|  |  |  |
| --- | --- | --- |
|  | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorio de docencia |

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| *Profesor:* | Rodriguez Espino Claudia |
| *Asignatura:* | Fundamentos de Programacion |
| *Grupo:* | 1102 |
| *No de Práctica(s):* | Practica numero 9 |
| *Integrante(s):* | Gutierrez Orozpe Luis Fernando |
|  |  |
|  |  |
| *Semestre:* | 2018-1 |
| *Fecha de entrega:* | 12 de Noviembre de 2017 |
| *Obervaciones:* |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_

OBJETIVOS

Elaborar programas en C para la resolución de problemas básicos que incluyan las estructuras de repetición y la directiva define.

ACTIVIDADES

Dentro de la siguiente practica se nos solicita que realicemos un programa que nos muestre las tablas de multiplicar del 1 al 10 con los 3 diferentes tipos de estructuras de repetición, While, Do While y For y posteriormente, una vez entendido el funcionamiento de cada una de estas estructuras, realizar un programa que nos muestre las potencias del 1 al 10 de cualquier numero que el usuario ingrese.

**ESTRUCTURA DE CONTROL REPETITIVA WHILE**

La estructura repetitiva (o iterativa) while primero valida la expresión lógica y si esta se cumple (es verdadera) procede a ejecutar el bloque de instrucciones de la estructura, el cual está delimitado por las llaves {}. Si la condición no se cumple continua el flujo normal del programa sin ejecutar el bloque de la estructura, es decir, el bloque se puede ejecutar de cero a ene veces. A continuación, se muestra el programa de las tablas de multiplicar con la estructura while.

**PROGRAMA 1 – TABLAS DE MULTIPLICAR CON WHILE**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <math.h>**

**#include <conio.h>**

**int main ()**

**{**

**/\*En este programa se muestran las tablas de multiplicar**

**del 1 al 10 con la estructura while\*/**

**int n=1,x=1,t;**

**while(n<=10) //La variable n es el contador del numero cuya tala se mostrara**

**{**

**printf("\n\tLa tabla del %d es: \n", n);**

**while(x<=10) /\*La variable x es el numero por el cual se multiplicara**

**el numero del cual se mostrara la tabla\*/**

**{**

**t=n\*x;**

**printf("%d \* %d = %d \n", n,x,t);**

**x++;**

**}**

**n++;**

**x=1;**

**}**

**getch();**

**}**

En el programa anterior podemos ver un ejemplo de while anidado, ya que dentro de un while existe otro while ejecutando otro bloque de instrucciones.

El primer while: **while(n<=10)** se nos muestra a la variable “n” que dentro de este programa será la encargada de llevar el contador y decirnos de que tabla de multiplicar se trata, por lo tanto, al incrementarse este contador significa que la tabla de multiplicar el número que se estaba mostrando ha terminado y pasa al siguiente número mientras éste sea menor o igual a 10.

El segundo while: **while(x<=10)** es el que se encarga de definir hasta que numero se multiplicara a la variable “n”, por lo tanto este contador es el que se encarga de ponerle el limite a la tabla de multiplicar del número “n”. Hasta que el número “n” es multiplicado por x=10 termina la multiplicación de ese número “n” y continua realizando las tablas mientras “n” sea menor o igual a 10.



Para entender mejor la explicación anterior podemos decir que el primer while es el que se encarga de mostrar el numero 1 hasta que éste se multiplica por 10, por lo tanto, el segundo while es el que se encarga de incrementar la numeración de la multiplicación del numero 1.

**ESTRUCTURA DE CONTROL REPETITIVA DO WHILE**

Do while es una estructura cíclica que ejecuta el bloque de código que se encuentra dentro de las llaves y después valida la condición, es decir, el bloque de código se ejecuta de una a ene veces. A continuación se muestra el programa de las tablas de multiplicar con la estructura do-while.

**PROGRAMA 2: TABLAS DE MULTIPLICAR CON DO WHILE**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <math.h>**

**#include <conio.h>**

**int main ()**

**{**

**/\*Este programa hace las tablas de multiplicar del 1 al 10 con la estructura**

**do while\*/**

**int n=1,x=1,i;**

**do**

**{**

**printf("\n\tLa tabla del %d es: \n", n);**

**do**

**{**

**i=n\*x;**

**printf("%d \* %d = %d \n", n,x,i);**

**x++;**

**}**

**while (x<=10); /\* La variable x es el contador del 1 al 10 para**

**multiplicar al numero cuya tabla se muestra \*/**

**n++;**

**x=1;**

**}**

**while (n<=10); /\* La variable n es el numero del cual se mostrara la tabla**

**de multiplicar\*/**

**getch();**

**}**

El programa anterior es un ejemplo de una estructura de repetición Do While anidada, quiere decir que dentro de un Do While, se encuentra otro Do While ejecutando otra condición.

En este programa a diferencia del while normal, primero ejecuta el código que hay dentro de las llaves y después evalúa la condición, por lo que en este caso, el segundo Do While se ejecutara solo si en el primer Do While se cumple con la condicion “**while (n<=10)”**



Al final de cuentas, el resultado del programa es el mismo que de haber utilizado la estructura de repetición while.

**ESTRUCTURA DE CONTROL DE REPETICION FOR**

Lenguaje C posee la estructura de repetición for la cual permite realizar repeticiones cuando se conoce el numero de elementos que se quiere recorrer. La estructura for ejecuta 3 acciones básicas antes o después de ejecutar el bloque de Codigo.

La primera acción es la inicialización, en la cual se pueden definir variables e inicializar sus valores; esta parte solo se ejecuta una vez cuando se ingresa al ciclo y es opcional. La segunda acción consta de una expresión lógica, la cual se evalua y, si ésta es verdadera, ejecuta el bloque de código, si no se cumple continua la ejecución del programa; esta parte es opcional. La tercera parte consta de un conjunto de operaciones que se realizan cada vez que termina de ejecutarse el bloque de código y antes de volver a validar la expresión lógica; esta parte también es opcional.

A continuación se anexara el código del programa de las tablas de multiplicar elaborado con la estructura de repetición for.

**PROGRAMA 3: TABLAS DE MULTIPLICAR CON FOR**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <math.h>**

**#include <conio.h>**

**int main ()**

**{**

**//Tabla de multiplicar del 1 al 10 con For**

**int x, y;**

**for(x=1;x<11;x++)**

**{**

**printf("\n");**

**printf("\tLa tabla del %d es:\n", x);**

**for(y=1;y<=10;y++)**

**{**

**printf("%d \* %d = %d\n", x, y, x\*y);**

**}**

**}**

**printf("\nPresiona cualquier tecla para continuar\n");**

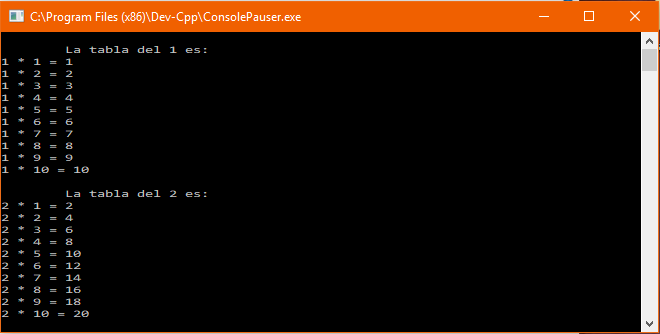
**getch();**

**}**

El programa anterior hace uso de una estructura de repetición For anidada, esto quiere decir que dentro de una repetición se encuentra otra que se ejecutara mientras la primera se cumpla.

El primer For nos indica que tabla de multiplicar se trata, por lo que debe iniciar en el número 1, debe de ser menor igual a 10 y debe incrementarse de 1 en 1.

El segundo For nos indica los números por los cuales será multiplicado el primer For, por lo que debe iniciar en el número 1, debe ser menor igual a 10 y debe incrementarse de 1 en 1, cuando este ciclo For cumple con sus condiciones, sale al ciclo For principal y se volverá a ejecutar hasta que el For principal cumpla con sus condiciones.



El primer For es el que se encarga de imprimir “La tabla del (número correspondiente al contador) es:” y el numero repetido 10 veces hasta que es multiplicado por el número 10.

El segundo For es el que se encarga de imprimir los números del 1 al 10 que serán los que multipliquen al número del primer For.

El primer For se ejecutará hasta que su contador sea menor o igual a 10, por esta razón, cuando termina de realizar la tabla de multiplicar del 1, continua con la del 2 y así sucesivamente hasta que llegue a una menor o igual a la del 10.

**PROGRAMA 4: POTENCIAS DE CUALQUIER NUMERO**

**#include <stdio.h>**

**#include <conio.h>**

**#include <math.h>**

**int main()**

**{**

**int n,x,p;**

**printf("Introduce el numero cuyas potencias del 1 al 10 quieras visualizar\n");**

**scanf("%d", &n);**

**for(x=1;x<=10;x++)**

**{**

**p=pow(n,x);**

**printf("El numero %d elevado a la potencia %d es igual a: %d\n", n,x,p);**

**}**

**printf("\nPresiona cualquier tecla para salir");**

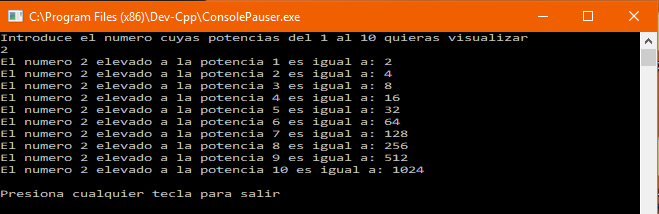
**getch();**

**}**

El programa anterior nos permite visualizar las potencias del 1 al 10 de cualquier número que el usuario introduzca en el programa.

En esta ocasión se usa el ciclo For ya que a mi parecer es la estructura de repetición que mas te facilita llevar a cabo la ejecución del programa.

Tenemos un For que nos sirve como el incremento del 1 al 10 por el cual se elevara el numero que el usuario ingrese.



Se le pide al usuario ingresar el valor que desee elevar a las potencias el 1 al 10. Una vez ingresado el valor entra en función el ciclo For que es el que nos servirá para poder incrementar el valor de la potencia conforme el programa avanza, por lo que ejecutará las potencias hasta que el valor de la variable “x” en el For sea menor o igual a 10.

CONCLUSION

Las estructuras de control repetitivas en el lenguaje C a mi parecer son esenciales para que algunos programas que nos sean solicitados funcionen de manera óptima y correcta, además de que nos ayudan a optimizar la tarea de programación, pues de no existir tendríamos que ejecutar el mismo código la cantidad de veces requerida, sin embargo, gracias a estas estructuras, solo escribimos una sola vez el código a ejecutar y la estructura se encarga de repetirlo mientras el valor de su variable incrementa.