



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: Ing. Claudia Rodriguez Espino

Asignatura: Fundamentos de Programación

Grupo: 1102

No de Práctica(s): Práctica número 9

Integrante(s): Alejandro Nava Cruz

Semestre: 2018-1

Fecha de entrega: 23 de Octubre del 2017

Observaciones:

CALIFICACIÓN: _____

Practica 9: Estructuras de repetición

Objetivos:

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

Desarrollo:

Las estructuras de repetición o cíclicas son estructuras de los lenguajes de programación (No solo en C/C++), las cuales permiten ejecutar un conjunto de instructivos u ordenes de una manera repetitiva o cíclica, tomando en cuenta la condición, esto es, mientras la expresión lógica a evaluar sea verdadera o se cumpla.

En el lenguaje de C existen 3 tipos “distintos” de estructuras de repetición, no del todo diferentes, pero que pueden llegar a acoplarse de una mejor manera con el usuario o programador, estos distintos tipos son:

While, Do While y For.

While: Esta estructura lo que hace primero es validar la expresión lógica, para después ejecutar las instrucciones, delimitado por las ya conocidas llaves (`{}`). Si esta condición no se llegara a cumplir, se continua con el flujo normal del programa. Cabe recalcar que el proceso se puede repetir desde 0 a ene veces.

Do While: esta estructura es muy parecida a la anterior, a diferencia de que esta primero ejecuta las instrucciones dentro del campo para después evaluar la condición, de la misma manera, puede ejecutarse desde 0 a ene veces.

For: La estructura for funciona casi de la misma manera, a diferencia de que esta realiza repeticiones cuando se conoce el número de elementos a los que se quiere llegar.

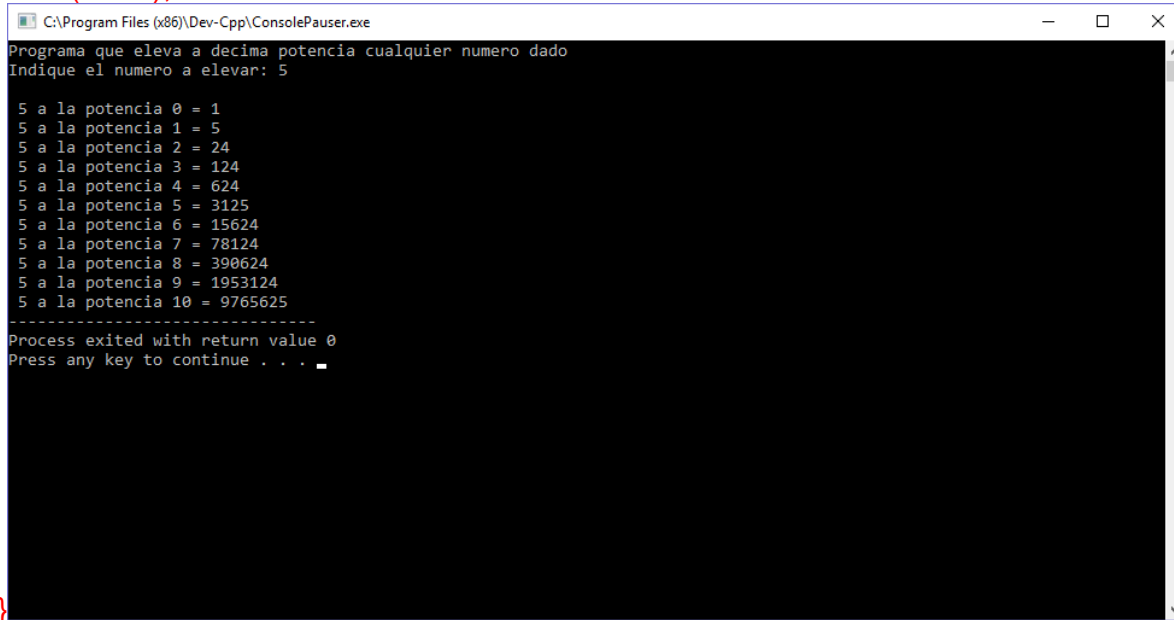
Ejercicios hechos en casa

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

```
#include<stdio.h>
#include<math.h>
```

```
int a,b,c;
```

```
main()
{
printf("Programa que eleva a decima potencia cualquier numero dado\n");
printf("Indique el numero a elevar: ");
scanf("%d",&a);
do{
c=pow(a,b);
printf("\n %d a la potencia %d = %d",a,b,c);
b++;
}
while(b<11);
```



```
C:\Program Files (x86)\Dev-Cpp\ConsolePauser.exe
Programa que eleva a decima potencia cualquier numero dado
Indique el numero a elevar: 5

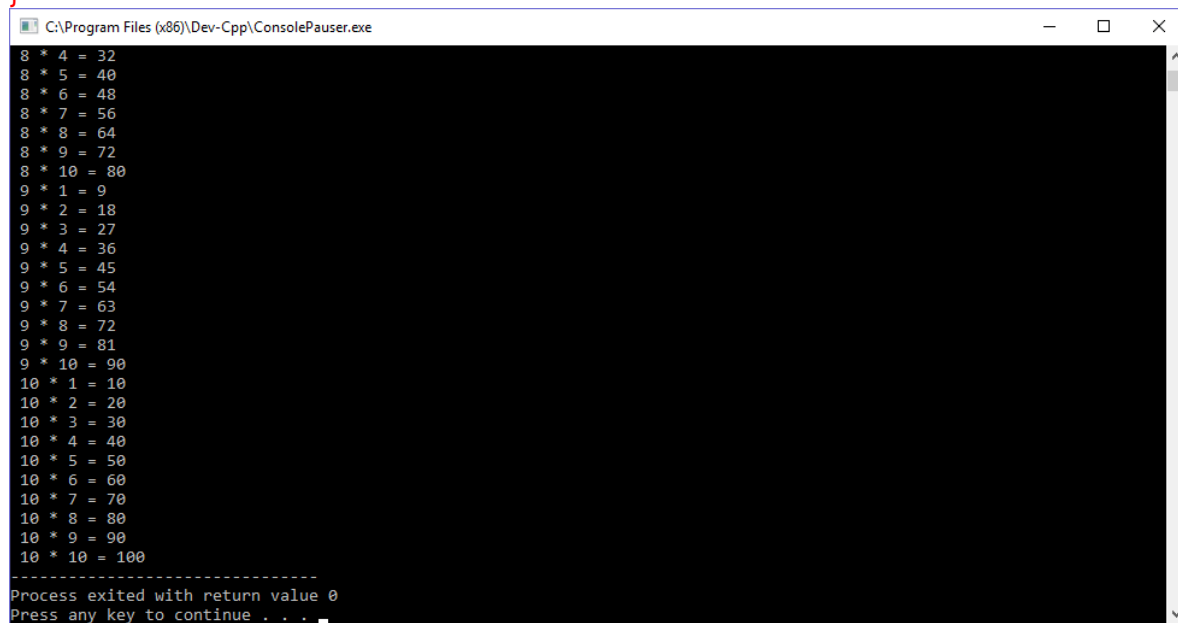
5 a la potencia 0 = 1
5 a la potencia 1 = 5
5 a la potencia 2 = 24
5 a la potencia 3 = 124
5 a la potencia 4 = 624
5 a la potencia 5 = 3125
5 a la potencia 6 = 15624
5 a la potencia 7 = 78124
5 a la potencia 8 = 390624
5 a la potencia 9 = 1953124
5 a la potencia 10 = 9765625
-----
Process exited with return value 0
Press any key to continue . . .
```

2. Tablas de multiplicar del 1 al 10 con el uso del ciclo "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;)
{
c=a*b;
printf("\n %d * %d = %d",a,b,c);
b++;
}
}
}
```



```
C:\Program Files (x86)\Dev-Cpp\ConsolePauser.exe
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 with return value 0
Press any key to continue . . .
```

3. Tablas de multiplicar del 1 al 10 con uso de ciclo "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;
```

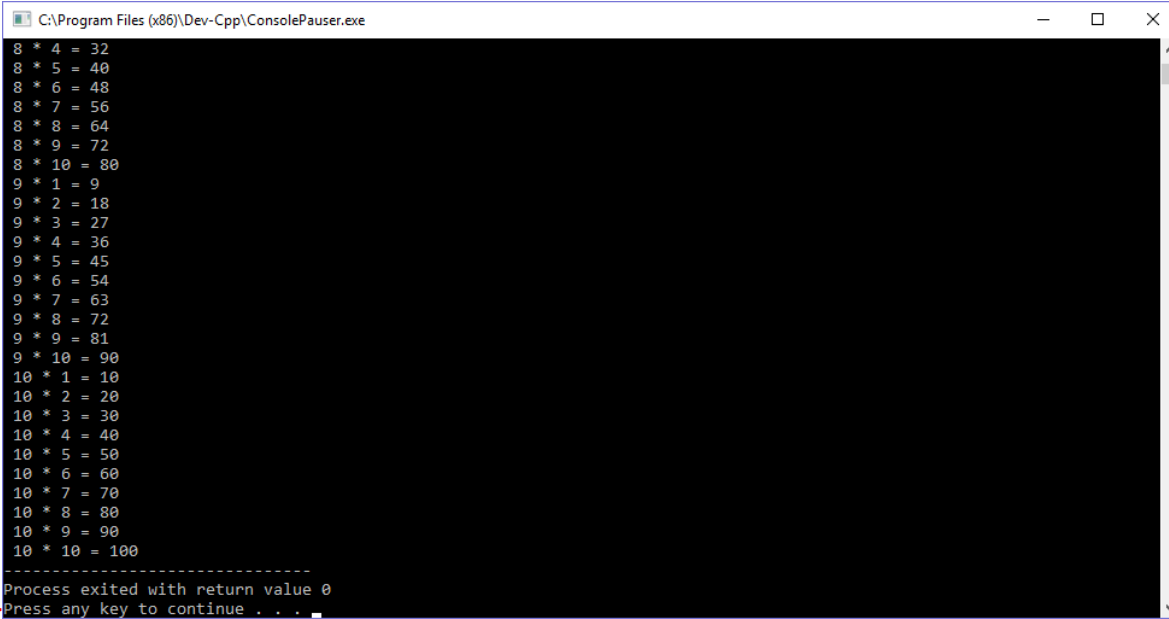
```
printf("\n %d * %d = %d",a,b,c);
```

```
b++;
```

```
}
```

```
a++;
```

```
}
```



The screenshot shows a console window titled "C:\Program Files (x86)\Dev-Cpp\ConsolePauser.exe". The output displays multiplication tables for numbers 8 through 10. Each table consists of 10 lines, showing the product of the number and integers from 1 to 10. The tables are separated by blank lines. At the bottom, a message indicates the process exited with return value 0 and prompts the user to press any key to continue.

```
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 with return value 0
Press any key to continue . . .
```

4. Tablas de multiplicar del 1 al 10 con uso del ciclo "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++;
```

```
}
```

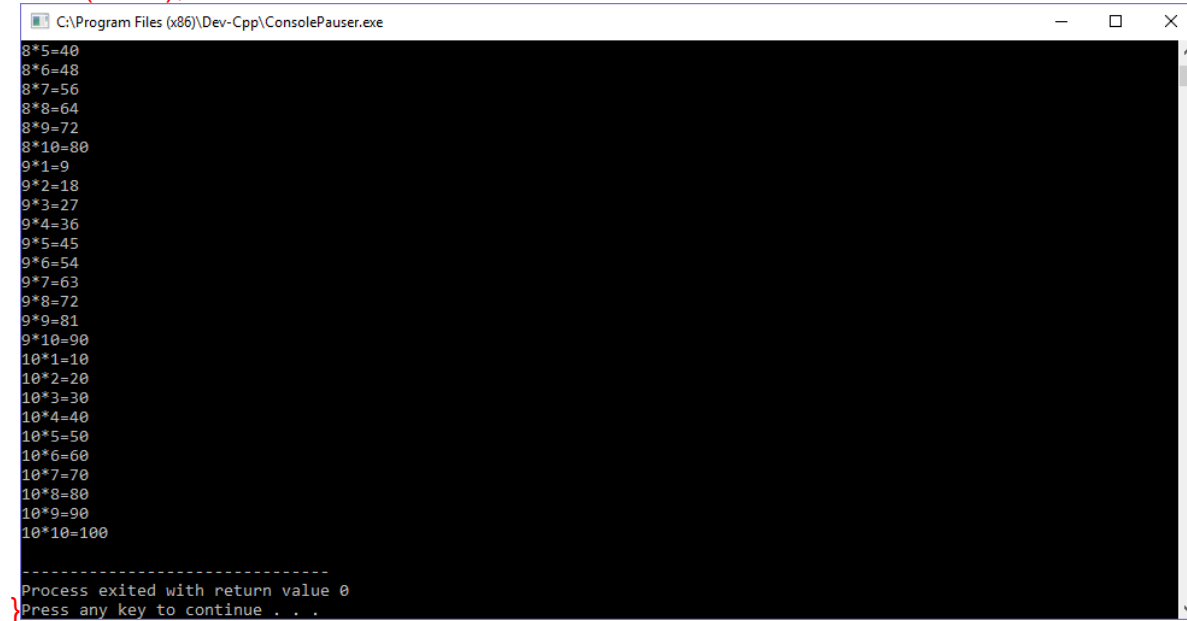
```
while(cont<11);
```

```
cont=1;
```

```
a++;
```

```
}
```

```
while(a<11);
```



The screenshot shows a console window titled "C:\Program Files (x86)\Dev-Cpp\ConsolePauser.exe". The output displays multiplication tables for numbers 8 through 10. Each table consists of 10 lines, showing the number multiplied by integers from 1 to 10. The tables are separated by a horizontal line. At the bottom, the text "Process exited with return value 0" and "Press any key to continue . . ." are visible.

```
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 with return value 0  
Press any key to continue . . .
```

Conclusiones:

Las estructuras de repetición son de gran ayuda para evitar escribir un proceso que se puede llegar a repetir, pero paso a paso, estas se requieren de conocer de una manera clara la manera de iniciar las variables, saber cuándo incrementarlas y cuando repetir o ciclar el proceso. Estas estructuras son útiles para ahorrar muchas líneas de código y tiempo.

Bibliografía:

Práctica número 9, Facultad de Ingeniería, en Laboratorios A y B (2017), Sitio Web:
<http://lcp02.fib.unam.mx/>