



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Alejandro Esteban Pimentel Alarcon

Profesor:

Fundamentos de programación

Asignatura:

3

Grupo:

9

No de Práctica(s):

Oscar García García

Integrante(s):

*No. de Equipo de
cómputo empleado:*

2712

No. de Lista o Brigada:

2020-1

Semestre:

14/10/19

Fecha de entrega:

Observaciones:

CALIFICACIÓN: _____

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

WHILE

```
while (expresión_lógica) {  
    // Bloque de código a repetir  
    // mientras que la expresión  
    // lógica sea verdadera.  
}
```

DO-WHILE

```
do {  
    /*  
    Bloque de código que se ejecuta  
    por lo menos una vez y se repite  
    mientras la expresión lógica sea  
    verdadera.  
    */  
} while (expresión_lógica);
```

FOR

```
for (inicialización ; expresión_lógica ; operaciones por iteración) {  
    /*  
    Bloque de código  
    a ejecutar  
    */  
}
```

DEFINE

El *define* es una palabra clave que se utiliza para declarar un nombre especial con un significado. Es muy parecido a una variable, con la diferencia de que no se puede cambiar a lo largo del programa.

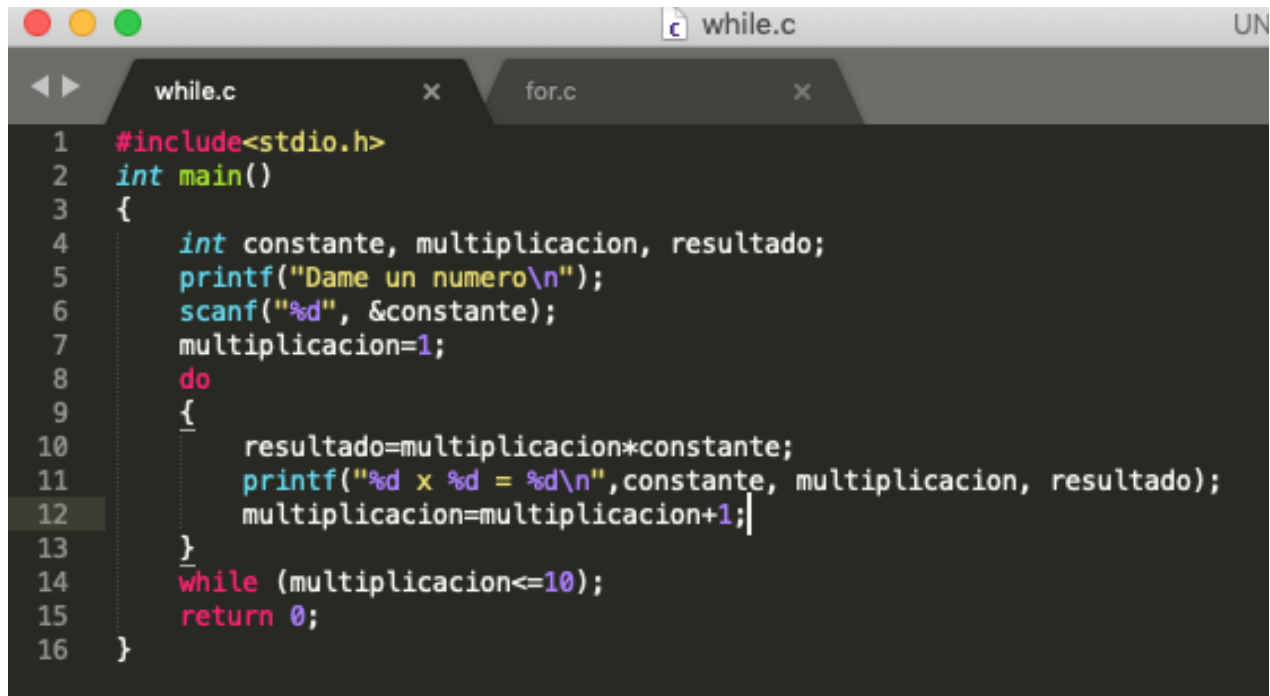
```
#define MAX 5
```

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.

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

Decidí usar DO-WHILE para este programa:

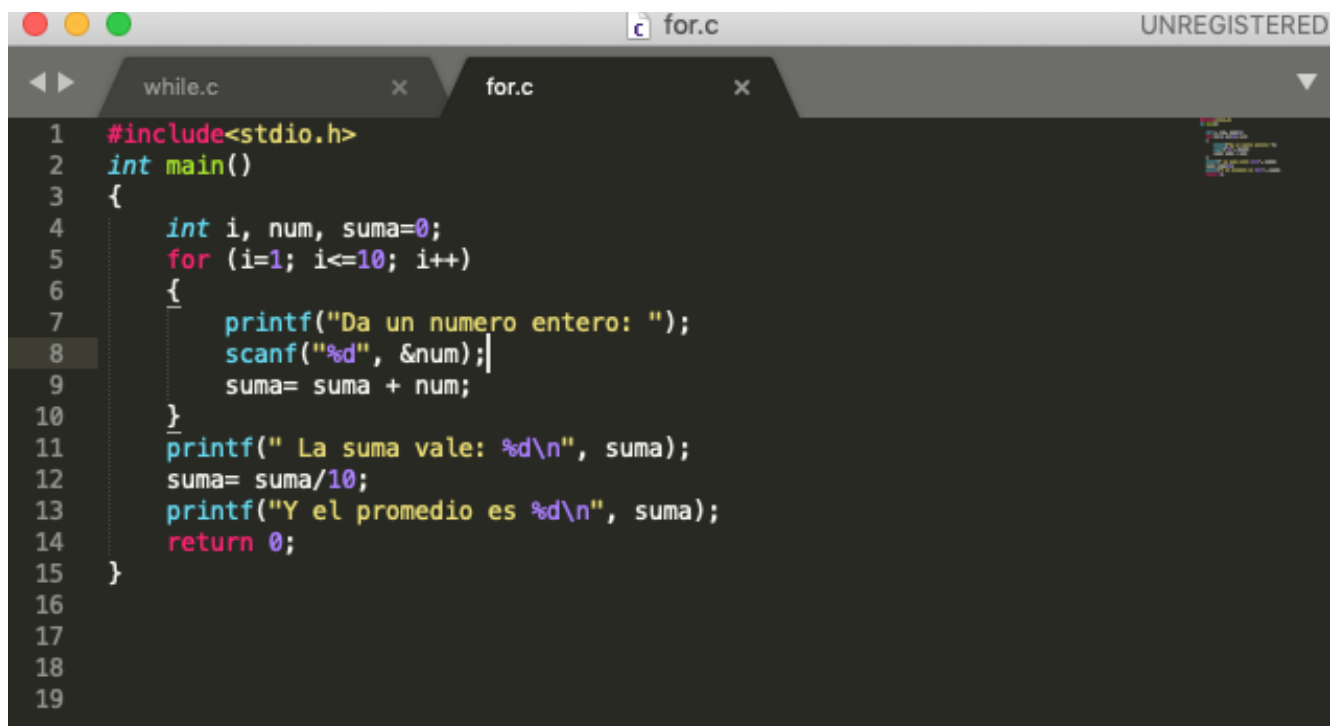


```
1  #include<stdio.h>
2  int main()
3  {
4      int constante, multiplicacion, resultado;
5      printf("Dame un numero\n");
6      scanf("%d", &constante);
7      multiplicacion=1;
8      do
9      {
10         resultado=multiplicacion*constante;
11         printf("%d x %d = %d\n",constante, multiplicacion, resultado);
12         multiplicacion=multiplicacion+1;
13     }
14     while (multiplicacion<=10);
15     return 0;
16 }
```

```
[Ruanda37:Documents fp03alu15$ gcc while.c -o main
[Ruanda37:Documents fp03alu15$ ./main
Dame un numero
5
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
```

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

Decidí usar FOR para este programa:



The screenshot shows a code editor window with two tabs: 'while.c' and 'for.c'. The 'for.c' tab is active, displaying a C program. The program includes `<stdio.h>` and defines a `main()` function. Inside `main()`, it declares `int i, num, suma=0;`. A `for` loop runs from `i=1` to `i=10`. In each iteration, it prompts the user for an integer (`printf("Da un numero entero: ");`), reads the input (`scanf("%d", &num);`), and adds it to the sum (`suma = suma + num;`). After the loop, it prints the total sum (`printf(" La suma vale: %d\n", suma);`), calculates the average (`suma = suma/10;`), and prints the average (`printf("Y el promedio es %d\n", suma);`). Finally, it returns 0.

```
1  #include<stdio.h>
2  int main()
3  {
4      int i, num, suma=0;
5      for (i=1; i<=10; i++)
6      {
7          printf("Da un numero entero: ");
8          scanf("%d", &num);
9          suma= suma + num;
10     }
11     printf(" La suma vale: %d\n", suma);
12     suma= suma/10;
13     printf("Y el promedio es %d\n", suma);
14     return 0;
15 }
16
17
18
19
```

```
[Ruanda37:Documents fp03alu15$ gcc for.c -o main
```

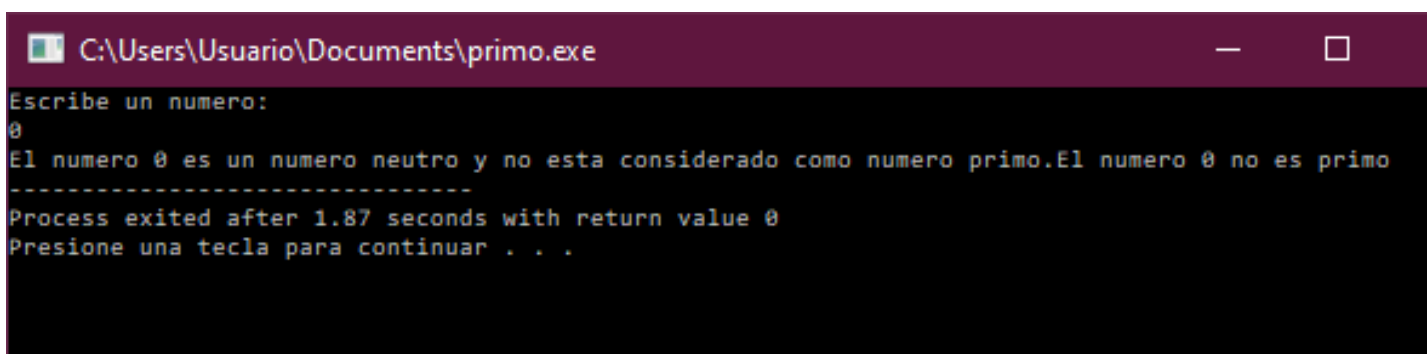
```
[Ruanda37:Documents fp03alu15$ ./main
```

```
Da un numero entero: 7
Da un numero entero: 8
Da un numero entero: 9
Da un numero entero: 10
Da un numero entero: 2
Da un numero entero: 3
Da un numero entero: 4
Da un numero entero: 5
Da un numero entero:
6
Da un numero entero: 7
La suma vale: 61
Y el promedio es 6
```

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

Decidí usar WHILE y DEFINE para este programa:

```
1  #include <stdio.h>
2  #define OSC 2
3  int main()
4  {
5      int numero, divisor;
6      printf ("Escribe un numero:\n");
7      scanf ("%d", &numero);
8      if (numero==0)
9      {
10         printf ("El numero 0 es un numero neutro y no esta considerado como numero primo.");
11     }
12     else
13     {
14         if (numero==1)
15         printf ("El numero 1 no esta considerado como numero primo.");
16     }
17     divisor = numero / OSC;
18     while ((divisor>=1) && (numero%divisor)==0);
19     {
20         divisor = divisor - 1;
21     }
22     if (divisor == 1)
23     {
24         printf ("El numero %d es primo", numero);
25     }
26     else
27     {
28         printf ("El numero %d no es primo", numero);
29     }
30     return 0;
31 }
```



```
C:\Users\Usuario\Documents\primo.exe
Escribe un numero:
0
El numero 0 es un numero neutro y no esta considerado como numero primo.El numero 0 no es primo
-----
Process exited after 1.87 seconds with return value 0
Presione una tecla para continuar . . .
```

```
C:\Users\Usuario\Documents\primo.exe
Escribe un numero:
1
El numero 1 no esta considerado como numero primo.El numero 1 no es primo
-----
Process exited after 1.086 seconds with return value 0
Presione una tecla para continuar . . .
```

```
C:\Users\Usuario\Documents\primo.exe
Escribe un numero:
5
El numero 5 es primo
-----
Process exited after 1.521 seconds with return value 0
Presione una tecla para continuar . . .
```

```
C:\Users\Usuario\Documents\primo.exe
Escribe un numero:
9
El numero 9 no es primo
-----
Process exited after 1.807 seconds with return value 0
Presione una tecla para continuar . . .
```

Conclusión:

Podemos concluir que son muy importantes las estructuras de repetición en programación para poder hacer sucesiones o tablas de multiplicar o una lista y muchas cosas más, por lo que es bueno aprenderlo y practicarlo.