



Estructuras de repetición.

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: Alejandro Esteban Pimentel Alarcon

Asignatura: Fundamentos de Programación

Grupo: 3

Alumna: Aguilar Lara Alexa Patricia

No. de Equipo de cómputo empleado: 26 Nepal

No. de lista o Brigada: 01

No. de cuenta: 316315515

Fecha de entrega: 26/08/2019

CALIFICACIÓN: 7

No se cumplió el objetivo de la práctica de utilizar los tres tipos de ciclo. Además, esta carátula no es correcta, no hay lugar definido para las observaciones, ya te he puesto notas sobre eso anteriormente

Estructuras de repetición

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

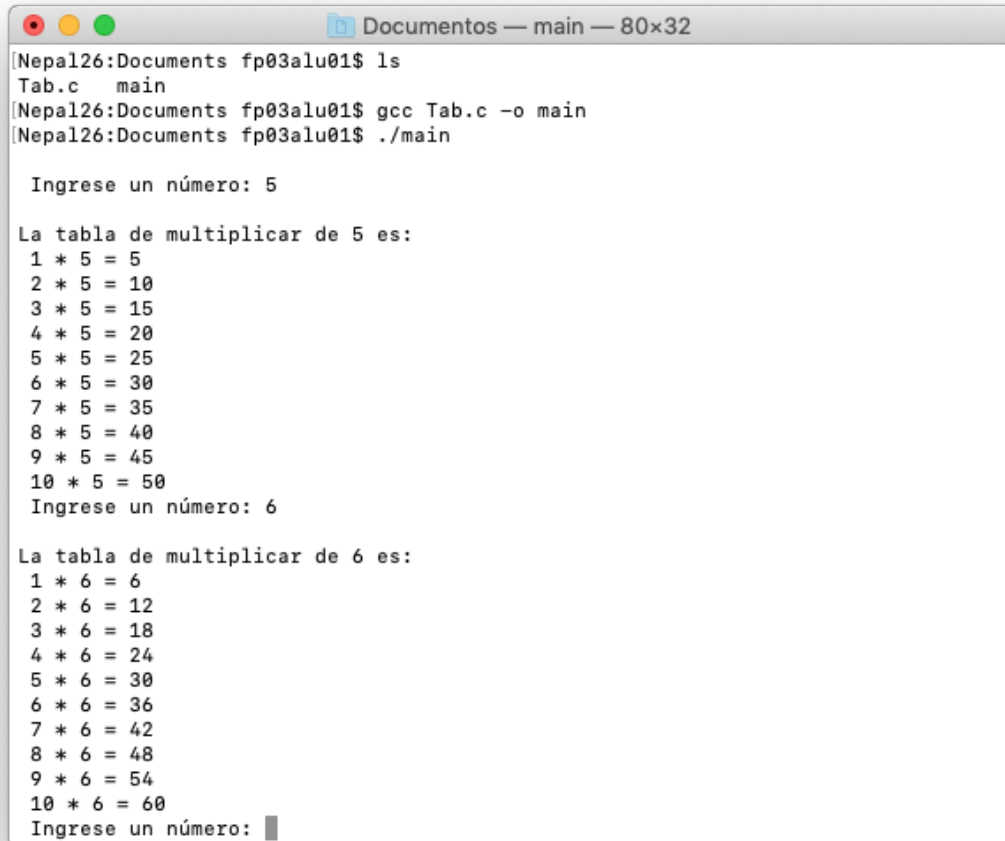
El define es una palabra clave que se utiliza para declarar un nombre especial con un significado. Es muy parecido a una variable, con la diferencia de que no se puede cambiar a lo largo del programa.

#define MAX 5

Actividades

Para cada uno de los siguientes problema, 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).**

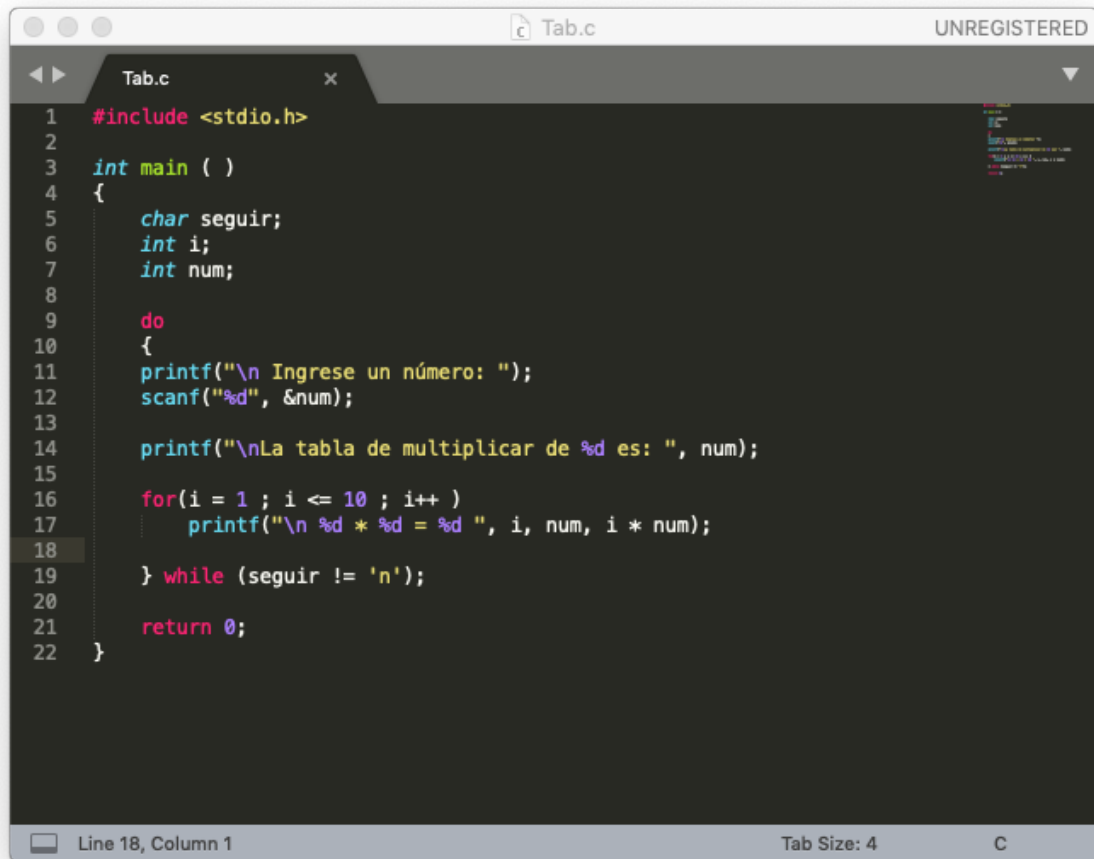


```
Documentos — main — 80x32
[Nepal26:Documents fp03alu01$ ls
Tab.c  main
[Nepal26:Documents fp03alu01$ gcc Tab.c -o main
[Nepal26:Documents fp03alu01$ ./main

Ingrese un número: 5

La tabla de multiplicar de 5 es:
1 * 5 = 5
2 * 5 = 10
3 * 5 = 15
4 * 5 = 20
5 * 5 = 25
6 * 5 = 30
7 * 5 = 35
8 * 5 = 40
9 * 5 = 45
10 * 5 = 50
Ingrese un número: 6

La tabla de multiplicar de 6 es:
1 * 6 = 6
2 * 6 = 12
3 * 6 = 18
4 * 6 = 24
5 * 6 = 30
6 * 6 = 36
7 * 6 = 42
8 * 6 = 48
9 * 6 = 54
10 * 6 = 60
Ingrese un número: █
```



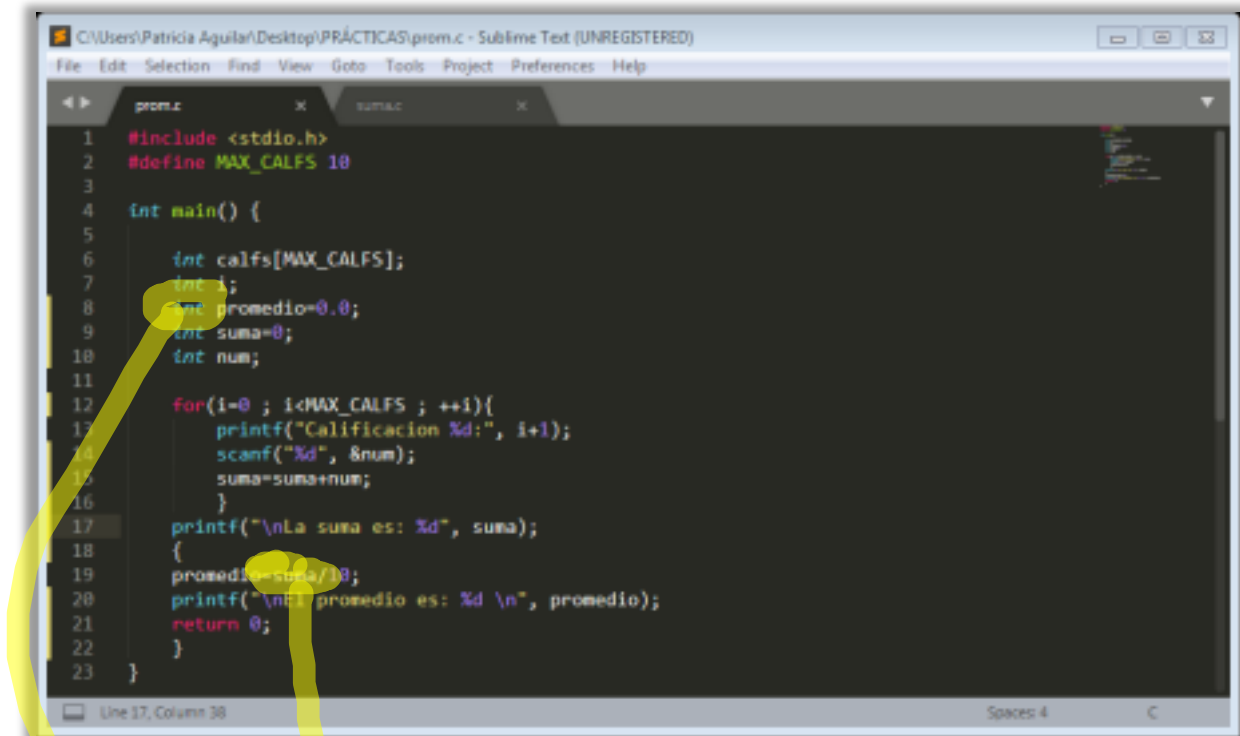
```
1  #include <stdio.h>
2
3  int main ( )
4  {
5      char seguir;
6      int i;
7      int num;
8
9      do
10     {
11         printf("\n Ingrese un número: ");
12         scanf("%d", &num);
13
14         printf("\nLa tabla de multiplicar de %d es: ", num);
15
16         for(i = 1 ; i <= 10 ; i++ )
17             printf("\n %d * %d = %d ", i, num, i * num);
18
19     } while (seguir != 'n');
20
21     return 0;
22 }
```

Line 18, Column 1 Tab Size: 4 C

Para la primera actividad se pidió un programa que mostrara la tabla de multiplicar de un número ingresado hasta el 10, se utilizaron los ciclos do-while y for, para detener la tabla de multiplicar y mostrar el número multiplicado respectivamente.

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

#include <stdio.h>



```
1  #include <stdio.h>
2  #define MAX_CALFS 10
3
4  int main() {
5
6      int calfs[MAX_CALFS];
7      int i;
8      float promedio=0.0;
9      int suma=0;
10     int num;
11
12     for(i=0 ; i<MAX_CALFS ; ++i){
13         printf("Calificacion %d:", i+1);
14         scanf("%d", &num);
15         suma=suma+num;
16     }
17     printf("\nLa suma es: %d", suma);
18     {
19         promedio=suma/10;
20         printf("\nEl promedio es: %d \n", promedio);
21     }
22     return 0;
23 }
```

Para hacer buenas divisiones
necesitas float o double

```

C:\Windows\system32\cmd.exe

C:\Users\Patricia Aguilar\Desktop\PRÁCTICAS>gcc pron.c -o main
C:\Users\Patricia Aguilar\Desktop\PRÁCTICAS>main.exe
Calificacion 1:8
Calificacion 2:8
Calificacion 3:8
Calificacion 4:8
Calificacion 5:8
Calificacion 6:8
Calificacion 7:8
Calificacion 8:8
Calificacion 9:8
Calificacion 10:8

La suma es: 80
El promedio es: 8

C:\Users\Patricia Aguilar\Desktop\PRÁCTICAS>main.exe
Calificacion 1:9
Calificacion 2:8
Calificacion 3:9
Calificacion 4:8
Calificacion 5:9
Calificacion 6:8
Calificacion 7:9
Calificacion 8:8
Calificacion 9:9
Calificacion 10:8

La suma es: 85
El promedio es: 8

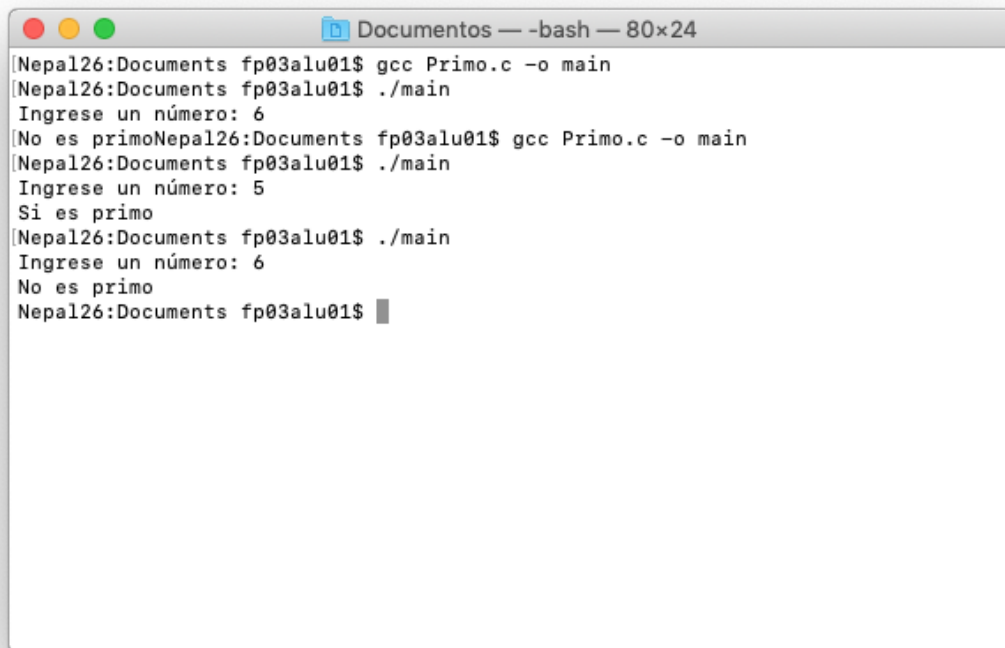
C:\Users\Patricia Aguilar\Desktop\PRÁCTICAS>main.exe
Calificacion 1:2
Calificacion 2:5
Calificacion 3:0
Calificacion 4:8
Calificacion 5:2
Calificacion 6:0
Calificacion 7:10
Calificacion 8:7
Calificacion 9:8
Calificacion 10:10

La suma es: 52
El promedio es: 5

C:\Users\Patricia Aguilar\Desktop\PRÁCTICAS>
```

Para el programa de esta actividad se pedía que se mostrara la suma y el promedio de 10 dígitos a ingresar, se hizo uso de for para definir hasta donde llegaría la suma, con ayuda de define se estableció la máxima cantidad de estos.

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



```
Documentos — -bash — 80x24
[Nepal26:Documents fp03alu01$ gcc Primo.c -o main
[Nepal26:Documents fp03alu01$ ./main
Ingrese un número: 6
No es primo
[Nepal26:Documents fp03alu01$ gcc Primo.c -o main
[Nepal26:Documents fp03alu01$ ./main
Ingrese un número: 5
Si es primo
[Nepal26:Documents fp03alu01$ ./main
Ingrese un número: 6
No es primo
Nepal26:Documents fp03alu01$
```

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

Line 22, Column 2 Tab Size: 4 C

En esta actividad se pedía elaborar un programa en el que se debía ingresar un número y determinar si era primo o no, para su resolución se hizo uso de for para hacer el uso posterior de un if para condición.

En conclusión, elaboramos programas que con ayuda de las estructuras de repetición, nos apoyaron a realizar de manera correcta los programas pedidos, el uso de define para otorgarle un valor dado a una variable en el programa estas estructuras nos dieron como resultado una resolución eficaz de cada actividad.

Aplicando también los conocimientos obtenidos en prácticas pasadas se pudo resolver, corregir y correr cada problema para evaluarlo de manera correcta.