**SMILEBASIC**

Esta basado en un lenguaje BASIC. Como se sabe basic es de los mejores lenguajes de programación para los que no saben programar y asi aprendemos a programar desde cero ¡Tan pronto como inicie SmileBASIC, puede comenzar a programar!

**INSTRUCCIONES**

**PRINT:** Instrucción en la cual se muestra impreso cualquier cadena de carácter, números, simbolos, etc entre comillas. Si se escribe 2 print y el priemro se le escribe al final ‘;’ salen impreso los 2 como uno solo pero si se le escribe ‘,’ al primero salen los dos print separados, siempre en la misma línea pero con un espacio.

EJEMPLO:

1. PRINT “hello”;
2. PRINT “world”

SALIDA:

helloworld

1. PRINT “hello”,
2. PRINT “world”

SALIDA:

hello world

**INPUT:** Instrucción en la cual se le ingresa valor a la variable durante la salida (con el teclado) en espera de un valor especifico a la variable ya sea numero o carácter y luego presiona ENTER. Si el número de elementos de entrada es insuficiente, se mostrará "Rehacer desde el inicio" para volver a ingresar

EJEMPLO:

1. INPUT “Valor de la base: ”;B
2. A = 4
3. C = B \* A
4. PRINT “El área de un triangulo es”; C

SALIDA:

Valor de la base: ? 6

El área de un triangulo es 24

**LINPUT:** Instrucción igual a INPUT con la excepción de que sólo se le ingrese una cadena de caracteres y siempre en espera hasta que se presiona ENTER

EJEMPLO:

1. LINPUT “Nombre Completo: ”; Name$
2. PRINT “Su nombre es”; Name$

SALIDA:

Nombre Completo: ? Edwin Nuñez

Su nombre es Edwin Nuñez

**GOTO:** Instrucción que hace un salto de línea de la primera hasta la ultima línea y siempre debe empezar con @ antes de los caracteres que se le escriba a lado. hará que este programa siempre regrese a la primera línea, por lo que para detenerlo, presione ENTER.

EJEMPLO:

1. @PRIMERO
2. INPUT “Valor de la base: ”;B
3. A = 4
4. C = B \* A
5. PRINT “El área de un triangulo es”; C

6 GOTO @PRIMERO

SALIDA:

Valor de la base: ? 6

El área de un triangulo es 24

Valor de la base: ? 2

El área de un triangulo es 8

Valor de la base: ?

**GOSUB:** Instrucción que llama una subrutina de acuerdo con el valor de la variable principal.

EJEMPLO:

1 GOSUB @SUB\_A, @SUB \_B   
2 PRINT "SALIR"   
3 @SUB \_A   
4 PRINT "ID = 0": RETURN  
5 @SUB \_B   
6 PRINT "ID = 1" : RETURN

**IF:** Instrucción para verificar el valor de una variable si cumple o no con la condición y ejecutar las instrucciones dentro del IF al final de la condición se coloca **THEN** para luego empezar a ejecutar la condición del IF si se cumple en caso contrario se va a la instrucción **ELSE** que se usa en caso que no se cumpla la condición IF para que cumpla entonces la alternativa. Si en cualquier otro caso no se cumple el IF y si necesitamos al mismo tiempo otra condición para verificar que se cumpla o no se usa **ELSEIF** al final de esto se usa **ENDIF** para marcar por terminado toda la condición IF.

EJEMPLO:

1. @PRIMERO
2. INPUT “Valor de la base: ”;B
3. IF B == 0 THEN
4. PRINT “continuar”;
5. ELSEIF B <= 2 THEN
6. B = B + 1
7. ELSE
8. A = 4
9. C = B \* A

10 PRINT “El área de un triangulo es”; C

11 GOTO @PRIMERO

12 ENDIF

**ON:** Instrucción que se extiende de acuerdo con el valor de la variable de control. El numero inicial seria 0. También llama subrutinas, al final se termina todo con la instrucción **WON**

EJEMPLO:

1 ON ID GOTO @SALTO\_A, @SALTO \_B   
2 PRINT "FINALIZADO"   
3 @SALTO \_A   
4 PRINT "ID = 0"  
5 @SALTO \_B   
6 PRINT "ID = 1"

7 WON

**SWITCH CASE:** instrucción que se usa como punto de control para dar multiples opciones y escoger. Al final de cada CASE se termina con un **BREAK** para terminar el ciclo de CASE en este caso, por otro lado si uno quiere que el ciclo siga se usaría **CONTINUE** al final de cada CASE.

Tanto CONTINUE como BREAK se usa también en IF FOR ... NEXT, WHILE ... WEND, REPEAT ... UNTIL. Y si ninguno de los CASES es valido se **DEFAULT** para mostrar el error**.**

EJEMPLO:

1 SWITCH (month) {

2 CASE 1: PRINT "Crear";

3 BREAK;

4 CASE 2: PRINT "Leer";

5 BREAK;

6 CASE 3: PRINT "Actualizar";

7 CONTINUE;

8 CASE 4: PRINT "Eliminar";

9 BREAK;

10 DEFAULT: PRINT "Opción Invalido";

11 BREAK;

**FOR:** Instrucción en la que se repite el mismo proceso una y otra vez dependiendo el numero de veces, usamos **TO** para definir el valor exacto del conteo del ciclo, pero si se quiere aumentar el conteo del ciclo usamos la **STEP** y para terminar el con todo el ciclo FOR usamos **NEXT**.

EJEMPLO:

1 FOR N=2 TO 0 STEP 0.25

2 PRINT N & “ ”;

3 NEXT

SALIDA: 2 1.75 1.5 1.25 1 0.75 0.5 0.25 0

**WHILE:** Instrucción que repite el proceso hasta que termine en **WEND** mientras trata de que se cumpla la condición.

EJEMPLO:

1 d = 0

2 WHILE d <= 10

3 PRINT d & “ ”

4 d = d+ 1

5 WEND

SALIDA: 1 2 3 4 5 6 7 8 9 10

**REPEAT:** Instrucción parecida al WHILE la diferencia es que ejecuta el proceso antes de determinar la condición. Sale del ciclo cuando cumple con la condición y entonces termina con la instrucción **UNTIL**.

EJEMPLO:

1 d = 0

2 REPEAT

3 PRINT d & “ ”

4 d = d+ 1

5 UNTIL d <= 10

SALIDA: 1 2 3 4 5 6 7 8 9 10

**DEF:** Define instrucciones(función) de usuario con valores que se pueden devolver y otros valores que no se devuelven. Si uno usa una instrucción unica desde la memoria se la instrucción **COMMON**.

Dentro de DEF se pueden usar otras instrucciones como **CALL** que es la instrucción definida por el usuario con el nombre especificado, ahora para devolver las instrucciones de un DEF usamos **RETURN**. Si un caso se requiere multiples valores o salidas en un DEF se usa la instrucción **OUT** y para terminar con la definición de DEF para una función o instrucción de usuario usamos **END.**

EJEMPLOS:

1 DEF FACTORIAL(N)

2 IF N == 1 THEN

3 RETURN N

4 RETURN N \* FACTORIAL(N-1)

5 END

1 CALL "USERCD",X,Y OUT A,B

2 DEF USERCD X,Y OUT A,B

3 A=X+Y:B=X\*Y

4 END

COMMON DEF RECTA(X, Y, Z)

**SWAP**: Instrucción que cambia los valores de 2 variables.

**INC:** Instrucción que incrementa el valor de una variable en +1(se agregará el valor de la expresión).

**DEC**: Instrucción que disminuye el valor de una variable en -1(se restara el valor de la expresión).

EJEMPLOS:

1 VAR i=1

2 WHILE i < 10

3 PRINT i

4 i++

5 WEND

1 VAR i=1

2 WHILE i > 10

3 PRINT i

4 i--

5 WEND

1 SWAP (VAR a, int b)

2 int temp = a

3 a = b

4 b = temp

5 END