



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor:

ALEJANDRO ESTEBAN PIMENTEL ALARCON

Asignatura:

FUNDAMENTOS DE PROGRAMACIÓN

Grupo:

3

No de Práctica(s):

9

Integrante(s):

MIRANDA GUTIERREZ CELINE

*No. de Equipo de
cómputo empleado:*

39

No. de Lista o Brigada:

30

Semestre:

2020-1

Fecha de entrega:

14/10/2019

Observaciones:

Tus programas funcionan bien, pero no cumples
con el objetivo de utilizar los tres tipos de ciclo

PRÁCTICA 9 “Estructuras de repetición”

INTRODUCCIÓN

Las estructuras de repetición, lazos, ciclos o bucles se utilizan para realizar un proceso repetidas veces. El código incluido entre las llaves { } (opcionales si el proceso repetitivo consta de una sola línea), se ejecutará mientras se cumplan determinadas condiciones. Hay que prestar especial atención a los ciclos infinitos, hecho que ocurre cuando la condición de finalizar el ciclo no se llega a cumplir nunca. Se trata de un fallo muy típico, habitual sobre todo entre programadores poco experimentados.

while

La estructura de repetición While ejecuta un ciclo que se repetirá mientras que la condición sea verdadera. **Sintaxis** while (<condición>) **Ejemplo:** while (a <= 10)

do/while

La estructura do/while es similar a la estructura while. En la while, la condición de continuación de **ciclo** se prueba al principio del **ciclo**, antes de ejecutarse el cuerpo del mismo. La estructura do/while prueba la condición de continuación del ciclo repetitivo, después de ejecutar el cuerpo del ciclo, por lo tanto, el cuerpo del ciclo repetitivo se ejecutará por lo menos una vez.

Cuando termina do/while, la ejecución continuará con el enunciado que aparezca después de la cláusula while. No es necesario utilizar llaves

Sintaxis do Sentencias while (<condicion>); **Ejemplo** do { cout <<"entre la nota"; cin>>nota; i++; } while(i <=10);

for

La estructura de repetición for manera de manera automática todos los detalles de la repetición controlada por contador. **Sintaxis** for (<inicio;final;contador>) **Ejemplo:** for (int i = 0; i <= 10; i++) cout <<"hola";

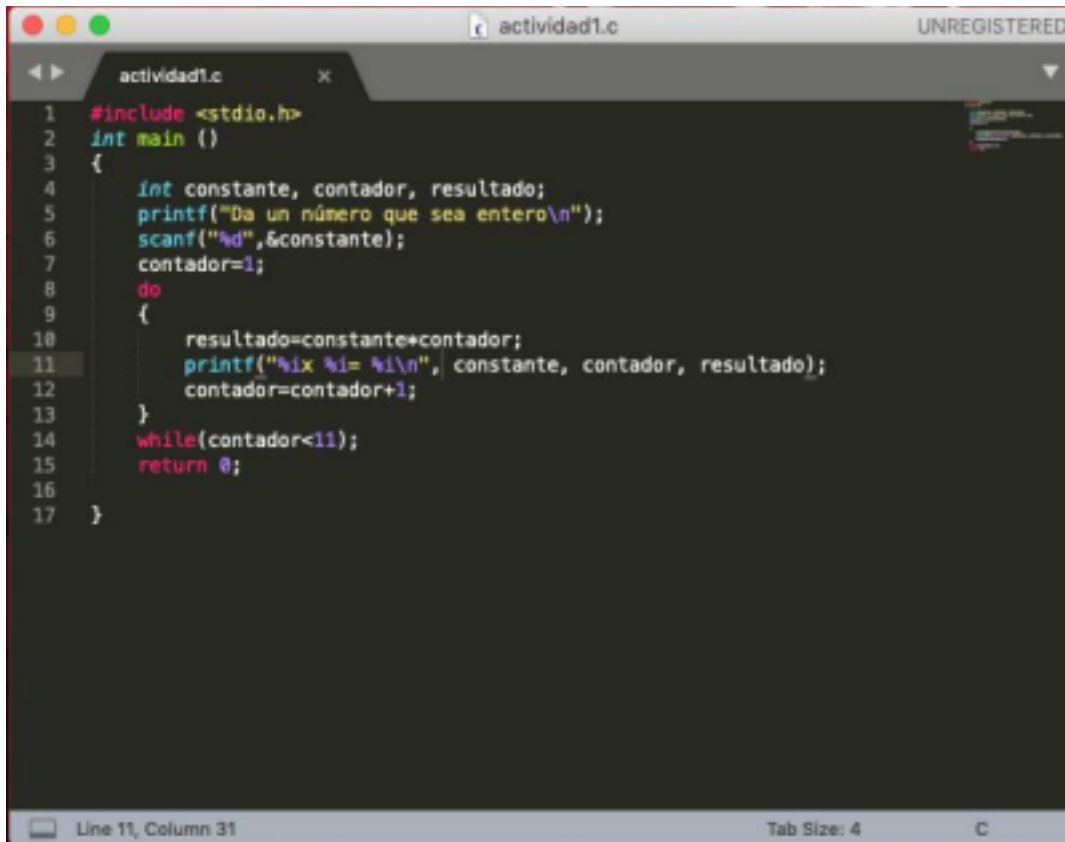
OBJETIVO

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

ACTIVIDADES

Para cada uno de los siguientes problemas, elegir un tipo de ciclo y resolverlo. Al final, deben usar los tres tipos de ciclos y usar define por lo menos una vez.

1. Hacer un programa que pida un número y muestre su tabla de multiplicar (hasta el 10)



```
1  #include <stdio.h>
2  int main ()
3  {
4      int constante, contador, resultado;
5      printf("Da un número que sea entero\n");
6      scanf("%d",&constante);
7      contador=1;
8      do
9      {
10         resultado=constante*contador;
11         printf("%ix %i= %i\n", constante, contador, resultado);
12         contador=contador+1;
13     }
14     while(contador<11);
15     return 0;
16
17 }
```



```
fp03alu30 — -bash — 80x24
Last login: Mon Oct  7 10:17:20 on ttys000
Portugal36:~ fp03alu30$ ls
Desktop      Library      Pictures      ejecutar
Documents    Movies       Public        ejecutar1
Downloads    Music        actividad1.c
Portugal36:~ fp03alu30$ gcc actividad1.c -o main
Portugal36:~ fp03alu30$ ./main
Da un número que sea entero
1
1x 1= 1
1x 2= 2
1x 3= 3
1x 4= 4
1x 5= 5
1x 6= 6
1x 7= 7
1x 8= 8
1x 9= 9
1x 10= 10
Portugal36:~ fp03alu30$
```

```
fp03alu30 — -bash — 80x24
5x 1= 5
5x 2= 10
5x 3= 15
5x 4= 20
5x 5= 25
5x 6= 30
5x 7= 35
5x 8= 40
5x 9= 45
5x 10= 50
Portugal36:~ fp03alu30$ ./main
Da un número que sea entero
7
7x 1= 7
7x 2= 14
7x 3= 21
7x 4= 28
7x 5= 35
7x 6= 42
7x 7= 49
7x 8= 56
7x 9= 63
7x 10= 70
Portugal36:~ fp03alu30$
```

2. Hacer un programa que pida y lea 10 números y muestre su suma y promedio.

```
actividad2.c
UNREGISTERED

1  #include <stdio.h>
2  #define Celine 10
3  int main()
4  {
5      int promedio = 0, numero, suma;
6
7      for (int i=1; i<=10; i++)
8      {
9          printf("Ingrese un numero %d:", i);
10         scanf("%d", &numero);
11         promedio += numero;
12     }
13     suma=suma+numero;
14     printf("\nLa suma de tus numeros es: %d", promedio);
15
16     promedio/=Celine;
17     printf("\nEl promedio es: %d\n", promedio);
18
19     return 0;
20 }
21
```

Line 16, Column 15 Tab Size: 4 C

```
alinemiranda — -bash — 80x24
Last login: Mon Oct 14 18:37:05 on ttys000
[MacBook-Air-de-Aline:~ alinemiranda$ ls
Creative Cloud Files      Library                actividad2
Desktop                  Movies                 actividad2.c
Documents                Music                  main.c
Downloads                Pictures
FP_2020-1_9144           Public
[MacBook-Air-de-Aline:~ alinemiranda$ gcc actividad2.c -o actividad2
[MacBook-Air-de-Aline:~ alinemiranda$ ./actividad2
Ingrese un numero 1:90
Ingrese un numero 2:36
Ingrese un numero 3:74
Ingrese un numero 4:21
Ingrese un numero 5:33
Ingrese un numero 6:80
Ingrese un numero 7:16
Ingrese un numero 8:17
Ingrese un numero 9:32
Ingrese un numero 10:11

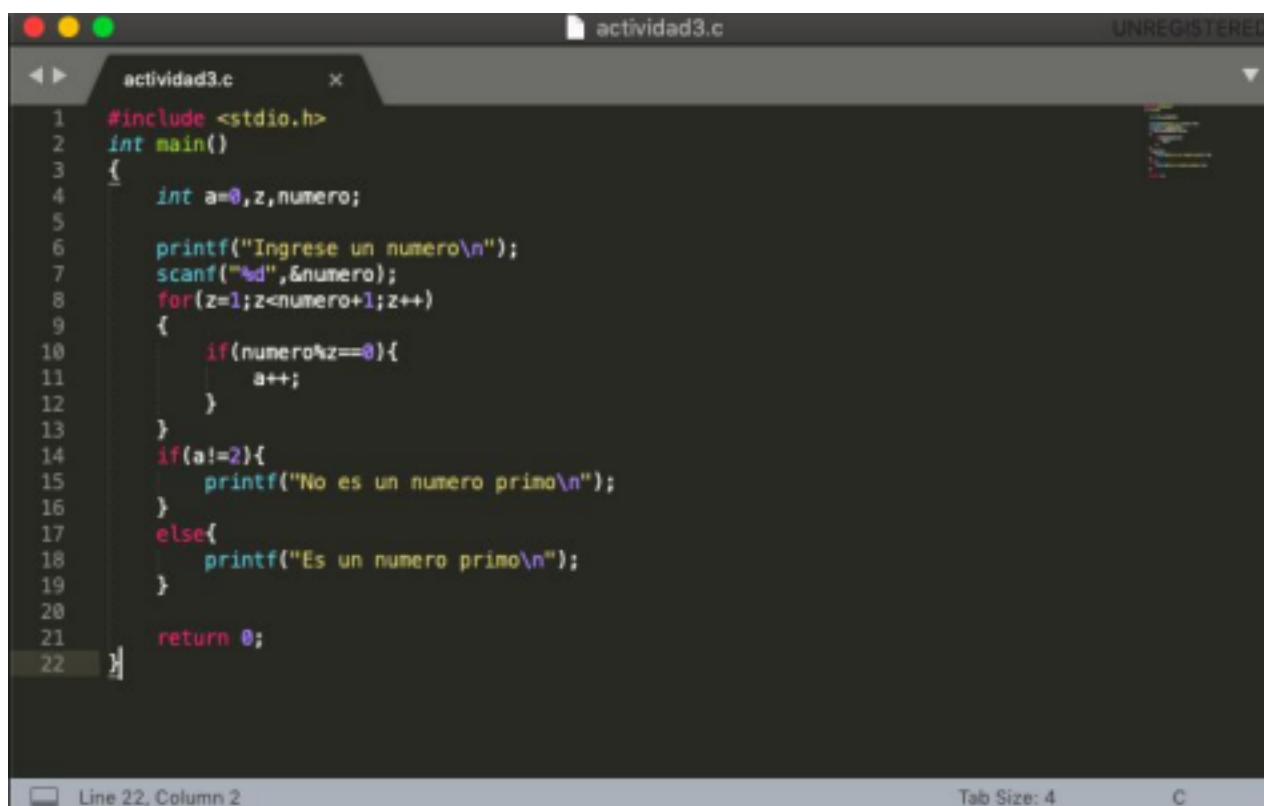
La suma de tus numeros es: 410
El promedio es: 41
[MacBook-Air-de-Aline:~ alinemiranda$ ./actividad2
Ingrese un numero 1:77
```

```
alinemiranda — -bash — 80x24
Ingrese un numero 5:33
Ingrese un numero 6:80
Ingrese un numero 7:16
Ingrese un numero 8:17
Ingrese un numero 9:32
Ingrese un numero 10:11

La suma de tus numeros es: 410
El promedio es: 41
[MacBook-Air-de-Aline:~ alinemiranda$ ./actividad2
Ingrese un numero 1:77
Ingrese un numero 2:43
Ingrese un numero 3:10
Ingrese un numero 4:20
Ingrese un numero 5:32
Ingrese un numero 6:76
Ingrese un numero 7:98
Ingrese un numero 8:76
Ingrese un numero 9:45
Ingrese un numero 10:30

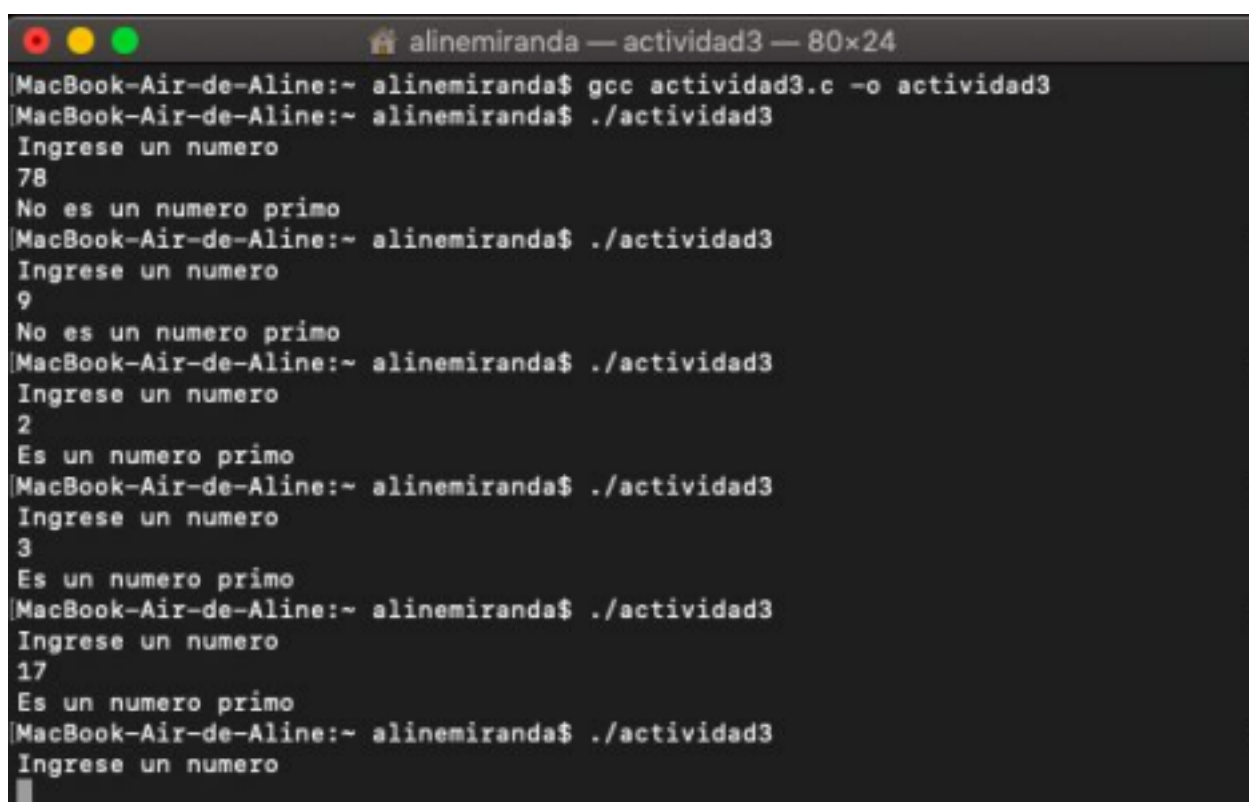
La suma de tus numeros es: 507
El promedio es: 50
[MacBook-Air-de-Aline:~ alinemiranda$
```

3. Hacer un programa que pida un número e indique si es primo o no.



```
1  #include <stdio.h>
2  int main()
3  {
4      int a=0,z,numero;
5
6      printf("Ingrese un numero\n");
7      scanf("%d",&numero);
8      for(z=1;z<numero+1;z++)
9      {
10         if(numero%z==0){
11             a++;
12         }
13     }
14     if(a!=2){
15         printf("No es un numero primo\n");
16     }
17     else{
18         printf("Es un numero primo\n");
19     }
20
21     return 0;
22 }
```

Line 22, Column 2 Tab Size: 4 C



```
MacBook-Air-de-Aline:~ alinemiranda$ gcc actividad3.c -o actividad3
MacBook-Air-de-Aline:~ alinemiranda$ ./actividad3
Ingrese un numero
78
No es un numero primo
MacBook-Air-de-Aline:~ alinemiranda$ ./actividad3
Ingrese un numero
9
No es un numero primo
MacBook-Air-de-Aline:~ alinemiranda$ ./actividad3
Ingrese un numero
2
Es un numero primo
MacBook-Air-de-Aline:~ alinemiranda$ ./actividad3
Ingrese un numero
3
Es un numero primo
MacBook-Air-de-Aline:~ alinemiranda$ ./actividad3
Ingrese un numero
17
Es un numero primo
MacBook-Air-de-Aline:~ alinemiranda$ ./actividad3
Ingrese un numero
```

CONCLUSIONES

Con esta práctica podemos concluir que las estructuras de repetición son una de las funciones más importantes para realizar tareas repetitivas de manera rápida y eficiente. Es una herramienta que facilita la ejecución de programas y sin ellas podría ser tedioso y lentas para los seres humanos, es decir, la velocidad con la que un programa ejecuta las acciones no se compara con la velocidad que podría alcanzar una persona por mucha practica que esta tenga.

Por este motivo es de mucha importancia saber utilizar estos tres ciclos para que nuestros programas trabajen con mayor eficacia.