



## Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

# Laboratorios de computación salas A y B

*Profesor:*

Ingeniera Claudia Rodriguez Espino

*Asignatura:*

Fundamentos de Programación

*Grupo:*

1102

*No de Práctica(s):*

Práctica número 9

*Integrante(s):*

Chaveste Bermejo Carlos Alberto

*Semestre:*

2018-1

*Fecha de entrega:*

20/10/2017

*Observaciones:*

**CALIFICACIÓN:** \_\_\_\_\_

## OBJETIVO:

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

## DESARROLLO:

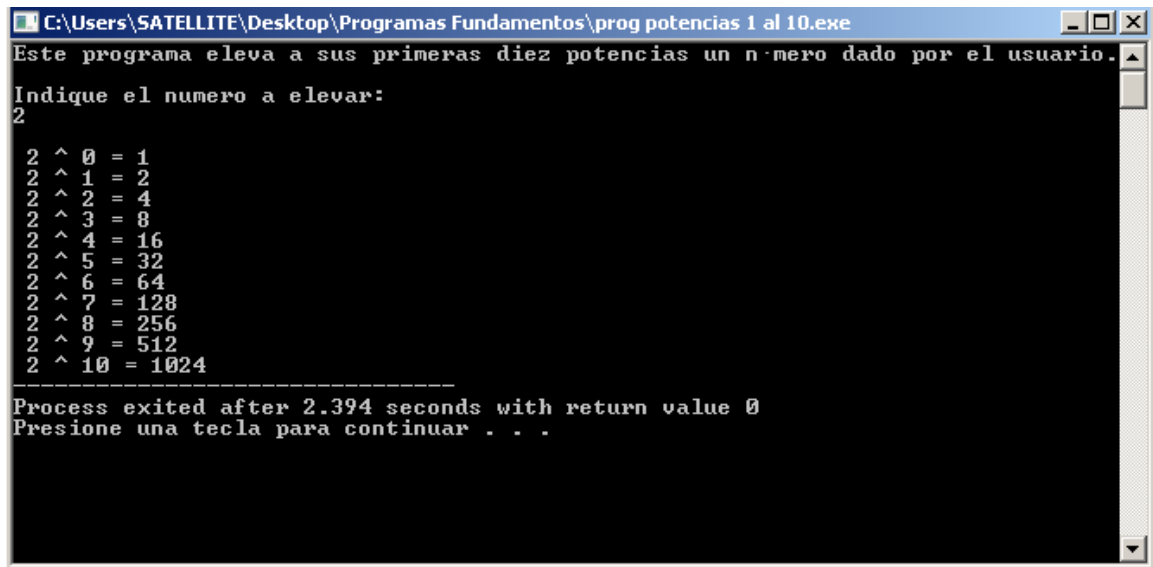
En la práctica aprendimos sobre las estructuras de repetición. Estas estructuras cumplen con una función muy específica y esencial para la programación estructurada, en C++. Constituyen la parte de este lenguaje orientada a ciclos, bucles o repeticiones, que evalúan una condición y ejecutan un proceso mientras la condición que evalúan sea verdadera. Hay 3 tipos de estructuras de repetición.

Estructura	Función
<b>for</b>	El ciclo for se utiliza para inicializar una variable, introducir la condición que se debe de cumplir y un incremento o decremento. Esta estructura es especialmente útil cuando se busca incrementar un número dentro de un parámetro específico. Su sintaxis es la siguiente: for(inicialización de variable; condición del ciclo; incremento) { bloque de instrucciones}
<b>while</b>	La estructura iterativa while repite un bloque de instrucciones mientras se cumpla la condición que se especifica, y se puede anidar una en otra, primero valida la expresión lógica luego hace el proceso, esa es la principal diferencia entre while y do-while, su sintaxis es la siguiente: while(condición) {bloque de instrucciones}
<b>do-while</b>	Esta estructura es en esencia, igual a while, con la única diferencia de que primero ejecuta el bloque de instrucciones y luego comprueba la condición, por lo que se ejecutará el bloque de instrucciones al menos una vez. Su sintaxis es: do{bloque de instrucciones} while(condición)

## ACTIVIDAD EN CLASE

### 1. Potencias de un número, del 1 al 10.

```
//Eleva a una potencia//
#include<stdio.h>
#include<math.h>
int a,b,c;
main()
{
printf("Este programa eleva a sus primeras diez potencias un número dado por el
usuario.\nIndique el numero a elevar:\n");
scanf("%d",&a);
do{
c=pow(a,b);
printf("\n %d ^ %d = %d",a,b,c);
b++;
}
while(b<11);
}
```

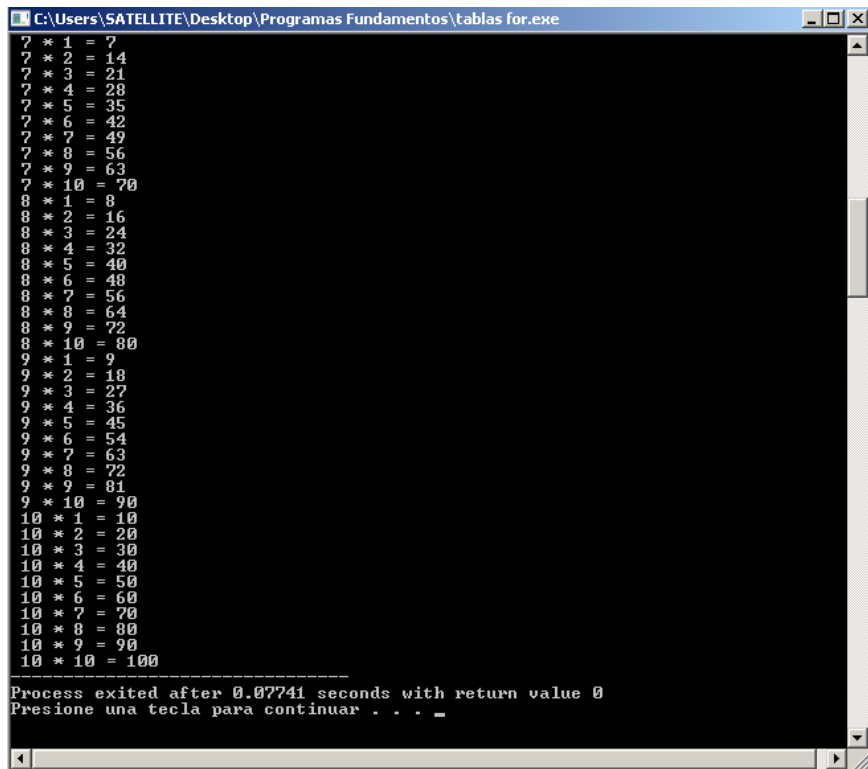


```
C:\Users\SATELLITE\Desktop\Programas Fundamentos\prog potencias 1 al 10.exe
Este programa eleva a sus primeras diez potencias un número dado por el usuario.
Indique el numero a elevar:
2
2 ^ 0 = 1
2 ^ 1 = 2
2 ^ 2 = 4
2 ^ 3 = 8
2 ^ 4 = 16
2 ^ 5 = 32
2 ^ 6 = 64
2 ^ 7 = 128
2 ^ 8 = 256
2 ^ 9 = 512
2 ^ 10 = 1024
-----
Process exited after 2.394 seconds with return value 0
Presione una tecla para continuar . . .
```

## 2. Tablas del 1 al 10 con ciclos.

### a) Ciclo for:

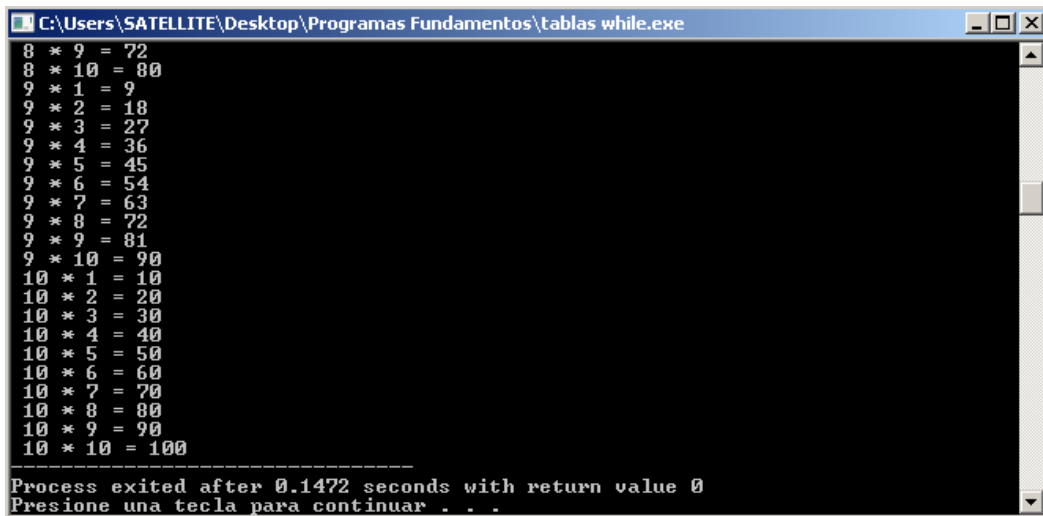
```
//Tabla de multiplicar de los primeros diez numeros for//
#include <stdio.h>
int a,b,c;
main()
{
    printf("Se imprimiran las tablas del 1 al 10\n\n");
    for (a=1;a<11;a++)
    {
        for (b=1;b<11;b++)
        {
            c=a*b;
            printf("\n %d * %d = %d",a,b,c);
            b++;
        }
    }
}
```



```
C:\Users\SATELLITE\Desktop\Programas Fundamentos\tablas for.exe
7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
7 * 6 = 42
7 * 7 = 49
7 * 8 = 56
7 * 9 = 63
7 * 10 = 70
8 * 1 = 8
8 * 2 = 16
8 * 3 = 24
8 * 4 = 32
8 * 5 = 40
8 * 6 = 48
8 * 7 = 56
8 * 8 = 64
8 * 9 = 72
8 * 10 = 80
9 * 1 = 9
9 * 2 = 18
9 * 3 = 27
9 * 4 = 36
9 * 5 = 45
9 * 6 = 54
9 * 7 = 63
9 * 8 = 72
9 * 9 = 81
9 * 10 = 90
10 * 1 = 10
10 * 2 = 20
10 * 3 = 30
10 * 4 = 40
10 * 5 = 50
10 * 6 = 60
10 * 7 = 70
10 * 8 = 80
10 * 9 = 90
10 * 10 = 100
-----
Process exited after 0.07741 seconds with return value 0
Presione una tecla para continuar . . . _
```

## b) Ciclo while

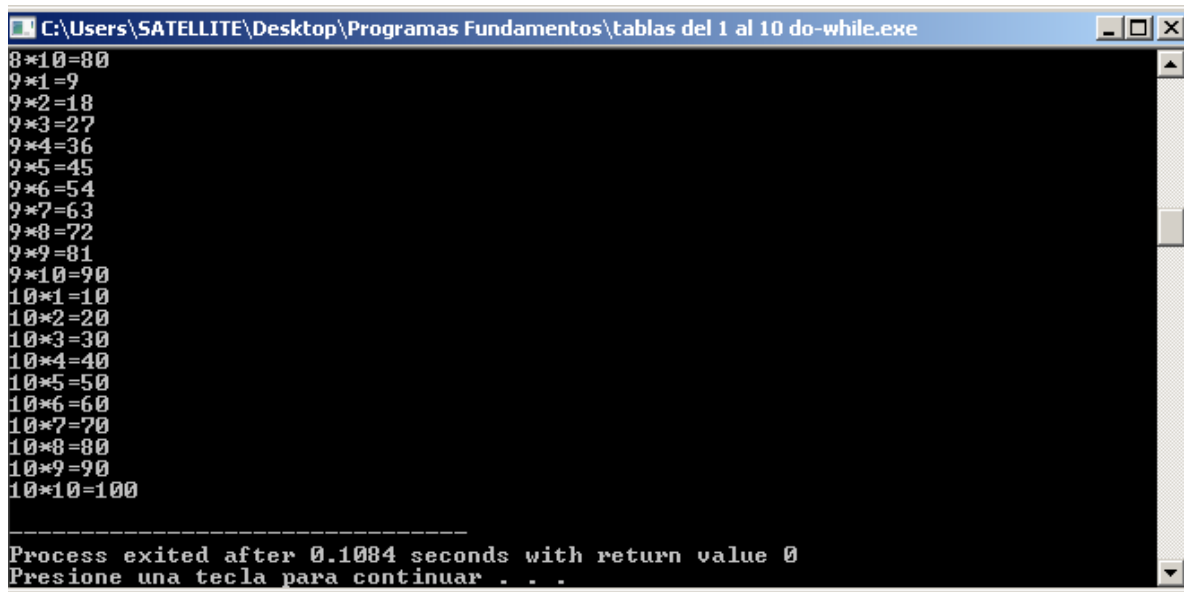
```
//Tabla de los primeros diez numeros while//
#include <stdio.h>
int a=1,b=1,c;
main()
{
    printf("Se imprimiran las tablas del 1 al 10\n\n");
    while (a<11){
        b=1;
        while(b<11)
        {
            c=a*b;
            printf("\n %d * %d = %d",a,b,c);
            b++;
        }
        a++;
    }
}
```



```
C:\Users\SATELLITE\Desktop\Programas Fundamentos\tablas while.exe
1 * 1 = 1
1 * 2 = 2
1 * 3 = 3
1 * 4 = 4
1 * 5 = 5
1 * 6 = 6
1 * 7 = 7
1 * 8 = 8
1 * 9 = 9
1 * 10 = 10
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
2 * 8 = 16
2 * 9 = 18
2 * 10 = 20
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
3 * 4 = 12
3 * 5 = 15
3 * 6 = 18
3 * 7 = 21
3 * 8 = 24
3 * 9 = 27
3 * 10 = 30
4 * 1 = 4
4 * 2 = 8
4 * 3 = 12
4 * 4 = 16
4 * 5 = 20
4 * 6 = 24
4 * 7 = 28
4 * 8 = 32
4 * 9 = 36
4 * 10 = 40
5 * 1 = 5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
5 * 8 = 40
5 * 9 = 45
5 * 10 = 50
6 * 1 = 6
6 * 2 = 12
6 * 3 = 18
6 * 4 = 24
6 * 5 = 30
6 * 6 = 36
6 * 7 = 42
6 * 8 = 48
6 * 9 = 54
6 * 10 = 60
7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
7 * 6 = 42
7 * 7 = 49
7 * 8 = 56
7 * 9 = 63
7 * 10 = 70
8 * 1 = 8
8 * 2 = 16
8 * 3 = 24
8 * 4 = 32
8 * 5 = 40
8 * 6 = 48
8 * 7 = 56
8 * 8 = 64
8 * 9 = 72
8 * 10 = 80
9 * 1 = 9
9 * 2 = 18
9 * 3 = 27
9 * 4 = 36
9 * 5 = 45
9 * 6 = 54
9 * 7 = 63
9 * 8 = 72
9 * 9 = 81
9 * 10 = 90
10 * 1 = 10
10 * 2 = 20
10 * 3 = 30
10 * 4 = 40
10 * 5 = 50
10 * 6 = 60
10 * 7 = 70
10 * 8 = 80
10 * 9 = 90
10 * 10 = 100
-----
Process exited after 0.1472 seconds with return value 0
Presione una tecla para continuar . . .
```

## c)Ciclo do-while.

```
//Tabla de los primeros diez numeros do-while//
#include<stdio.h>
int a=1,b,cont=1;
main()
{
printf("Tabla de multiplicar de los primeros diez numeros.\n\t\n");
do{
    do{
        b=a*cont;
        printf("%d*%d=%d\n",a,cont,b);
        cont++;
    }
    while(cont<11);
    cont=1;
    a++;
}
while(a<11);
}
```



```
C:\Users\SATELLITE\Desktop\Programas Fundamentos\tablas del 1 al 10 do-while.exe
8*10=80
9*1=9
9*2=18
9*3=27
9*4=36
9*5=45
9*6=54
9*7=63
9*8=72
9*9=81
9*10=90
10*1=10
10*2=20
10*3=30
10*4=40
10*5=50
10*6=60
10*7=70
10*8=80
10*9=90
10*10=100

-----
Process exited after 0.1084 seconds with return value 0
Presione una tecla para continuar . . .
```

## **CONCLUSIÓN**

Las estructuras de iterativas requieren de conocer el programa y cuando inicializar variables, cuando incrementarlas y cuando resetearlas, en el ciclo while, se puede ver esto fácilmente, ya que a la hora de anidar un proceso en otro, se requiere restear la variable. Estas estructuras pueden ser muy útiles cuando se tienen manejo de ellas. Ahorran muchas líneas de código y tiempo, de igual manera.

## **BIBLIOGRAFÍA**

<http://lcp02.fi-b.unam.mx/#>