-A+-B+-C+-D+-E+-F)';
    end
    else
     if ((Cont_T=2) and (Ecuacion='X=-(-A+-B+-C+-D+-E+-F)+CD(ABEF+-
B-F(-AE+A-E))')) then
     begin
      ContB:=ContB+1;
      Tres.Visible:=True;
      Ecuacion:='X=--A--B--C--D--E--F+CD(ABEF+-B-F(-AE+A-E))';
      Cont_T:=Cont_T+1;
      Tres.Text:=Ecuacion;
      Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(-A+-B+-C+-D+-E+-F)';
     end
     else
     begin
      Panel4.Visible:=True;
      Memo2.Text:='La Ley 16 no es aplicable en este caso';
      NError:=NError+1;
     end
  else
   if                  (Ejercicio='X=-(A+B+C+D+E+F)+-A-B-C-D-E-F+-A-
BCDEF+ABCDEF+AB-CDE-F') then
    if (Cont_T=0) then
    begin
     ContB:=ContB+1;
     OculL;
     LEcuacion.Visible:=True;
     LEcuacion.Text:=Ejercicio;
     Uno.Visible:=True;
     Ecuacion:='X=-A-B-C-D-E-F+-A-B-C-D-E-F+-A-BCDEF+ABCDEF+AB-
CDE-F';
     Cont_T:=Cont_T+1;
     Uno.Text:=Ecuacion;
     Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(A+B+C+D+E+F)';
    end
    else
```

```
     if ((Cont_T=1) and (Ecuacion='X=-(-A+-B+-C+-D+-E+-F)+-C-F(-A-B-
D-E+ABDE)+CDEF(-A-B+AB)')) then
     begin
      ContB:=ContB+1;
      Dos.Visible:=True;
      Ecuacion:='X=-A-B-C-D-E-F+-C-F(-A-B-D-E+ABDE)+CDEF(-A-
B+AB)';
      Cont_T:=Cont_T+1;
      Dos.Text:=Ecuacion;
      Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(-A+-B+-C+-D+-E+-F)';
     end
     else
     begin
      Panel4.Visible:=True;
      Memo2.Text:='La Ley 16 no es aplicable en este caso';
      NError:=NError+1;
     end
   else
    if (Ejercicio='X=A-BC-D+-(A+B+C+-D)+-A-BCD') then
     if (Cont_T=0) then
     begin
      ContB:=ContB+1;
      OculL;
      LEcuacion.Visible:=True;
      LEcuacion.Text:=Ejercicio;
      Uno.Visible:=True;
      Ecuacion:='X=A-BC-D+-A-B-C--D+-A-BCD';
      Cont_T:=Cont_T+1;
      Uno.Text:=Ecuacion;
      Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(A+B+C+-D)';
     end
     else
      if ((Cont_T=1) and (Ecuacion='X=-BC(A-D+-AD)+-(A+B+C+-D)'))
then
      begin
       ContB:=ContB+1;
       Dos.Visible:=True;
       Ecuacion:='X=-BC(A-D+-AD)+-A-B-C--D';
       Cont_T:=Cont_T+1;
       Dos.Text:=Ecuacion;
       Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(A+B+C+-D)';
      end
      else
      begin
       Panel4.Visible:=True;
```

```
      Memo2.Text:='La Ley 16 no es aplicable en este caso';
      NError:=NError+1;
     end
    else
     if (Ejercicio='X=-(A+B+-C)+-A-BCD+-AB-CD') then
      if (Cont_T=0) then
      begin
       ContB:=ContB+1;
       OculL;
       LEcuacion.Visible:=True;
       LEcuacion.Text:=Ejercicio;
       Uno.Visible:=True;
       Ecuacion:='X=-A-B--C+-A-BCD+-AB-CD';
       Cont_T:=Cont_T+1;
       Uno.Text:=Ecuacion;
       Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(A+B+-C)';
      end
      else
       if ((Cont_T=1) and (Ecuacion='X=-(A+B+-C)+-AD(-BC+B-C)')) then
       begin
        ContB:=ContB+1;
        Dos.Visible:=True;
        Ecuacion:='X=-A-B--C+-AD(-BC+B-C)';
        Cont_T:=Cont_T+1;
        Dos.Text:=Ecuacion;
        Panel3.Visible:=True;
        Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(A+B+-C)';
       end
       else
       begin
        Panel4.Visible:=True;
        Memo2.Text:='La Ley 16 no es aplicable en este caso';
        NError:=NError+1;
       end
      else
      if      (Ejercicio='X=-(A+B+-C+-D+E)+-ABC-DE+A-B-C-DE+ABCDE')
then
       if (Cont_T=0) then
       begin
        ContB:=ContB+1;
        OculL;
        LEcuacion.Visible:=True;
        LEcuacion.Text:=Ejercicio;
        Uno.Visible:=True;
        Ecuacion:='X=-A-B--C--D-E+-ABC-DE+A-B-C-DE+ABCDE';
        Cont_T:=Cont_T+1;
        Uno.Text:=Ecuacion;
        Panel3.Visible:=True;
```

```
      Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(A+B+-C+-D+E)';
     end
     else
      if ((Cont_T=1) and (Ecuacion='X=-(A+B+-C+-D+E)+BCE(-A-
D+AD)+A-B-C-DE')) then
      begin
       ContB:=ContB+1;
       Dos.Visible:=True;
       Ecuacion:='X=-A-B--C--D-E+BCE(-A-D+AD)+A-B-C-DE';
       Cont_T:=Cont_T+1;
       Dos.Text:=Ecuacion;
       Panel3.Visible:=True;
         Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(A+B+-C+-D+E)';
      end
      else
      begin
       Panel4.Visible:=True;
       Memo2.Text:='La Ley 16 no es aplicable en este caso';
       NError:=NError+1;
      end
     else
     if (Ejercicio='X=(B+-C)(-B+C)+-(-A+B+-C)') then
      if (Cont_T=0) then
      begin
       ContB:=ContB+1;
       OculL;
       LEcuacion.Visible:=True;
       LEcuacion.Text:=Ejercicio;
       Uno.Visible:=True;
       Ecuacion:='X=(B+-C)(-B+C)+--A-B--C';
       Cont_T:=Cont_T+1;
       Uno.Text:=Ecuacion;
       Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(-A+B+-C)';
      end
      else
       if ((Cont_T=2) and (Ecuacion='X=BC+-B-C+-(-A+B+-C)')) then
       begin
        ContB:=ContB+1;
        Tres.Visible:=True;
        Ecuacion:='X=BC+-B-C+--A-B--C';
        Cont_T:=Cont_T+1;
        Tres.Text:=Ecuacion;
        Panel3.Visible:=True;
          Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(-A+B+-C)';
       end
       else
```

```
        if ((Cont_T=1) and (Ecuacion='X=-BB+BC+-B-C+-CC+-(-A+B+-
C)')) then
        begin
         ContB:=ContB+1;
         Dos.Visible:=True;
         Ecuacion:='X=-BB+BC+-B-C+-CC+--A-B--C';
         Cont_T:=Cont_T+1;
         Dos.Text:=Ecuacion;
         Panel3.Visible:=True;
          Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(-A+B+-C)';
        end
        else
        begin
         Panel4.Visible:=True;
         Memo2.Text:='La Ley 16 no es aplicable en este caso';
         NError:=NError+1;
        end
      else
       if      (Ejercicio='X=-(A+B+-C+-D+E)+-A-B-C-E+ABCD-E+ABC+-A-
BCD-E') then
        if (Cont_T=0) then
        begin
         ContB:=ContB+1;
         OculL;
         LEcuacion.Visible:=True;
         LEcuacion.Text:=Ejercicio;
         Uno.Visible:=True;
         Ecuacion:='X=-A-B--C--D-E+-A-B-C-E+ABCD-E+ABC+-A-BCD-E';
         Cont_T:=Cont_T+1;
         Uno.Text:=Ecuacion;
         Panel3.Visible:=True;
          Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(A+B+-C+-D+E)';
        end
        else
         if ((Cont_T=3) and (Ecuacion='X=-(A+B+-C+-D+E)+-A-B-E(-
C+D)+ABC')) then
        begin
         ContB:=ContB+1;
         Cuatro.Visible:=True;
         Ecuacion:='X=-A-B--C--D-E+-A-B-E(-C+D)+ABC';
         Cont_T:=Cont_T+1;
         Cuatro.Text:=Ecuacion;
         Panel3.Visible:=True;
          Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(A+B+-C+-D+E)';
        end
        else
         if ((Cont_T=1) and (Ecuacion='X=-(A+B+-C+-D+E)+-A-B-E(-
C+CD)+ABC(D-E+1)')) then
```

```
          begin
           ContB:=ContB+1;
           Dos.Visible:=True;
           Ecuacion:='X=-A-B--C--D-E+-A-B-E(-C+CD)+ABC(D-E+1)';
           Cont_T:=Cont_T+1;
           Dos.Text:=Ecuacion;
           Panel3.Visible:=True;
            Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(A+B+-C+-D+E)';
          end
          else
           if ((Cont_T=2) and (Ecuacion='X=-(A+B+-C+-D+E)+-A-B-E(-
C+CD)+ABC')) then
           begin
            ContB:=ContB+1;
            Tres.Visible:=True;
            Ecuacion:='X=-A-B--C--D-E+-A-B-E(-C+CD)+ABC';
            Cont_T:=Cont_T+1;
            Tres.Text:=Ecuacion;
            Panel3.Visible:=True;
            Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16 al
término -(A+B+-C+-D+E)';
           end
           else
            if ((Cont_T=2) and (Ecuacion='X=-(A+B+-C+-D+E)+-A-B-E(-
C+D)+ABC(D-E+1)')) then
            begin
             ContB:=ContB+1;
             Tres.Visible:=True;
             Ecuacion:='X=-A-B--C--D-E+-A-B-E(-C+D)+ABC(D-E+1)';
             Cont_T:=Cont_T+1;
             Tres.Text:=Ecuacion;
             Panel3.Visible:=True;
               Memo1.Text:='Obtuvimos este resultado aplicando la Ley 16
al término -(A+B+-C+-D+E)';
            end
            else
            begin
             Panel4.Visible:=True;
             Memo2.Text:='La Ley 16 no es aplicable en este caso';
             NError:=NError+1;
            end
       else
       begin
        if (Cont_T=0) then
         MostL;
        Panel4.Visible:=True;
        Memo2.Text:='La Ley 16 no es aplicable en este caso';
        NError:=NError+1;
       end;
end;
```

PROCEDIMIENTO EVALUAR_SIMPLIFICA8. Permite aplicar el teorema X+-X=1 del álgebra booleana en la evaluación sobre simplificación algebraica, una vez lo ha seleccionado el usuario.

```
procedure TFEvaluarSA.B8Click(Sender: TObject);
begin
 if (Ejercicio='S=-P-Q-R+-P-QR+-PQ-R+-PQR+PQR') then
  if ((Cont_T=1) and (Ecuacion='S=-P-Q(-R+R)+-PQ(-R+R)+PQR')) then
  begin
   ContB:=ContB+1;
   Dos.Visible:=True;
   Ecuacion:='S=-P-Q+-PQ+PQR';
   Cont_T:=Cont_T+1;
   Dos.Text:=Ecuacion;
   Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 8 a los
términos -P-Q(-R+R) Y -PQ(-R+R)';
  end
  else
   if ((Cont_T=3) and (Ecuacion='S=-P(-Q+Q)+PQR')) then
   begin
    ContB:=ContB+1;
    Cuatro.Visible:=True;
    Ecuacion:='S=-P+PQR';
    Cuatro.Font.Color:=clWhite;
    Cuatro.Text:=Ecuacion;
    Cont_T:=0;
    DeshBot(FEvaluarSA);
    Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 8 al
término -P(-Q+Q)';
    Label29.Visible:=True;
    BRegresar.Visible:=True;
    BRegresar.Enabled:=True;
   end
   else
   begin
    Panel4.Visible:=True;
    Memo2.Text:='El Teorema 8 no es aplicable en este caso';
    NError:=NError+1;
   end
 else
```

```
  if (Ejercicio='X=ABC+A-BC+-A') then
   if (Cont_T=1) then
   begin
    ContB:=ContB+1;
    Dos.Visible:=True;
    Ecuacion:='X=AC+-A';
    Cont_T:=Cont_T+1;
    Dos.Text:=Ecuacion;
    Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 8 a los
términos B y -B';
   end
   else
   begin
    Panel4.Visible:=True;
    Memo2.Text:='El Teorema 8 no es aplicable en este caso';
    NError:=NError+1;
   end
  else
   if (Ejercicio='X=-A-B-C-D+-A-BC-D+-AB-C-D') then
    if ((Cont_T=1) and (Ecuacion='X=-A-C-D(-B+B)+-A-BC-D')) then
    begin
     ContB:=ContB+1;
     Dos.Visible:=True;
     Ecuacion:='X=-A-C-D+-A-BC-D';
     Cont_T:=Cont_T+1;
     Dos.Text:=Ecuacion;
     Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 8 al
término -A-C-D(-B+B)';
    end
    else
     if ((Cont_T=2) and (Ecuacion='X=-A-D(-C(-B+B)+-BC)')) then
     begin
      ContB:=ContB+1;
      Tres.Visible:=True;
      Ecuacion:='X=-A-D(-C+-BC)';
      Cont_T:=Cont_T+1;
      Tres.Text:=Ecuacion;
      Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 8 al
término -C(-B+B)';
     end
     else
     begin
      Panel4.Visible:=True;
      Memo2.Text:='El Teorema 8 no es aplicable en este caso';
      NError:=NError+1;
     end
    else
     if (Ejercicio='X=ABCDE+-A-BCDE+ABCD-E+-A-B-C-DE') then
```

```
    if (Cont_T=1) then
    begin
     ContB:=ContB+1;
     Dos.Visible:=True;
     Ecuacion:='X=ABCD+-A-BE(CD+-C-D)';
     Dos.Font.Color:=clWhite;
     Dos.Text:=Ecuacion;
     Cont_T:=0;
     DeshBot(FEvaluarSA);
     Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 8 al
término ABCD(E+-E)';
     Label29.Visible:=True;
     BRegresar.Visible:=True;
     BRegresar.Enabled:=True;
    end
    else
    begin
     Panel4.Visible:=True;
     Memo2.Text:='El Teorema 8 no es aplicable en este caso';
     NError:=NError+1;
    end
   else
    if (Ejercicio='X=-A-BCD-EF+ABCDEF+AB-CD-EF+A-B-CD-EF+-A-B-C-
D-E-F') then
     if (Cont_T=1) then
     begin
      ContB:=ContB+1;
      Dos.Visible:=True;
      Ecuacion:='X=-A-B-E(CDF+-C-D-F)+ABCDEF+A-CD-EF';
      Cont_T:=Cont_T+1;
      Dos.Text:=Ecuacion;
      Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 8 a
el término A-CD-EF(B+-B)';
     end
     else
     begin
      Panel4.Visible:=True;
      Memo2.Text:='El Teorema 8 no es aplicable en este caso';
      NError:=NError+1;
     end
    else
     if (Ejercicio='X=A-BC-D+-(A+B+C+-D)+-A-BCD') then
      if ((Cont_T=3) and (Ecuacion='X=A-BC-D+-A-BD(-C+C)')) then
      begin
       ContB:=ContB+1;
       Cuatro.Visible:=True;
       Ecuacion:='X=A-BC-D+-A-BD';
       Cont_T:=Cont_T+1;
       Cuatro.Text:=Ecuacion;
```

```
      Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 8 a
el término -A-BD(-C+C)';
      end
      else
       if ((Cont_T=6) and (Ecuacion='X=-B(AC-D+-AD(C+-C))')) then
       begin
        ContB:=ContB+1;
        Siete.Visible:=True;
        Ecuacion:='X=-B(AC-D+-AD)';
        Siete.Font.Color:=clWhite;
        Siete.Text:=Ecuacion;
        Cont_T:=0;
        DeshBot(FEvaluarSA);
        Panel3.Visible:=True;
         Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 8 a
el término -AD(-C+C)';
        Label29.Visible:=True;
        BRegresar.Visible:=True;
        BRegresar.Enabled:=True;
       end
       else
       begin
        Panel4.Visible:=True;
        Memo2.Text:='El Teorema 8 no es aplicable en este caso';
        NError:=NError+1;
       end
     else
      if (Ejercicio='X=(-A+B)(A+B+C)-D') then
       if ((Cont_T=5) and (Ecuacion='X=B-D(-A+A)+C-D(-A+B)+B-D')) then
       begin
        ContB:=ContB+1;
        Seis.Visible:=True;
        Ecuacion:='X=B-D+C-D(-A+B)+B-D';
        Cont_T:=Cont_T+1;
        Seis.Text:=Ecuacion;
        Panel3.Visible:=True;
         Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 8 a
el término B-D(-A+A)';
       end
       else
       begin
        Panel4.Visible:=True;
        Memo2.Text:='El Teorema 8 no es aplicable en este caso';
        NError:=NError+1;
       end
      else
      begin
       if (Cont_T=0) then
        MostL;
       Panel4.Visible:=True;
```

```
      Memo2.Text:='El Teorema 8 no es aplicable en este caso';
      NError:=NError+1;
     end;
end;
```

PROCEDIMIENTO EVALUAR_SIMPLIFICA15. Permite aplicar el teorema X+-XY=X+Y del álgebra booleana en la evaluación sobre simplificación algebraica, una vez lo ha seleccionado el usuario.

```
procedure TFEvaluarSA.B15Click(Sender: TObject);
begin
 if (Ejercicio='X=ABC+-AC') then
  if ((Cont_T=1) and (Ecuacion='X=C(AB+-A)')) then
   begin
    ContB:=ContB+1;
    Dos.Visible:=True;
    Ecuacion:='X=C(-A+B)';
    Dos.Font.Color:=clWhite;
    Dos.Text:=Ecuacion;
    Cont_T:=0;
    DeshBot(FEvaluarSA);
    Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 15 al
término (AB+-A)';
    Label29.Visible:=True;
    BRegresar.Visible:=True;
    BRegresar.Enabled:=True;
   end
   else
   begin
    Panel4.Visible:=True;
    Memo2.Text:='El Teorema 15 no es aplicable en este caso';
    NError:=NError+1;
   end
 else
  if (Ejercicio='X=ABC+A-BC+-A') then
   if (Cont_T=2) then
   begin
    ContB:=ContB+1;
    Tres.Visible:=True;
    Ecuacion:='X=-A+C';
    Tres.Font.Color:=clWhite;
    Tres.Text:=Ecuacion;
    Cont_T:=0;
```

```
    DeshBot(FEvaluarSA);
    Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 15 al
término AC+-A)';
    Label29.Visible:=True;
    BRegresar.Visible:=True;
    BRegresar.Enabled:=True;
   end
   else
   begin
    Panel4.Visible:=True;
    Memo2.Text:='El Teorema 15 no es aplicable en este caso';
    NError:=NError+1;
   end
  else
   if (Ejercicio='X=-A-B-C-D+-A-BC-D+-AB-C-D') then
    if (Cont_T=3) then
    begin
     ContB:=ContB+1;
     Cuatro.Visible:=True;
     Ecuacion:='X=-A-D(-B+-C)';
     Cuatro.Font.Color:=clWhite;
     Cuatro.Text:=Ecuacion;
     Cont_T:=0;
     DeshBot(FEvaluarSA);
     Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 15 al
término -A-D(-C+-BC)';
     Label29.Visible:=True;
     BRegresar.Visible:=True;
     BRegresar.Enabled:=True;
    end
    else
    begin
     Panel4.Visible:=True;
     Memo2.Text:='El Teorema 15 no es aplicable en este caso';
     NError:=NError+1;
    end
   else
    if (Ejercicio='X=(B+-C)(-B+C)+-(-A+B+-C)') then
     if ((Cont_T=5) and (Ecuacion='X=C(B+A-B)+-B-C')) then
     begin
      ContB:=ContB+1;
      Seis.Visible:=True;
      Ecuacion:='X=C(A+B)+-B-C';
      Seis.Font.Color:=clWhite;
      Seis.Text:=Ecuacion;
      Cont_T:=0;
      DeshBot(FEvaluarSA);
      Panel3.Visible:=True;
```

```
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 15 a
el término C(B+A-B)';
     Label29.Visible:=True;
     BRegresar.Visible:=True;
     BRegresar.Enabled:=True;
    end
    else
     if ((Cont_T=4) and (Ecuacion='X=-BB+C(B+A-B)+-B-C+-CC')) then
     begin
      ContB:=ContB+1;
      Cinco.Visible:=True;
      Ecuacion:='X=-BB+C(A+B)+-B-C+-CC';
      Cont_T:=Cont_T+1;
      Cinco.Text:=Ecuacion;
      Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 15 a
el término C(B+A-B)';
     end
     else
     begin
      Panel4.Visible:=True;
      Memo2.Text:='El Teorema 15 no es aplicable en este caso';
      NError:=NError+1;
     end
   else
    if    (Ejercicio='X=-(A+B+-C+-D+E)+-A-B-C-E+ABCD-E+ABC+-A-BCD-
E') then
     if ((Cont_T=4) and (Ecuacion='X=-A-B-E(CD+-C)+ABC+-A-BCD-E'))
then
     begin
      ContB:=ContB+1;
      Cinco.Visible:=True;
      Ecuacion:='X=-A-B-E(-C+D)+ABC+-A-BCD-E';
      Cont_T:=Cont_T+1;
      Cinco.Text:=Ecuacion;
      Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 15 a
el término -A-B-E(CD+-C)';
     end
     else
      if   ((Cont_T=4)   and   (Ecuacion='X=-A-B-E(CD+-C)+ABC(D-E+1)'))
then
      begin
       ContB:=ContB+1;
       Cinco.Visible:=True;
       Ecuacion:='X=-A-B-E(-C+D)+ABC(D-E+1)';
       Cont_T:=Cont_T+1;
       Cinco.Text:=Ecuacion;
       Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 15 a
el término -A-B-E(CD+-C)';
```

```
      end
      else
       if ((Cont_T=5) and (Ecuacion='X=-A-B-E(CD+-C)+ABC')) then
       begin
        ContB:=ContB+1;
        Seis.Visible:=True;
        Ecuacion:='X=-A-B-E(-C+D)+ABC';
        Seis.Font.Color:=clWhite;
        Seis.Text:=Ecuacion;
        Cont_T:=0;
        DeshBot(FEvaluarSA);
        Panel3.Visible:=True;
         Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 15
a el término -A-B-E(CD+-C)';
        Label29.Visible:=True;
        BRegresar.Visible:=True;
        BRegresar.Enabled:=True;
       end
       else
        if    ((Cont_T=2)    and    (Ecuacion='X=-(A+B+-C+-D+E)+-A-B-E(-
C+CD)+ABC')) then
        begin
         ContB:=ContB+1;
         Tres.Visible:=True;
         Ecuacion:='X=-(A+B+-C+-D+E)+-A-B-E(-C+D)+ABC';
         Cont_T:=Cont_T+1;
         Tres.Text:=Ecuacion;
         Panel3.Visible:=True;
          Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 15
a el término -A-B-E(-C+CD)';
        end
        else
         if    ((Cont_T=1)    and    (Ecuacion='X=-(A+B+-C+-D+E)+-A-B-E(-
C+CD)+ABC(D-E+1)')) then
         begin
          ContB:=ContB+1;
          Dos.Visible:=True;
          Ecuacion:='X=-(A+B+-C+-D+E)+-A-B-E(-C+D)+ABC(D-E+1)';
          Cont_T:=Cont_T+1;
          Dos.Text:=Ecuacion;
          Panel3.Visible:=True;
          Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 15
a el término -A-B-E(-C+CD)';
         end
         else
          if      ((Cont_T=3)      and      (Ecuacion='X=-A-BCD-E+-A-B-E(-
C+CD)+ABC(D-E+1)')) then
          begin
           ContB:=ContB+1;
           Cuatro.Visible:=True;
           Ecuacion:='X=-A-BCD-E+-A-B-E(-C+D)+ABC(D-E+1)';
```

```
        Cont_T:=Cont_T+1;
        Cuatro.Text:=Ecuacion;
        Panel3.Visible:=True;
        Memo1.Text:='Obtuvimos este resultado aplicando el Teorema
15 a el término -A-B-E(-C+CD)';
       end
       else
        if    ((Cont_T=3)    and    (Ecuacion='X=-A-B--C--D-E+-A-B-E(-
C+CD)+ABC')) then
        begin
         ContB:=ContB+1;
         Cuatro.Visible:=True;
         Ecuacion:='X=-A-B--C--D-E+-A-B-E(-C+D)+ABC';
         Cont_T:=Cont_T+1;
         Cuatro.Text:=Ecuacion;
         Panel3.Visible:=True;
          Memo1.Text:='Obtuvimos este resultado aplicando el Teorema
15 a el término -A-B-E(-C+CD)';
        end
        else
         if    ((Cont_T=2)    and    (Ecuacion='X=-A-B--C--D-E+-A-B-E(-
C+CD)+ABC(D-E+1)')) then
         begin
          ContB:=ContB+1;
          Tres.Visible:=True;
          Ecuacion:='X=-A-B--C--D-E+-A-B-E(-C+D)+ABC(D-E+1)';
          Cont_T:=Cont_T+1;
          Tres.Text:=Ecuacion;
          Panel3.Visible:=True;
           Memo1.Text:='Obtuvimos este resultado aplicando el Teorema
15 a el término -A-B-E(-C+CD)';
         end
         else
          if     ((Cont_T=4)     and     (Ecuacion='X=-A-BCD-E+-A-B-E(-
C+CD)+ABC')) then
          begin
           ContB:=ContB+1;
           Cinco.Visible:=True;
           Ecuacion:='X=-A-BCD-E+-A-B-E(-C+D)+ABC';
           Cont_T:=Cont_T+1;
           Cinco.Text:=Ecuacion;
           Panel3.Visible:=True;
           Memo1.Text:='Obtuvimos este resultado aplicando el Teorema
15 a el término -A-B-E(-C+CD)';
          end
          else
           if     ((Cont_T=4)     and     (Ecuacion='X=-A-BCD-E+-A-B-E(-
C+CD)+ABC')) then
           begin
            ContB:=ContB+1;
            Cinco.Visible:=True;
```

```
            Ecuacion:='X=-A-BCD-E+-A-B-E(-C+D)+ABC';
            Cont_T:=Cont_T+1;
            Cinco.Text:=Ecuacion;
            Panel3.Visible:=True;
            Memo1.Text:='Obtuvimos este resultado aplicando el
Teorema 15 a el término -A-B-E(-C+CD)';
           end
           else
           begin
            Panel4.Visible:=True;
            Memo2.Text:='El Teorema 15 no es aplicable en este caso';
            NError:=NError+1;
           end
    else
    begin
     if (Cont_T=0) then
      MostL;
     Panel4.Visible:=True;
     Memo2.Text:='El Teorema 15 no es aplicable en este caso';
     NError:=NError+1;
    end;
end;
```

PROCEDIMIENTO EVALUAR_SIMPLIFICA7. ermite aplicar el teorema X+X=X del álgebra booleana en la evaluación sobre simplificación algebraica, una vez lo ha seleccionado el usuario.

```
procedure TFEvaluarSA.B7Click(Sender: TObject);
begin
 if (Ejercicio='X=-A-B-C-D-E+-(A+B+C+D+E)+ABCDE+AB-CD-E') then
  if ((Cont_T=1) and (Ecuacion='X=-A-B-C-D-E+-A-B-C-D-E+ABCDE+AB-
CD-E')) then
  begin
   ContB:=ContB+1;
   Dos.Visible:=True;
   Ecuacion:='X=-A-B-C-D-E+ABCDE+AB-CD-E';
   Cont_T:=Cont_T+1;
   Dos.Text:=Ecuacion;
   Panel3.Visible:=True;
   Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 7 al
término -A-B-C-D-E con -A-B-C-D-E';
  end
  else
```

```
   if ((Cont_T=2) and (Ecuacion='X=-A-B-C-D-E+-A-B-C-D-E+ABD(CE+-C-
E)')) then
   begin
    ContB:=ContB+1;
    Tres.Visible:=True;
    Ecuacion:='X=-A-B-C-D-E+ABD(CE+-C-E)';
    Tres.Font.Color:=clWhite;
    Tres.Text:=Ecuacion;
    Cont_T:=0;
    DeshBot(FEvaluarSA);
    Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 7 al
término -A-B-C-D-E con -A-B-C-D-E';
    Label29.Visible:=True;
    BRegresar.Visible:=True;
    BRegresar.Enabled:=True;
   end
   else
   begin
    Panel4.Visible:=True;
    Memo2.Text:='El Teorema 7 no es aplicable en este caso';
    NError:=NError+1;
   end
 else
  if (Ejercicio='X=-(-A+-B+-C+-D+-E+-F)+ABCDEF+-A-BCDE-F+A-BCD-E-
F') then
   if ((Cont_T=2) and (Ecuacion='X=ABCDEF+ABCDEF+-A-BCDE-F+A-
BCD-E-F')) then
   begin
    ContB:=ContB+1;
    Tres.Visible:=True;
    Ecuacion:='X=ABCDEF+-A-BCDE-F+A-BCD-E-F';
    Cont_T:=Cont_T+1;
    Tres.Text:=Ecuacion;
    Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 7 al
término ABCDEF con ABCDEF';
   end
   else
    if ((Cont_T=3) and (Ecuacion='X=ABCDEF+ABCDEF+-BCD-F(-AE+A-
E)')) then
    begin
     ContB:=ContB+1;
     Cuatro.Visible:=True;
     Ecuacion:='X=ABCDEF+-BCD-F(-AE+A-E)';
     Cont_T:=Cont_T+1;
     Cuatro.Text:=Ecuacion;
     Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 7 al
término ABCDEF con ABCDEF';
    end
```

```
   else
    if (Cont_T=5) then
    begin
     ContB:=ContB+1;
     Seis.Visible:=True;
     Ecuacion:='X=CD(ABEF+-B-F(-AE+A-E))';
     Seis.Font.Color:=clWhite;
     Seis.Text:=Ecuacion;
     Cont_T:=0;
     DeshBot(FEvaluarSA);
     Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 7 al
término ABCDEF con ABCDEF';
     Label29.Visible:=True;
     BRegresar.Visible:=True;
     BRegresar.Enabled:=True;
    end
    else
    begin
     Panel4.Visible:=True;
     Memo2.Text:='El Teorema 7 no es aplicable en este caso';
     NError:=NError+1;
    end
  else
   if                    (Ejercicio='X=-(A+B+C+D+E+F)+-A-B-C-D-E-F+-A-
BCDEF+ABCDEF+AB-CDE-F') then
   if ((Cont_T=1) and (Ecuacion='X=-A-B-C-D-E-F+-A-B-C-D-E-F+-A-
BCDEF+ABCDEF+AB-CDE-F')) then
   begin
    ContB:=ContB+1;
    Dos.Visible:=True;
    Ecuacion:='X=-A-B-C-D-E-F+-A-BCDEF+ABCDEF+AB-CDE-F';
    Cont_T:=Cont_T+1;
    Dos.Text:=Ecuacion;
    Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 7 al
término -A-B-C-D-E-F';
   end
   else
    if     ((Cont_T=3)     and     (Ecuacion='X=-C-F(-A-B-D-E+-A-B-D-
E+ABDE)+CDEF(-A-B+AB)')) then
    begin
     ContB:=ContB+1;
     Cuatro.Visible:=True;
     Ecuacion:='X=-C-F(-A-B-D-E+ABDE)+CDEF(-A-B+AB)';
     Cuatro.Font.Color:=clWhite;
     Cuatro.Text:=Ecuacion;
     Cont_T:=0;
     DeshBot(FEvaluarSA);
     Panel3.Visible:=True;
```

```
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 7 al
término -A-B-D-E con -A-B-D-E';
     Label29.Visible:=True;
     BRegresar.Visible:=True;
     BRegresar.Enabled:=True;
    end
    else
    begin
     Panel4.Visible:=True;
     Memo2.Text:='El Teorema 7 no es aplicable en este caso';
     NError:=NError+1;
    end
   else
    if (Ejercicio='X=(-A+B)(A+B+C)-D') then
    if ((Cont_T=6) and (Ecuacion='X=B-D+C-D(-A+B)+B-D')) then
    begin
     ContB:=ContB+1;
     Siete.Visible:=True;
     Ecuacion:='X=B-D+C-D(-A+B)';
     Siete.Font.Color:=clWhite;
     Siete.Text:=Ecuacion;
     Cont_T:=0;
     DeshBot(FEvaluarSA);
     Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 7 a el
término B-D con B-D';
     Label29.Visible:=True;
     BRegresar.Visible:=True;
     BRegresar.Enabled:=True;
    end
    else
    begin
     Panel4.Visible:=True;
     Memo2.Text:='El Teorema 7 no es aplicable en este caso';
     NError:=NError+1;
    end
   else
    if    (Ejercicio='X=-(A+B+-C+-D+E)+-A-B-C-E+ABCD-E+ABC+-A-BCD-
E') then
     if    ((Cont_T=2)    and    (Ecuacion='X=-A-BCD-E+-A-B-C-E+ABCD-
E+ABC+-A-BCD-E')) then
     begin
      ContB:=ContB+1;
      Tres.Visible:=True;
      Ecuacion:='X=-A-BCD-E+-A-B-C-E+ABCD-E+ABC';
      Cont_T:=Cont_T+1;
      Tres.Text:=Ecuacion;
      Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 7 a
los términos -A-BCD-E y -A-BCD-E';
     end
```

```
     else
     begin
      Panel4.Visible:=True;
      Memo2.Text:='El Teorema 7 no es aplicable en este caso';
      NError:=NError+1;
     end
    else
    begin
     if (Cont_T=0) then
      MostL;
     Panel4.Visible:=True;
     Memo2.Text:='El Teorema 7 no es aplicable en este caso';
     NError:=NError+1;
    end;
end;
```

PROCEDIMIENTO EVALUAR_SIMPLIFICA18. Permite aplicar el teorema --X=X del álgebra booleana en la evaluación sobre simplificación algebraica, una vez lo ha seleccionado el usuario.

```
procedure TFEvaluarSA.B18Click(Sender: TObject);
begin
 if (Ejercicio='X=-(A+B+-C+D+E+F)+A-B-CDEF+-ABCD-E-F+-(-A+-B+-C+-
D+-E+-F)') then
  if ((Cont_T=1) and (Ecuacion='X=-A-B--C-D-E-F+A-B-CDEF+-ABCD-E-
F+--A--B--C--D--E--F')) then
  begin
   ContB:=ContB+1;
   Dos.Visible:=True;
   Ecuacion:='X=-A-BC-D-E-F+A-B-CDEF+-ABCD-E-F+ABCDEF';
   Cont_T:=Cont_T+1;
   Dos.Text:=Ecuacion;
   Panel3.Visible:=True;
   Memo1.Text:='Obtuvimos este resultado cancelando las inversiones
dobles en los término -A-B--C-D-E-F y --A--B--C--D--E--F';
  end
  else
   if ((Cont_T=2) and (Ecuacion='X=-A-B--C-D-E-F+D(A-B-CEF+-ABC-E-
F)+--A--B--C--D--E--F')) then
   begin
    ContB:=ContB+1;
    Tres.Visible:=True;
    Ecuacion:='X=-A-BC-D-E-F+D(A-B-CEF+-ABC-E-F)+ABCDEF';
    Cont_T:=Cont_T+1;
    Tres.Text:=Ecuacion;
```

```
   Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado cancelando las inversiones
dobles en los término -A-B--C-D-E-F y --A--B--C--D--E--F';
  end
  else
  begin
   Panel4.Visible:=True;
   Memo2.Text:='Esta ecuación no tiene inversiones dobles que
cancelar';
   NError:=NError+1;
  end
 else
  if (Ejercicio='X=-(-A+-B+-C+-D+-E+-F)+ABCDEF+-A-BCDE-F+A-BCD-E-
F') then
  if ((Cont_T=1) and (Ecuacion='X=--A--B--C--D--E--F+ABCDEF+-A-
BCDE-F+A-BCD-E-F')) then
  begin
   ContB:=ContB+1;
   Dos.Visible:=True;
   Ecuacion:='X=ABCDEF+ABCDEF+-A-BCDE-F+A-BCD-E-F';
   Cont_T:=Cont_T+1;
   Dos.Text:=Ecuacion;
   Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado cancelando las inversiones
dobles en el término --A--B--C--D--E--F';
  end
  else
    if ((Cont_T=2) and (Ecuacion='X=--A--B--C--D--E--F+ABCDEF+-BCD-
F(-AE+A-E)')) then
   begin
    ContB:=ContB+1;
    Tres.Visible:=True;
    Ecuacion:='X=ABCDEF+ABCDEF+-BCD-F(-AE+A-E)';
    Cont_T:=Cont_T+1;
    Tres.Text:=Ecuacion;
    Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado cancelando las inversiones
dobles en el término --A--B--C--D--E--F';
   end
   else
    if ((Cont_T=3) and (Ecuacion='X=--A--B--C--D--E--F+CD(ABEF+-B-F(-
AE+A-E))')) then
    begin
     ContB:=ContB+1;
     Cuatro.Visible:=True;
     Ecuacion:='X=ABCDEF+CD(ABEF+-B-F(-AE+A-E))';
     Cont_T:=Cont_T+1;
     Cuatro.Text:=Ecuacion;
     Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado cancelando las inversiones
dobles en el término --A--B--C--D--E--F';
```

```
    end
    else
    begin
     Panel4.Visible:=True;
     Memo2.Text:='Esta ecuación no tiene inversiones dobles que
cancelar';
     NError:=NError+1;
    end
  else
   if (Ejercicio='X=A-BC-D+-(A+B+C+-D)+-A-BCD') then
   if ((Cont_T=1) and (Ecuacion='X=A-BC-D+-A-B-C--D+-A-BCD')) then
   begin
    ContB:=ContB+1;
    Dos.Visible:=True;
    Ecuacion:='X=A-BC-D+-A-B-CD+-A-BCD';
    Cont_T:=Cont_T+1;
    Dos.Text:=Ecuacion;
    Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado cancelando las inversiones
dobles en el término -A-B-C--D';
   end
   else
    if ((Cont_T=2) and (Ecuacion='X=-BC(A-D+-AD)+-A-B-C--D')) then
    begin
     ContB:=ContB+1;
     Tres.Visible:=True;
     Ecuacion:='X=-BC(A-D+-AD)+-A-B-CD';
     Cont_T:=Cont_T+1;
     Tres.Text:=Ecuacion;
     Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado cancelando las inversiones
dobles en el término -A-B-C--D';
    end
    else
    begin
     Panel4.Visible:=True;
     Memo2.Text:='Esta ecuación no tiene inversiones dobles que
cancelar';
     NError:=NError+1;
    end
  else
   if (Ejercicio='X=-(A+B+-C)+-A-BCD+-AB-CD') then
    if ((Cont_T=2) and (Ecuacion='X=-A-B--C+-AD(-BC+B-C)')) then
    begin
     ContB:=ContB+1;
     Tres.Visible:=True;
     Ecuacion:='X=-A-BC+-AD(-BC+B-C)';
     Cont_T:=Cont_T+1;
     Tres.Text:=Ecuacion;
     Panel3.Visible:=True;
```

```
     Memo1.Text:='Obtuvimos este resultado cancelando las inversiones
dobles en el término -A-B--C';
    end
    else
     if ((Cont_T=1) and (Ecuacion='X=-A-B--C+-A-BCD+-AB-CD')) then
     begin
      ContB:=ContB+1;
      Dos.Visible:=True;
      Ecuacion:='X=-A-BC+-A-BCD+-AB-CD';
      Cont_T:=Cont_T+1;
      Dos.Text:=Ecuacion;
      Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado cancelando las inversiones
dobles en el término -A-B--C';
     end
     else
     begin
      Panel4.Visible:=True;
      Memo2.Text:='Esta ecuación no tiene inversiones dobles que
cancelar';
      NError:=NError+1;
     end
   else
    if (Ejercicio='X=-(A+B+-C+-D+E)+-ABC-DE+A-B-C-DE+ABCDE') then
     if ((Cont_T=2) and (Ecuacion='X=-A-B--C--D-E+BCE(-A-D+AD)+A-B-
C-DE')) then
     begin
      ContB:=ContB+1;
      Tres.Visible:=True;
      Ecuacion:='X=-A-BCD-E+BCE(-A-D+AD)+A-B-C-DE';
      Cont_T:=Cont_T+1;
      Tres.Text:=Ecuacion;
      Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado cancelando las inversiones
dobles en el término -A-B--C--D-E';
     end
     else
      if ((Cont_T=1) and (Ecuacion='X=-A-B--C--D-E+-ABC-DE+A-B-C-
DE+ABCDE')) then
      begin
       ContB:=ContB+1;
       Dos.Visible:=True;
       Ecuacion:='X=-A-BCD-E+-ABC-DE+A-B-C-DE+ABCDE';
       Cont_T:=Cont_T+1;
       Dos.Text:=Ecuacion;
       Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado cancelando las
inversiones dobles en el término -A-B--C--D-E';
      end
      else
      begin
```

```
      Panel4.Visible:=True;
      Memo2.Text:='Esta ecuación no tiene inversiones dobles que
cancelar';
      NError:=NError+1;
      end
    else
     if (Ejercicio='X=(B+-C)(-B+C)+-(-A+B+-C)') then
      if ((Cont_T=3) and (Ecuacion='X=BC+-B-C+--A-B--C')) then
      begin
       ContB:=ContB+1;
       Cuatro.Visible:=True;
       Ecuacion:='X=BC+-B-C+A-BC';
       Cont_T:=Cont_T+1;
       Cuatro.Text:=Ecuacion;
       Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado cancelando las
inversiones dobles en el término --A-B--C';
      end
      else
       if ((Cont_T=1) and (Ecuacion='X=(B+-C)(-B+C)+--A-B--C')) then
       begin
        ContB:=ContB+1;
        Dos.Visible:=True;
        Ecuacion:='X=(B+-C)(-B+C)+A-BC';
        Cont_T:=Cont_T+1;
        Dos.Text:=Ecuacion;
        Panel3.Visible:=True;
        Memo1.Text:='Obtuvimos este resultado cancelando las
inversiones dobles en el término --A-B--C';
       end
       else
        if ((Cont_T=2) and (Ecuacion='X=-BB+BC+-B-C+-CC+--A-B--C'))
then
        begin
         ContB:=ContB+1;
         Tres.Visible:=True;
         Ecuacion:='X=-BB+BC+-B-C+-CC+A-BC';
         Cont_T:=Cont_T+1;
         Tres.Text:=Ecuacion;
         Panel3.Visible:=True;
         Memo1.Text:='Obtuvimos este resultado cancelando las
inversiones dobles en el término --A-B--C';
        end
        else
        begin
         Panel4.Visible:=True;
         Memo2.Text:='Esta ecuación no tiene inversiones dobles que
cancelar';
         NError:=NError+1;
        end
     else
```

```
     if (Ejercicio='X=-(A+B+-C+-D+E)+-A-B-C-E+ABCD-E+ABC+-A-BCD-
E') then
     if ((Cont_T=1) and (Ecuacion='X=-A-B--C--D-E+-A-B-C-E+ABCD-
E+ABC+-A-BCD-E')) then
      begin
       ContB:=ContB+1;
       Dos.Visible:=True;
       Ecuacion:='X=-A-BCD-E+-A-B-C-E+ABCD-E+ABC+-A-BCD-E';
       Cont_T:=Cont_T+1;
       Dos.Text:=Ecuacion;
       Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado cancelando las
inversiones dobles en el término -A-B--C--D-E';
      end
      else
       if ((Cont_T=4) and (Ecuacion='X=-A-B--C--D-E+-A-B-E(-
C+D)+ABC')) then
       begin
        ContB:=ContB+1;
        Cinco.Visible:=True;
        Ecuacion:='X=-A-BCD-E+-A-B-E(-C+D)+ABC';
        Cont_T:=Cont_T+1;
        Cinco.Text:=Ecuacion;
        Panel3.Visible:=True;
        Memo1.Text:='Obtuvimos este resultado cancelando las
inversiones dobles en el término -A-B--C--D-E';
       end
       else
        if ((Cont_T=2) and (Ecuacion='X=-A-B--C--D-E+-A-B-E(-
C+CD)+ABC(D-E+1)')) then
        begin
         ContB:=ContB+1;
         Tres.Visible:=True;
         Ecuacion:='X=-A-BCD-E+-A-B-E(-C+CD)+ABC(D-E+1)';
         Cont_T:=Cont_T+1;
         Tres.Text:=Ecuacion;
         Panel3.Visible:=True;
         Memo1.Text:='Obtuvimos este resultado cancelando las
inversiones dobles en el término -A-B--C--D-E';
        end
        else
         if ((Cont_T=3) and (Ecuacion='X=-A-B--C--D-E+-A-B-E(-
C+CD)+ABC')) then
         begin
          ContB:=ContB+1;
          Cuatro.Visible:=True;
          Ecuacion:='X=-A-BCD-E+-A-B-E(-C+CD)+ABC';
          Cont_T:=Cont_T+1;
          Cuatro.Text:=Ecuacion;
          Panel3.Visible:=True;
```

```
        Memo1.Text:='Obtuvimos este resultado cancelando las
inversiones dobles en el término -A-B--C--D-E';
       end
       else
        if ((Cont_T=3) and (Ecuacion='X=-A-B--C--D-E+-A-B-E(-
C+D)+ABC(D-E+1)')) then
        begin
         ContB:=ContB+1;
         Cuatro.Visible:=True;
         Ecuacion:='X=-A-BCD-E+-A-B-E(-C+D)+ABC(D-E+1)';
         Cont_T:=Cont_T+1;
         Cuatro.Text:=Ecuacion;
         Panel3.Visible:=True;
         Memo1.Text:='Obtuvimos este resultado cancelando las
inversiones dobles en el término -A-B--C--D-E';
        end
        else
        begin
         Panel4.Visible:=True;
         Memo2.Text:='Esta ecuación no tiene inversiones dobles que
cancelar';
         NError:=NError+1;
        end
     else
     begin
      if (Cont_T=0) then
       MostL;
      Panel4.Visible:=True;
      Memo2.Text:='Esta ecuación no tiene inversiones dobles que
cancelar';
      NError:=NError+1;
     end;
end;
```

PROCEDIMIENTO EVALUAR_SIMPLIFICA6. Permite aplicar el teorema X+1=1 del álgebra booleana en la evaluación sobre simplificación algebraica, una vez lo ha seleccionado el usuario.

```
procedure TFEvaluarSA.B6Click(Sender: TObject);
begin
 if (Ejercicio='X=-(A+B+-C)+-A-BCD+-AB-CD') then
  if ((Cont_T=3) and (Ecuacion='X=-A-BC(1+D)+-AB-CD')) then
  begin
   ContB:=ContB+1;
```

```
   Cuatro.Visible:=True;
   Ecuacion:='X=-A-BC+-AB-CD';
   Cont_T:=Cont_T+1;
   Cuatro.Text:=Ecuacion;
   Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 6 a el
término -A-BC(1+D)';
  end
  else
   if ((Cont_T=6) and (Ecuacion='X=-A(-BC(1+D)+B-CD)')) then
   begin
    ContB:=ContB+1;
    Siete.Visible:=True;
    Ecuacion:='X=-A(-BC+B-CD)';
    Siete.Font.Color:=clWhite;
    Siete.Text:=Ecuacion;
    Cont_T:=0;
    DeshBot(FEvaluarSA);
    Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 6 a el
término -BC(1+D)';
    Label29.Visible:=True;
    BRegresar.Visible:=True;
    BRegresar.Enabled:=True;
   end
   else
   begin
    Panel4.Visible:=True;
    Memo2.Text:='ElTeorema 6 no es aplicable en este caso';
    NError:=NError+1;
   end
 else
  if   (Ejercicio='X=-(A+B+-C+-D+E)+-A-B-C-E+ABCD-E+ABC+-A-BCD-E')
then
   if   ((Cont_T=3)   and   (Ecuacion='X=-A-B-E(CD+-C)+ABC(D-E+1)+-A-
BCD-E')) then
   begin
    ContB:=ContB+1;
    Cuatro.Visible:=True;
    Ecuacion:='X=-A-B-E(CD+-C)+ABC+-A-BCD-E';
    Cont_T:=Cont_T+1;
    Cuatro.Text:=Ecuacion;
    Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 6 a el
término ABC(D-E+1)';
   end
   else
    if ((Cont_T=7) and (Ecuacion='X=-A-B-E(-C+D(1+C))+ABC')) then
    begin
     ContB:=ContB+1;
     Ocho.Visible:=True;
```

```
    Ecuacion:='X=-A-B-E(-C+D(1))+ABC';
    Cont_T:=Cont_T+1;
    Ocho.Text:=Ecuacion;
    Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 6 a el
término 1 con C';
   end
   else
    if ((Cont_T=5) and (Ecuacion='X=-A-B-E(-C+D)+ABC(D-E+1)')) then
    begin
     ContB:=ContB+1;
     Seis.Visible:=True;
     Ecuacion:='X=-A-B-E(-C+D)+ABC';
     Seis.Font.Color:=clWhite;
     Seis.Text:=Ecuacion;
     Cont_T:=0;
     DeshBot(FEvaluarSA);
     Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 6 a el
término ABC(D-E+1)';
     Label29.Visible:=True;
     BRegresar.Visible:=True;
     BRegresar.Enabled:=True;
    end
    else
     if ((Cont_T=4) and (Ecuacion='X=-A-B-E(CD+-C)+ABC(D-E+1)'))
then
     begin
      ContB:=ContB+1;
      Cinco.Visible:=True;
      Ecuacion:='X=-A-B-E(CD+-C)+ABC';
      Cont_T:=Cont_T+1;
      Cinco.Text:=Ecuacion;
      Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 6 a
el término ABC(D-E+1)';
     end
     else
    if ((Cont_T=1) and (Ecuacion='X= -(A+B+-C+-D+E) +
   -A-B-E(-C+CD)+ABC(D-E+1)')) then
      begin
       ContB:=ContB+1;
       Dos.Visible:=True;
       Ecuacion:='X=-(A+B+-C+-D+E)+-A-B-E(-C+CD)+ABC';
       Cont_T:=Cont_T+1;
       Dos.Text:=Ecuacion;
       Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 6 a
       el término ABC(D-E+1)';
      end
      else
```

```
      if ((Cont_T=2) and (Ecuacion='X= -(A+B+-C+-D+E)+
     -A-B-E(-C+D)+ABC(D-E+1)')) then
      begin
       ContB:=ContB+1;
       Tres.Visible:=True;
       Ecuacion:='X=-(A+B+-C+-D+E)+-A-B-E(-C+D)+ABC';
       Cont_T:=Cont_T+1;
       Tres.Text:=Ecuacion;
       Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 6 a
      el término ABC(D-E+1)';
      end
      else
       if ((Cont_T=4) and (Ecuacion='X=-A-BCD-E+-A-B-E(-C+D)+
      ABC(D-E+1)')) then
       begin
        ContB:=ContB+1;
        Cinco.Visible:=True;
        Ecuacion:='X=-A-BCD-E+-A-B-E(-C+D)+ABC';
        Cont_T:=Cont_T+1;
        Cinco.Text:=Ecuacion;
        Panel3.Visible:=True;
        Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 6 a
       el término ABC(D-E+1)';
       end
       else
        if ((Cont_T=2) and (Ecuacion='X=-A-B--C--D-E+-A-B-E(-
C+CD)+ABC(D-E+1)')) then
        begin
         ContB:=ContB+1;
         Tres.Visible:=True;
         Ecuacion:='X=-A-B--C--D-E+-A-B-E(-C+CD)+ABC';
         Cont_T:=Cont_T+1;
         Tres.Text:=Ecuacion;
         Panel3.Visible:=True;
          Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 6
a el término ABC(D-E+1)';
        end
        else
         if ((Cont_T=3) and (Ecuacion='X=-A-B--C--D-E+-A-B-E(-
C+D)+ABC(D-E+1)')) then
         begin
          ContB:=ContB+1;
          Cuatro.Visible:=True;
          Ecuacion:='X=-A-B--C--D-E+-A-B-E(-C+D)+ABC';
          Cont_T:=Cont_T+1;
          Cuatro.Text:=Ecuacion;
          Panel3.Visible:=True;
          Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 6
a el término ABC(D-E+1)';
         end
```

```
        else
         if     ((Cont_T=3)     and     (Ecuacion='X=-A-BCD-E+-A-B-E(-
C+CD)+ABC(D-E+1)')) then
         begin
          ContB:=ContB+1;
          Cuatro.Visible:=True;
          Ecuacion:='X=-A-BCD-E+-A-B-E(-C+CD)+ABC';
          Cont_T:=Cont_T+1;
          Cuatro.Text:=Ecuacion;
          Panel3.Visible:=True;
           Memo1.Text:='Obtuvimos este resultado aplicando el Teorema
6 a el término ABC(D-E+1)';
         end
         else
         begin
          Panel4.Visible:=True;
          Memo2.Text:='El Teorema 6 no es aplicable en este caso';
          NError:=NError+1;
         end
  else
  begin
   if (Cont_T=0) then
    MostL;
   Panel4.Visible:=True;
   Memo2.Text:='El Teorema 6 no es aplicable en este caso';
   NError:=NError+1;
  end;
end;
```

PROCEDIMIENTO EVALUAR_SIMPLIFICA13. Permite aplicar el teorema (W+X)(Y+Z)=WY+XY+WZ+XZ del álgebra booleana en la evaluación sobre simplificación algebraica, una vez lo ha seleccionado el usuario.

```
procedure TFEvaluarSA.B13BClick(Sender: TObject);
begin
 if (Ejercicio='X=(-A+B)(A+B+C)-D') then
  if (Cont_T=0) then
  begin
   ContB:=ContB+1;
   OculL;
   LEcuacion.Visible:=True;
   LEcuacion.Text:=Ejercicio;
   Uno.Visible:=True;
```

```
   Ecuacion:='X=(-AA+-AB+-AC+AB+BB+BC)-D';
   Cont_T:=Cont_T+1;
   Uno.Text:=Ecuacion;
   Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 13B a
el término (-A+B) con (A+B+C)';
  end
  else
   if ((Cont_T=1) and (Ecuacion='X=(-A+B)(A-D+B-D+C-D)')) then
   begin
    ContB:=ContB+1;
    Dos.Visible:=True;
    Ecuacion:='X=-AA-D+-AB-D+-AC-D+AB-D+BB-D+BC-D';
    Cont_T:=Cont_T+1;
    Dos.Text:=Ecuacion;
    Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 13B a
los términos (-A+B) con (A-D+B-D+C-D)';
   end
   else
   begin
    Panel4.Visible:=True;
    Memo2.Text:='El Teorema 13B no es aplicable en este caso';
    NError:=NError+1;
   end
 else
  if (Ejercicio='X=(B+-C)(-B+C)+-(-A+B+-C)') then
   if (Cont_T=0) then
   begin
    ContB:=ContB+1;
    OculL;
    LEcuacion.Visible:=True;
    LEcuacion.Text:=Ejercicio;
    Uno.Visible:=True;
    Ecuacion:='X=-BB+BC+-B-C+-CC+-(-A+B+-C)';
    Cont_T:=Cont_T+1;
    Uno.Text:=Ecuacion;
    Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 13B a
el término (B+-C) con (-B+C)';
   end
   else
    if ((Cont_T=2) and (Ecuacion='X=(B+-C)(-B+C)+A-BC')) then
    begin
     ContB:=ContB+1;
     Tres.Visible:=True;
     Ecuacion:='X=-BB+BC+-B-C+-CC+A-BC';
     Cont_T:=Cont_T+1;
     Tres.Text:=Ecuacion;
     Panel3.Visible:=True;
```

```
    Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 13B a
el término (B+-C) con (-B+C)';
   end
   else
    if ((Cont_T=1) and (Ecuacion='X=(B+-C)(-B+C)+--A-B--C')) then
    begin
     ContB:=ContB+1;
     Dos.Visible:=True;
     Ecuacion:='X=-BB+BC+-B-C+-CC+--A-B--C';
     Cont_T:=Cont_T+1;
     Dos.Text:=Ecuacion;
     Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 13B a
el término (B+-C) con (-B+C)';
    end
    else
    begin
     Panel4.Visible:=True;
     Memo2.Text:='El Teorema 13B no es aplicable en este caso';
     NError:=NError+1;
    end
  else
  begin
   if (Cont_T=0) then
    MostL;
   Panel4.Visible:=True;
   Memo2.Text:='El Teorema 13B no es aplicable en este caso';
   NError:=NError+1;
  end;
end;
```

PROCEDIMIENTO EVALUAR_SIMPLIFICA4. Permite aplicar el teorema X-X=0 del álgebra booleana en la evaluación sobre simplificación algebraica, una vez lo ha seleccionado el usuario.

```
procedure TFEvaluarSA.B4Click(Sender: TObject);
begin
 if (Ejercicio='X=(-A+B)(A+B+C)-D') then
  if ((Cont_T=2) and (Ecuacion='X=-AA-D+-AB-D+-AC-D+AB-D+BB-D+BC-
D')) then
  begin
   ContB:=ContB+1;
   Tres.Visible:=True;
   Ecuacion:='X=-AB-D+-AC-D+AB-D+BB-D+BC-D';
   Cont_T:=Cont_T+1;
```

```
   Tres.Text:=Ecuacion;
   Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 4 a el
término -AA-D';
  end
  else
   if ((Cont_T=3) and (Ecuacion='X=-AA-D+-AB-D+-AC-D+AB-D+B-D+BC-
D')) then
   begin
    ContB:=ContB+1;
    Cuatro.Visible:=True;
    Ecuacion:='X=-AB-D+-AC-D+AB-D+B-D+BC-D';
    Cont_T:=Cont_T+1;
    Cuatro.Text:=Ecuacion;
    Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 4 a el
término -AA-D';
   end
   else
    if ((Cont_T=1) and (Ecuacion='X=(-AA+-AB+-AC+AB+BB+BC)-D'))
then
    begin
     ContB:=ContB+1;
     Dos.Visible:=True;
     Ecuacion:='X=(-AB+-AC+AB+BB+BC)-D';
     Cont_T:=Cont_T+1;
     Dos.Text:=Ecuacion;
     Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 4 a el
término -AA';
    end
    else
     if ((Cont_T=2) and (Ecuacion='X=(-AA+-AB+-AC+AB+B+BC)-D')) then
     begin
      ContB:=ContB+1;
      Tres.Visible:=True;
      Ecuacion:='X=(-AB+-AC+AB+B+BC)-D';
      Cont_T:=Cont_T+1;
      Tres.Text:=Ecuacion;
      Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 4 a el
término -AA';
     end
     else
     begin
      Panel4.Visible:=True;
      Memo2.Text:='El Teorema 4 no es aplicable en este caso';
      NError:=NError+1;
     end
 else
  if (Ejercicio='X=(B+-C)(-B+C)+-(-A+B+-C)') then
```

```
  if ((Cont_T=1) and (Ecuacion='X=-BB+BC+-B-C+-CC+-(-A+B+-C)')) then
  begin
   ContB:=ContB+1;
   Dos.Visible:=True;
   Ecuacion:='X=BC+-B-C+-(-A+B+-C)';
   Cont_T:=Cont_T+1;
   Dos.Text:=Ecuacion;
   Panel3.Visible:=True;
   Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 4 a los términos -BB y -CC';
  end
  else
   if ((Cont_T=3) and (Ecuacion='X=B-B+BC+-B-C+C-C+A-BC')) then
   begin
    ContB:=ContB+1;
    Cuatro.Visible:=True;
    Ecuacion:='X=BC+-B-C+A-BC';
    Cont_T:=Cont_T+1;
    Cuatro.Text:=Ecuacion;
    Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 4 a los términos B-B y C-C';
   end
   else
    if ((Cont_T=2) and (Ecuacion='X=-BB+BC+-B-C+-CC+--A-B--C')) then
    begin
     ContB:=ContB+1;
     Tres.Visible:=True;
     Ecuacion:='X=BC+-B-C+--A-B--C';
     Cont_T:=Cont_T+1;
     Tres.Text:=Ecuacion;
     Panel3.Visible:=True;
      Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 4 a los términos -BB y -CC';
    end
    else
     if ((Cont_T=3) and (Ecuacion='X=-BB+BC+-B-C+-CC+A-BC')) then
     begin
      ContB:=ContB+1;
      Cuatro.Visible:=True;
      Ecuacion:='X=BC+-B-C+A-BC';
      Cont_T:=Cont_T+1;
      Cuatro.Text:=Ecuacion;
      Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 4 a los términos -BB y -CC';
     end
     else
      if ((Cont_T=5) and (Ecuacion='X=-BB+C(A+B)+-B-C+-CC')) then
      begin
```

```
      ContB:=ContB+1;
      Seis.Visible:=True;
      Ecuacion:='X=C(A+B)+-B-C';
      Seis.Font.Color:=clWhite;
      Seis.Text:=Ecuacion;
      Cont_T:=0;
      DeshBot(FEvaluarSA);
      Panel3.Visible:=True;
       Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 4 a
los términos -BB y -CC';
      Label29.Visible:=True;
      BRegresar.Visible:=True;
      BRegresar.Enabled:=True;
     end
     else
      if ((Cont_T=4) and (Ecuacion='X=-BB+C(B+A-B)+-B-C+-CC')) then
      begin
       ContB:=ContB+1;
       Cinco.Visible:=True;
       Ecuacion:='X=C(B+A-B)+-B-C';
       Cont_T:=Cont_T+1;
       Cinco.Text:=Ecuacion;
       Panel3.Visible:=True;
        Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 4 a
los términos -BB y -CC';
     end
     else
     begin
      Panel4.Visible:=True;
      Memo2.Text:='El Teorema 4 no es aplicable en este caso';
      NError:=NError+1;
     end
  else
  begin
   if (Cont_T=0) then
    MostL;
   Panel4.Visible:=True;
   Memo2.Text:='El Teorema 4 no es aplicable en este caso';
   NError:=NError+1;
  end;
end;
```

PROCEDIMIENTO EVALUAR_SIMPLIFICA3. Permite aplicar el teorema X· X=X del álgebra booleana en la evaluación sobre simplificación algebraica, una vez lo ha seleccionado el usuario.

```
procedure TFEvaluarSA.B3Click(Sender: TObject);
begin
 if (Ejercicio='X=(-A+B)(A+B+C)-D') then
  if ((Cont_T=3) and (Ecuacion='X=-AB-D+-AC-D+AB-D+BB-D+BC-D'))
then
  begin
   ContB:=ContB+1;
   Cuatro.Visible:=True;
   Ecuacion:='X=-AB-D+-AC-D+AB-D+B-D+BC-D';
   Cont_T:=Cont_T+1;
   Cuatro.Text:=Ecuacion;
   Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 3 a el
término BB-D';
  end
  else
   if ((Cont_T=2) and (Ecuacion='X=-AA-D+-AB-D+-AC-D+AB-D+BB-
D+BC-D')) then
   begin
    ContB:=ContB+1;
    Tres.Visible:=True;
    Ecuacion:='X=-AA-D+-AB-D+-AC-D+AB-D+B-D+BC-D';
    Cont_T:=Cont_T+1;
    Tres.Text:=Ecuacion;
    Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 3 a el
término BB-D';
   end
   else
    if ((Cont_T=2) and (Ecuacion='X=(-AB+-AC+AB+BB+BC)-D')) then
    begin
     ContB:=ContB+1;
     Tres.Visible:=True;
     Ecuacion:='X=(-AB+-AC+AB+B+BC)-D';
     Cont_T:=Cont_T+1;
     Tres.Text:=Ecuacion;
     Panel3.Visible:=True;
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 3 a el
término BB';
    end
    else
     if ((Cont_T=1) and (Ecuacion='X=(-AA+-AB+-AC+AB+BB+BC)-D'))
then
     begin
      ContB:=ContB+1;
      Dos.Visible:=True;
      Ecuacion:='X=(-AA+-AB+-AC+AB+B+BC)-D';
      Cont_T:=Cont_T+1;
      Dos.Text:=Ecuacion;
      Panel3.Visible:=True;
```

```
     Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 3 a el
término BB';
    end
    else
    begin
     Panel4.Visible:=True;
     Memo2.Text:='El Teorema 4 no es aplicable en este caso';
     NError:=NError+1;
    end
 else
 begin
  if (Cont_T=0) then
   MostL;
  Panel4.Visible:=True;
  Memo2.Text:='El Teorema 4 no es aplicable en este caso';
  NError:=NError+1;
 end;
end;
```

PROCEDIMIENTO EVALUAR_SIMPLIFICA2. Permite aplicar el teorema X· 1=X del álgebra booleana en la evaluación sobre simplificación algebraica, una vez lo ha seleccionado el usuario.

```
procedure TFEvaluarSA.B2Click(Sender: TObject);
begin
 if    (Ejercicio='X=-(A+B+-C+-D+E)+-A-B-C-E+ABCD-E+ABC+-A-BCD-E')
then
  if ((Cont_T=8) and (Ecuacion='X=-A-B-E(-C+D(1))+ABC')) then
  begin
   ContB:=ContB+1;
   Nueve.Visible:=True;
   Ecuacion:='X=-A-B-E(-C+D)+ABC';
   Nueve.Font.Color:=clWhite;
   Nueve.Text:=Ecuacion;
   Cont_T:=0;
   DeshBot(FEvaluarSA);
   Panel3.Visible:=True;
    Memo1.Text:='Obtuvimos este resultado aplicando el Teorema 9 a el
término D(1)';
   Label29.Visible:=True;
   BRegresar.Visible:=True;
   BRegresar.Enabled:=True;
  end
  else
```

```
  begin
   Panel4.Visible:=True;
   Memo2.Text:='El Teorema 9 no es aplicable en este caso';
   NError:=NError+1;
  end
 else
 begin
  if (Cont_T=0) then
   MostL;
  Panel4.Visible:=True;
  Memo2.Text:='El Teorema 9 no es aplicable en este caso';
  NError:=NError+1;
 end;
end;
```

PROCEDIMIENTO ACTUALIZAR_SIMPLIFICACIÓN. Una vez el usuario ha sido evaluado en la leccció simplificación algebraica este procedimiento procede a actualizar sus datos en la base de datos tablas de acuerdo a los resultados obtenidos.

```
procedure TFEvaluarSA.BRegresarClick(Sender: TObject);
var
 PorcE,Punt,Desc,Puntaje,PuntE:Real;
 NumEjer:integer;
begin
 TError:=NError+ContB;
 NombForm:='FEvaluarSA';
 panel3.Visible:=false;
 panel4.Visible:=false;
 NumEjer:=FSimpAlg.Ejercicios_Propuestos.Fields[7].AsInteger;
 EjerSiNo[NumEjer]:='1';
 Punt:=FSimpAlg.Ejercicios_Propuestos.Fields[5].AsFloat;
 PorcE:=(NError*100)/TError;
 FSimpAlg.Ejercicios_Propuestos.Close;
 Leccion_Simplificacion.open;
 Leccion_Simplificacion.DisableControls;
 Leccion_Simplificacion.SetKey;
 Leccion_Simplificacion.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_Simplificacion.GotoKey;
 if (Leccion_Simplificacion.GotoKey=True) then
 begin//2
  Leccion_Simplificacion.Edit;
  if (PorcE<=40) then
```

```
  begin//3
   Punt:=((100-((NError*100)/(2*(Contm+1))))*Punt)/100;
   if (Leccion_Simplificacion.Fields[3].AsInteger=0) then
   begin
    Puntaje:=Punt;

Leccion_Simplificacion.Fields[2].AsFloat:=Leccion_Simplificacion.Fields[
2].AsFloat+Punt;
   end
   else
   begin
    Desc:=Punt-(Punt*0.05);
    Puntaje:=Desc;

Leccion_Simplificacion.Fields[2].AsFloat:=Leccion_Simplificacion.Fields[
2].AsFloat+Desc;
   end;
   Leccion_Simplificacion.Fields[4].AsString:=EjerSiNo;
   Leccion_Simplificacion.Post;
   Leccion_Simplificacion.EnableControls;
   Leccion_Simplificacion.Close;
   Estudiantes.Open;
   Estudiantes.DisableControls;
   Estudiantes.Setkey;
   Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
   Estudiantes.GotoKey;
   if (estudiantes.gotokey=true) then
   begin//4
    Estudiantes.Edit;
    if (niv<4)then
     Estudiantes.fields[6].AsInteger:=Estudiantes.fields[6].AsInteger+1;
    Estudiantes.fields[3].Asfloat:=Estudiantes.fields[3].Asfloat+Punt;
    PunT:=((Estudiantes.fields[3].AsFloat)/20);
    Estrellas:=Trunc(PunT);
    NEstr:=Estudiantes.fields[5].AsInteger;
    if (Estrellas>Estudiantes.fields[5].AsInteger) then
    begin
     Estudiantes.fields[5].AsInteger:=Estrellas;
     Panel3.Visible:=True;
     FEstrella.Show;
    end;
    Estudiantes.Post;
    Estudiantes.EnableControls;
    Estudiantes.Close;
   end;//4
   NumEquiv:=0;
   FFelicit.Memo2.Text:=FormatFloat('0.00',Puntaje);
   FFelicit.Show;
   FEvaluarSA.Hide;
  end//3
  else
```

```
  begin
   PuntE:=(Leccion_Simplificacion.Fields[2].AsFloat)*0.1;

Leccion_Simplificacion.Fields[2].AsFloat:=Leccion_Simplificacion.Fields[
2].AsFloat-PuntE;
   Leccion_Simplificacion.Fields[5].AsString:=EjerSiNo;
   Leccion_Simplificacion.Post;
   Leccion_Simplificacion.EnableControls;
   Leccion_Simplificacion.Close;
   Estudiantes.Open;
   Estudiantes.DisableControls;
   Estudiantes.Setkey;
   Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
   Estudiantes.GotoKey;
   if (estudiantes.gotokey=true) then
   begin
    Estudiantes.Edit;
    if (niv>1)then
     Estudiantes.fields[6].AsInteger:=Estudiantes.fields[6].AsInteger-1;
    Estudiantes.fields[3].Asfloat:=Estudiantes.fields[3].Asfloat-PuntE;
    PunT:=((Estudiantes.fields[3].AsFloat)/20);
    Estrellas:=Trunc(PunT);
    Estudiantes.fields[5].AsInteger:=Estrellas;
    Estudiantes.Post;
    Estudiantes.EnableControls;
    Estudiantes.Close;
   end;
//   FEvaluarSA.Hide;
   NumEquiv:=NumEquiv+1;
   FEquivoc.Memo6.Text:=FormatFloat('0.00',PuntE);
   FEquivoc.Show;
   FEvaluarSA.Hide;
  end;
 end;
end;
```

LECCION APLICACIONES DEL ÁLGEBRA BOOLEANA.

PROCEDIMIENTO INICIAR_APLICACIONES. Cuando un usuario visita la lección Aplicaciones del Álgebra Booleana, actualiza los datos de éste en las tablas Estudiantes Lección_Simplificación.

```
procedure TForm7.BAplicacClick(Sender: TObject);
begin
```

```
 form7.hide;
 Estudiantes.Open;
 Estudiantes.SetKey;
 Estudiantes.FieldByName('Cod_Est').AsString:=Codigo1;
 Estudiantes.GotoKey;
 if (Estudiantes.GotoKey=True) then
 begin
  Estudiantes.Edit;
  Estudiantes.Fields[4].AsString:='Aplicaciones del Algebra   Booleana';
  Estudiantes.Post;
  Estudiantes.EnableControls;
  Estudiantes.Close;
 end;
 Leccion_Aplicaciones.Open;
 Leccion_Aplicaciones.SetKey;
 Leccion_Aplicaciones.FieldByName('Cod_Est').AsString:=Codigo1;
 Leccion_Aplicaciones.GotoKey;
 if (Leccion_Aplicaciones.GotoKey=True) then
 begin
  Leccion_Aplicaciones.Edit;

Leccion_Aplicaciones.Fields[1].AsInteger:=Leccion_Aplicaciones.Fields[
1].AsInteger+1;
  Leccion_Aplicaciones.Post;
  Leccion_Aplicaciones.EnableControls;
  Leccion_Aplicaciones.Close;
 end
 else
  begin
   Leccion_Aplicaciones.DisableControls;
   Leccion_Aplicaciones.Last;
   Leccion_Aplicaciones.Insert;
   Leccion_Aplicaciones.Edit;
   Leccion_Aplicaciones.Fields[0].AsString:=Codigo1;
   Leccion_Aplicaciones.Fields[1].AsInteger:=1;
   Leccion_Aplicaciones.Fields[2].AsFloat:=0;
   Leccion_Aplicaciones.Fields[3].AsInteger:=0;

Leccion_Aplicaciones.Fields[4].AsString:='000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000';

Leccion_Aplicaciones.Fields[5].AsString:='000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000';

Leccion_Aplicaciones.Fields[6].AsString:='000000000000000000000000
```

```
00000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000';
  Leccion_Aplicaciones.post;
  Leccion_Aplicaciones.EnableControls;
  Leccion_Aplicaciones.Close;
end;
 FAplicac.Show;
end;
```

4. ARBOL DE DIRECTORIOS

La carpeta del programa fuente está en el CD, para acceder a ella se ubica desde el explorador de windows en la unidad de CD; selecciona la carpeta MrBoole ésta contiene la carpeta con el nombre ProgramaFuente, desde esa ubicación puede copiar: el código fuente del instructor en la carpeta instructor o el programa fuente del programa Mr_BooleConsultaayAdicion diseñado para alimentar la base de Datos Tablas.

El siguiente arbol se muestra la carpeta Instaladores de Mr Boole, es aquí donde está el archivo parala instalación del programa.


Explorando - INSTALADORES
Archivo Edición Ver Ir a Favoritos Herramientas Ayuda
Atrás Adelante Arriba Cortar Copiar Pegar Deshacer Eliminar Propied... Vistas
Dirección C:\INSTALADORES
Carpetas
Escritorio
MI PC
Disco de 3½ (A:)
Santo (C:)
Acrobat3
Archivos de programa
ddvf.SC8
INSTALADORES
Mis documentos
MrBoole
Adicionar
Componentes
Gráficos y Animaci
Instructor
Tablas
MrBoole1
Rig2Demo
SismiCAD 8 Demo
INSTALAD
PMrBoole.exe
Aplicación
Modificado:
14/01/01
04:54 a.m.
Tamaño:
6.541 KB
INSTRUCTOR
UFormEcMin1.dcu
ULibro.dcu
ULibroH.dcu
UOperacAB0.dcu
UOperacABAnd1.dcu
Oemsetup
Format
instalar
instalar
mini
Net10
Net7
Net8
Net9
Oemsetup.bin
extract
6,38 MB
Mi PC

MANUAL DE USUARIO

1. REQUERIMIENTOS

Para la instalación y ejecución optima de este programa se requiere un equipo compatible con Windows con una memoria RAM mínima de 16 MB. Un procesador de 133 Mhz o más, Mouse, impresora, unidad de CD-ROM, espacio en disco para su ejecución de 200 Mb. o más y tarjeta de sonido y videos.

1.1 PAQUETES REQUERIDOS

- **Windows 95/98**
- **Delphi 3.0**
- **Corel Draw**
- **Ilustrador Cómico**
- **Photo Paint**

- **Gif Animator**
- **Office 2000**
- **Internet Explorer**

2. PROCESO DE INSTALACIÓN

2.1 INSTALACION

Para instalar el programa se introduce el CD-ROM en la unidad de CD, desde el explorador acceda a la unidad de CD, escoja sucesivamente las siguientes carpetas.

- **MrBoole**
- **Instaladores**
- **Instructor**

2.2 QUITAR INSTALACION

La eliminación del programa se puede realizar desde la opción agregar o quitar programas de Window, para llegar a esta opción se hace desde el acceso directo de pantalla Mi PC, luego se hace click a la opción Panel de Control y por último la opción agregar o

quitar programas. Aparecerá una pantalla en la cual se muestra una lista de todos los programas instalados en el PC, de esta lista se selecciona el nombre de Mr Boole, hacemos click sobre el botón agregar o quitar. El sistema se encarga de eliminar el programa completamente.

2.3 EJECUTAR PROGRAMA

Para la ejecución del programa se puede hacer desde la opción I**nicio** de su PC, en la lista de programas busque el lugar donde se instala MrBoole y haga click sobre esta opción, se despliega otro menú seleccione **Instructor**, finalmente elija entre el programa Mr Boole o la Ayuda de éste.

Otra forma de ejecutar el programa es desde el explorador seleccionando las siguientes carpetas.

- **Mr Boole**
- **Instructor**
- **Mr Boole o Ayuda**

3. MANEJO DEL PROGRAMA

3.1 ICONOS MAS USUALES.

Aceptar. Da por terminada las acciones que se estén solicitando en la pantalla .

Regresar. Volver a el paso o pantalla anterior.

Continuar. Avanzar al siguiente paso o pantalla.

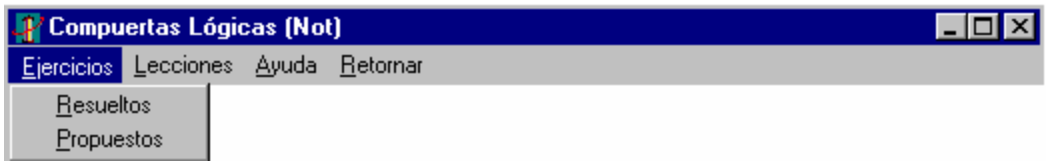
Salir. Permite salir del software.

Iniciar Reducción. Realizar la primera reducción a partir de la tabla de agrupamiento base.

Crear Tabla. Crea la siguiente tabla de reducción.

Terminar Reducción. Crea la ecuación simplificada.

3.2 **MENU PRINCIPAL.** Usted puede acceder a las siguientes opciones desde el Menú Principal (Figura 1):



- **Ejercicios:** Puede consultar una serie de ejercicios *Resueltos* de la lección que usted esté visitando o optar por ser evaluado con los ejercicios *Propuestos* de la lección.

- **Lecciones:** En cualquier instante puedes desplazarte a una de las lecciones de Mr Boole, estas son: Historia, Simplificación Algebraica, Mapas de Karnaugh, Teoría Básica, Simplificación de Quine McCluskey.

- **Ayuda:** Al seleccionar esta opción puedes obtener información *Acerca de* (Figura 2) generalidades del software y a un *Contenido* que integra una serie de ítem específicos.

- **Retornar:** Te permite abandonar a Mr Boole en cualquier instante.

Figura 2. Acerca de.

3.3 COMPLETAR TABLAS DE VERDAD.

Cuando en una lección o en una evaluación se solicita al estudiante completar o llenar una tabla de verdad como lo muestra la Figura 2, la tabla se llena utilizando los botones 0 y 1 que aparecen en la pantalla, el llenado de la tabla se realiza fila por fila de arriba hacia abajo.

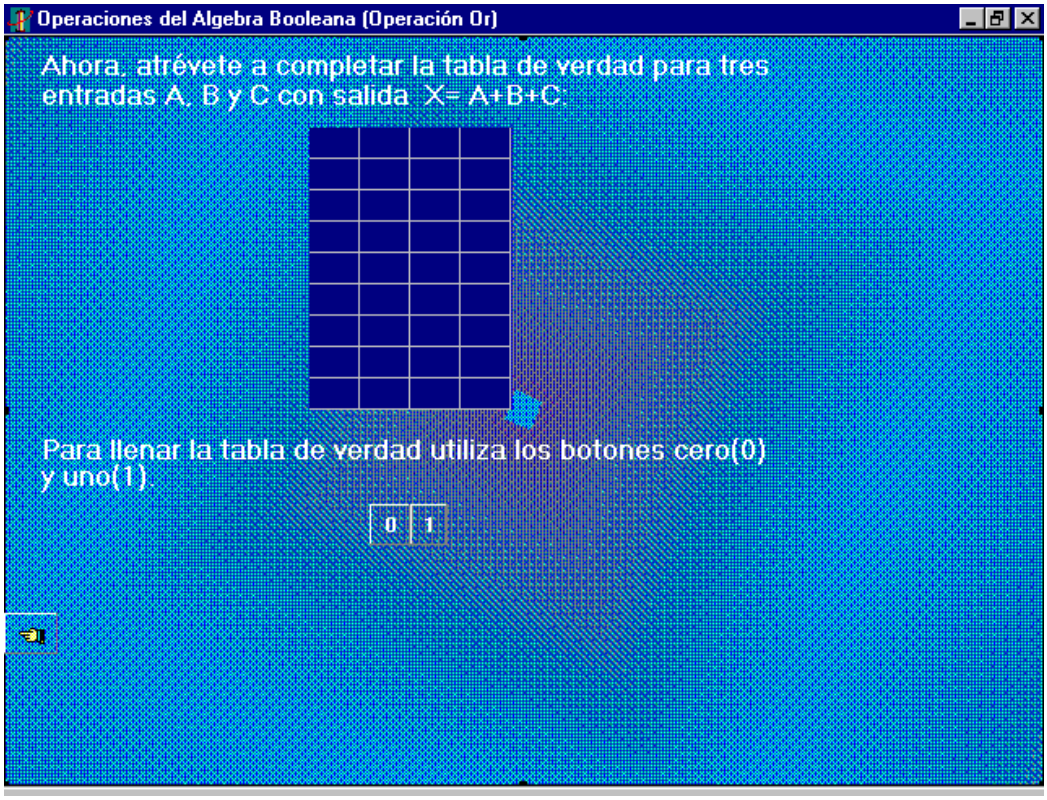


Figura 3. Completar Tablas de verdad.

3.4 FORMACION DE ECUACIONES BOOLEANA.

Al usuario se le solicita formar una ecuación booleana ya sea en la forma minterms o maxterms a partir de una tabla de verdad, el usuario debe ubicar el cursor en la casilla blanca que aparece en la parte inferior de la pantalla y digitar la fila de la cual va a sacar el termino que va a formar y dar click en el botón Aceptar, al instante aparecen los botones A,B,C, -B,-C,-D, al dar clic sobre ellos se comienza a formar la ecuación booleana.

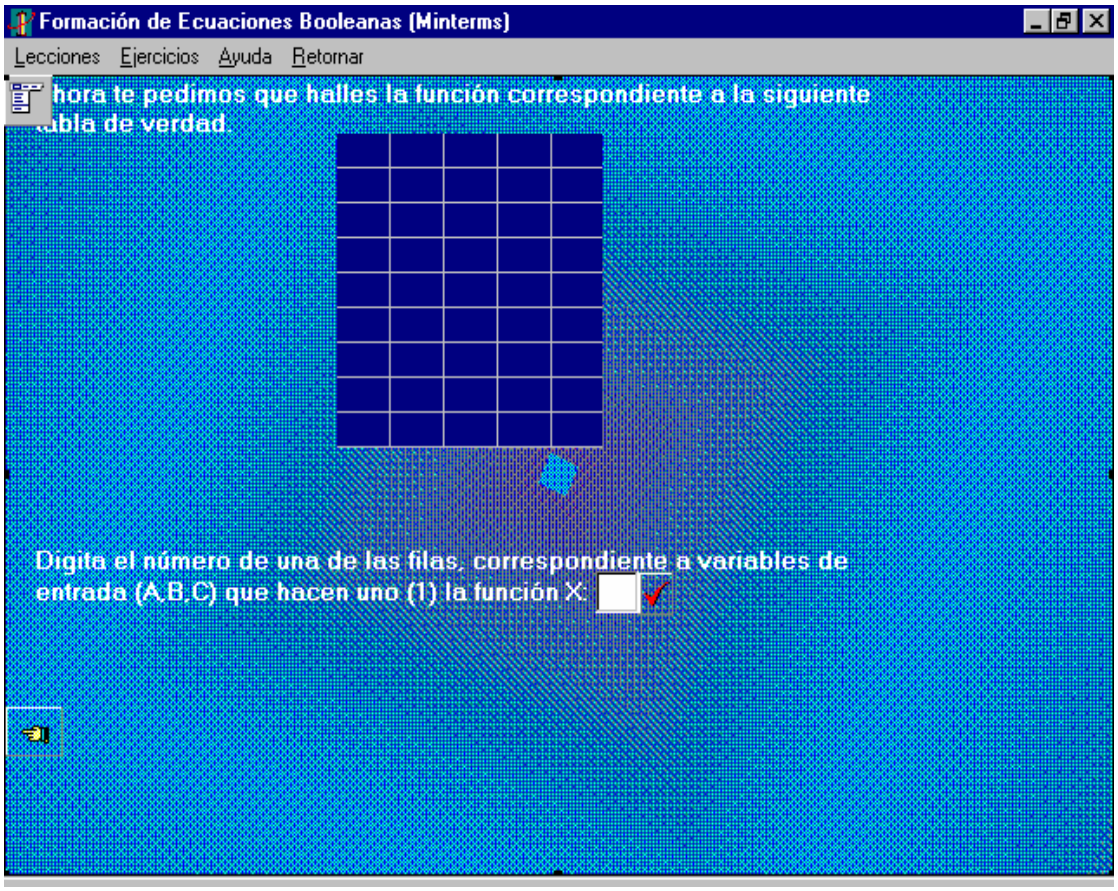


Figura 4. Formación de Ecuaciones Booleanas.

3.5 CREACION TABLA DE AGRUPAMIENTO BASE.

En la lección Simplificación de Quine McCluskey al usuario se le invita a crear la tabla de Agrupamiento debe realizar los siguientes pasos:

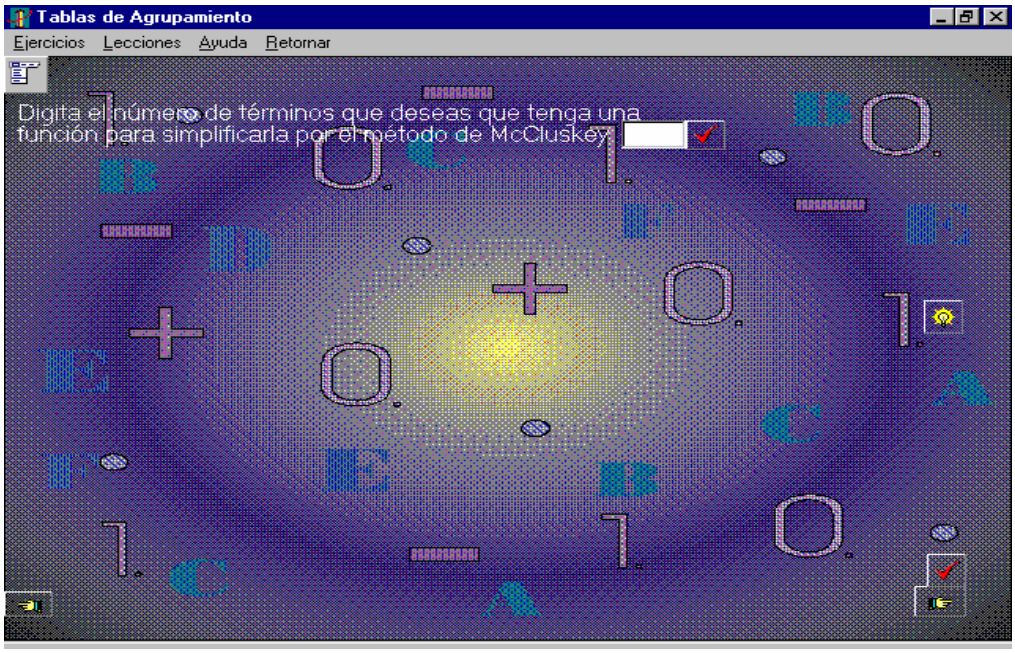


Figura 5. Numero de Términos

1. Ubicar el cursor en la casilla blanca de la parte superior, digitar el número de términos que tiene la ecuación que se va a reducir y haces click en el botón Aceptar.

2. Seleccionar las variables que vas a utilizar.

3. Para crear la primera columna de la tabla de agrupamiento base debe utilizar los botones de las variables y sus negaciones que aparecen en la parte inferior de la pantalla, cada vez que termines una casilla haces click en el botón Aceptar.

4. Para la segunda columna utiliza los botones 0’ y 1, al terminar cada casilla haces click en el botón Aceptar.

5. La tercera columna ubica el cursor en la casilla blanca digita el número según las instrucciones que se dan en la lección y haces click en el botón Aceptar.

6. Para crear la tercera columna ubica el cursor en la casilla blanca digita el número según las instrucciones que se dan en la lección y haces click en el botón Aceptar.

3.6 SIMPLIFICACION ALGEBRAICA. Tanto en la lección como en la evaluación de Simplificación Algebraica, se le solicita al usuario simplificar una ecuación Algebraica utilizando los botones que tienen los teoremas del Álgebra Booleana ubicados en la parte derecha de la pantalla como lo indica la figura 6.

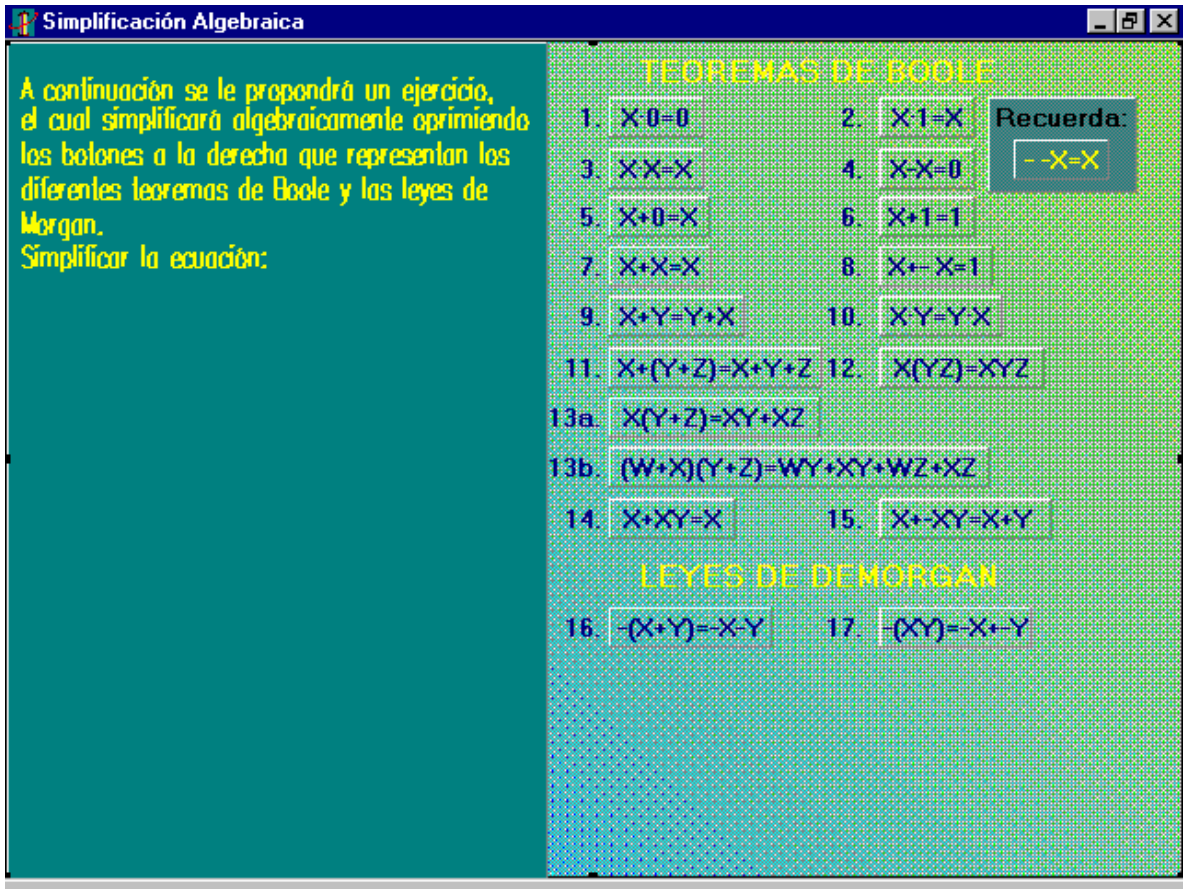


Figura 6. Simplificación Algebraica.

3.7 ELABORACION DE UN CIRCUITO DIGITAL POR EL USUARIO.

En la lección circuitos digitales el usuario tiene la oportunidad de crear un circuito digital y pedirle al software didáctico que le muestre la salida o ecuación booleana que representa este circuito. En la figura 6 es la pantalla en la que se trabaja para esto. Los pasos a seguir para el diseño del circuito son los siguientes.

Figura 7. Elaboración de un Circuito Lógico

1. En la parte superior de la pantalla observa la barra contiene todas las compuertas que puedes utilizar, pasa el mouse sobre ellas y un pequeño mensaje te indica que clase de compuerta es. Selecciona la que desees dando clic sobre ella, se crea una igual en la parte superior izquierda de la pantalla y desde ahí tu puedes arrastrarla con el mouse hasta el lugar

del lienzo donde desees que se ubique. En el ejemplo se arrastró la compuerta not. Figura 7.

2. En la parte derecha de la pantalla tienes una tabla con Entradas/Compuertas en donde están todas las entradas que puedes utilizar para cada una de las compuertas que has seleccionado, en la parte superior de la tabla tienes un botón para seleccionar la compuerta de la tabla a la que le vas asignar las entradas. Cuando seleccionas la compuerta se ilumina la que está en el lienzo. En el ejemplo se seleccionó la compuerta not. Figura 7. De igual forma selecciona en la tabla la entrada para la compuerta, en el ejemplo se seleccionó la entrada B para la compuerta not que de igual forma se iluminó.

3. Un mensaje color gris permite elegir a que posición de la entrada quieres unir la compuerta, digitas el número elegido en la casilla y das clic en el botón Aceptar. Figura 7.

4. El número de entradas de una compuerta lo puedes cambiar una sola vez dando clic derecho sobre la compuerta, seleccionas en el menú el numero de entradas que deseas que tenga la compuerta.

5. Realizas los pasos 1,2 y 3 sucesivamente hasta terminar de crear tu circuito.

6. En la barra seleccionas el botón  *“Finalizar la elaboración del circuito y calcular la salida de este”.* Figura 7.