



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: MC. Alejandro Esteban Pimentel Alarcón

Asignatura: Fundamentos de Programación

Grupo: 3

No de Práctica(s): Práctica 9

Integrante(s): Martínez Marcelino Dalila

*No. de Equipo de
cómputo empleado:* No. de cuenta: 313080119

No. de Lista o Brigada:

Semestre: 2020-1

Fecha de entrega: Lunes 14 de octubre de 2019

Observaciones: Muy bien

CALIFICACIÓN: 10

Práctica No. 9

Introducción:

En esta práctica se utilizará procesos de iteraciones para resolver problemas. El estudiante utilizará ciclos como Do-while, While y For para resolver problemas que requieran hacer un proceso de repetición. También el estudiante conocerá y utilizará una constante simbólica llamada *define* al cual asignará un nombre y valor que serán constante.

Objetivo:

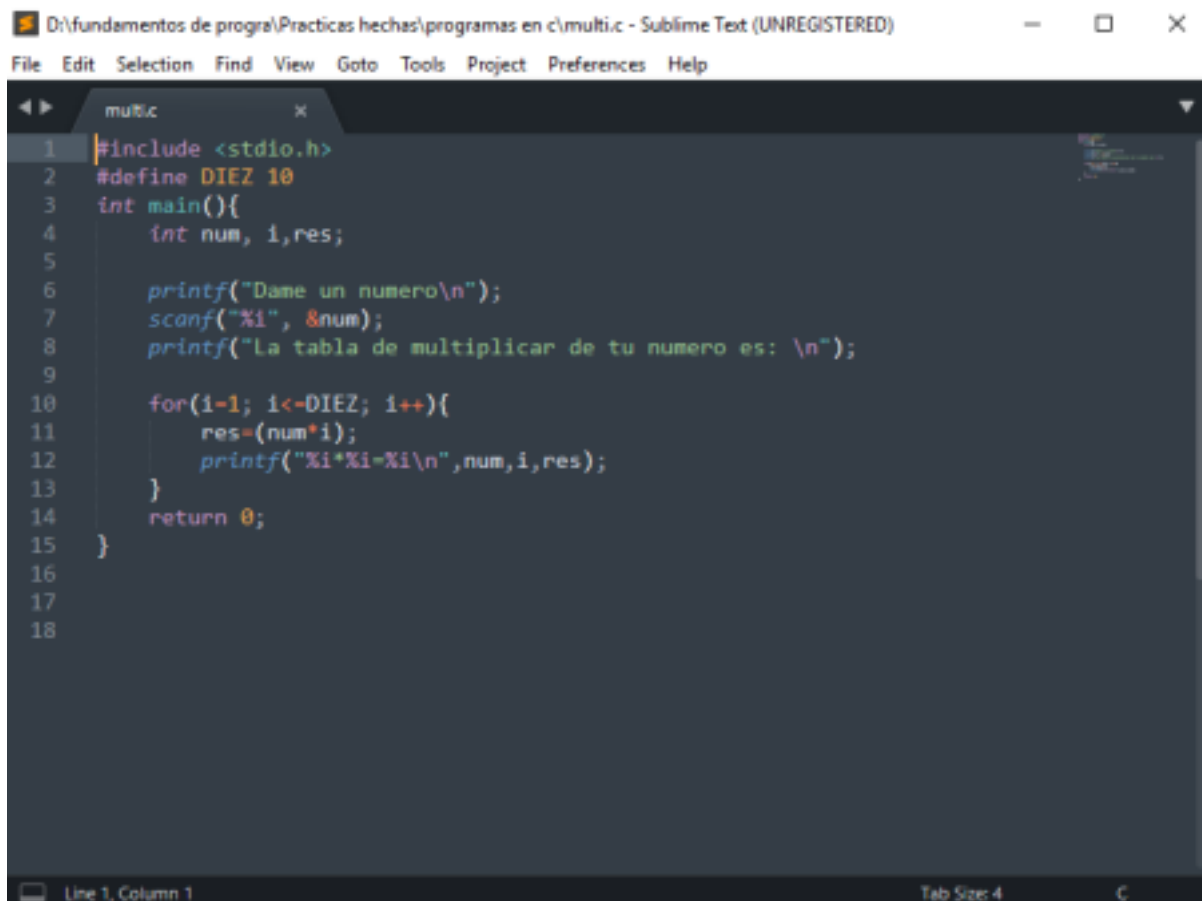
Elaborar programas en C para la resolución de problemas básicos que incluyan las estructuras de repetición y la directiva *define*.

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.

Programa 1.

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

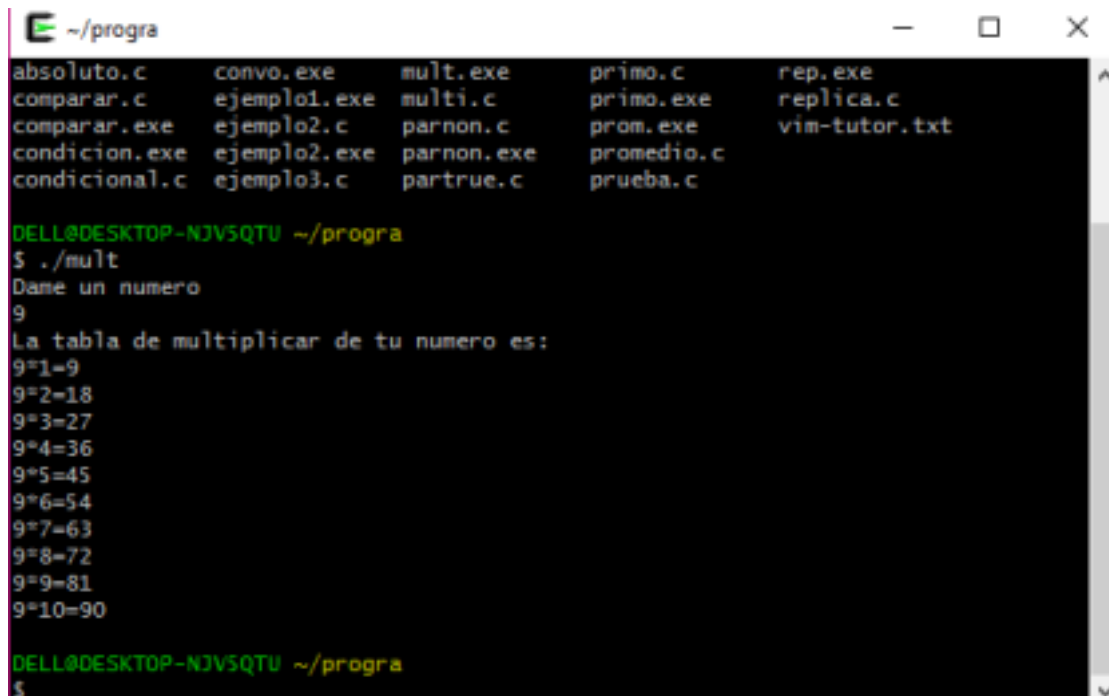


```
D:\fundamentos de progra\Practicas hechas\programas en c\multi.c - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

1 #include <stdio.h>
2 #define DIEZ 10
3 int main(){
4     int num, i,res;
5
6     printf("Dame un numero\n");
7     scanf("%i", &num);
8     printf("La tabla de multiplicar de tu numero es: \n");
9
10    for(i=1; i<=DIEZ; i++){
11        res=(num*i);
12        printf("%i*%i=%i\n",num,i,res);
13    }
14    return 0;
15 }
16
17
18

Line 1, Column 1 Tab Size: 4 C
```

Al hacer la compilación podemos ver que el resultado será la tabla de multiplicar de cualquier número dado. Para este programa se utilizó el ciclo for. También se utilizó la directiva define.



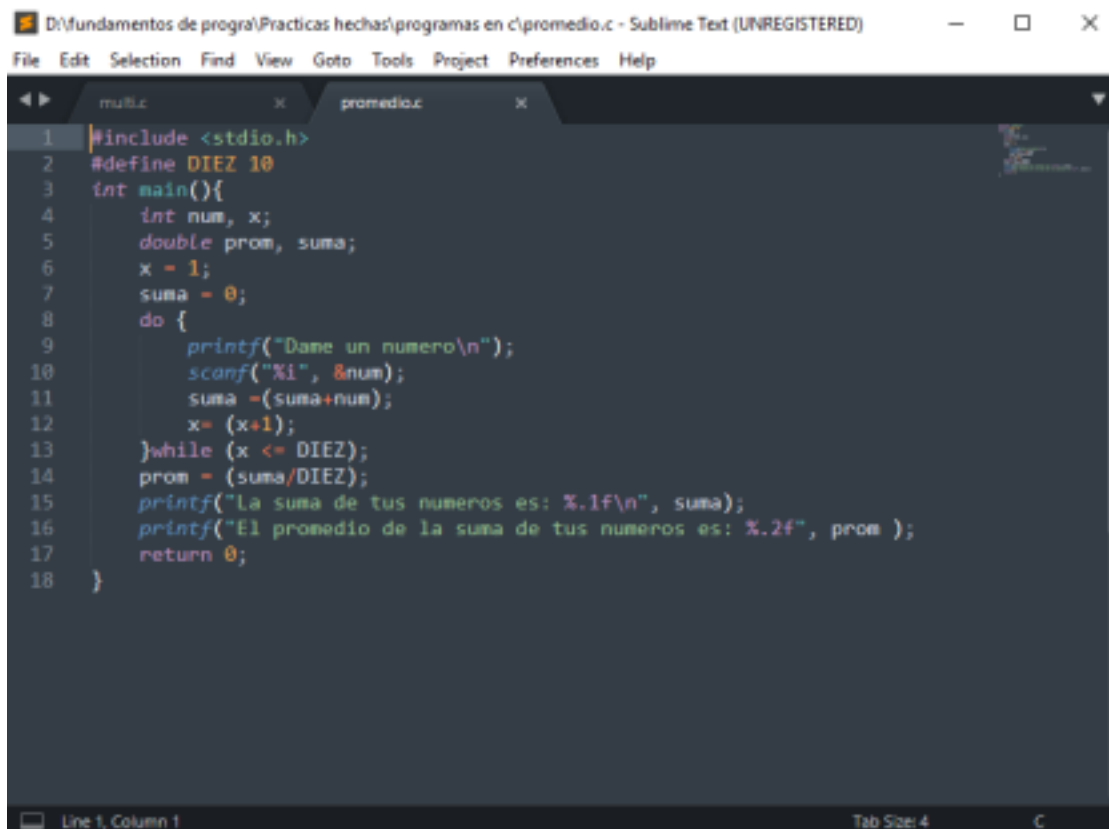
```
~/progra
absoluto.c  convo.exe  mult.exe    primo.c     rep.exe
comparar.c  ejemplo1.exe multi.c     primo.exe   replica.c
comparar.exe ejemplo2.c  parnon.c    prom.exe    vim-tutor.txt
condicion.exe ejemplo2.exe parnon.exe  promedio.c
condicional.c ejemplo3.c  partrue.c   prueba.c

DELL@DESKTOP-NJV5QTU ~/progra
$ ./mult
Dame un numero
9
La tabla de multiplicar de tu numero es:
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

DELL@DESKTOP-NJV5QTU ~/progra
$
```

Programa 2.

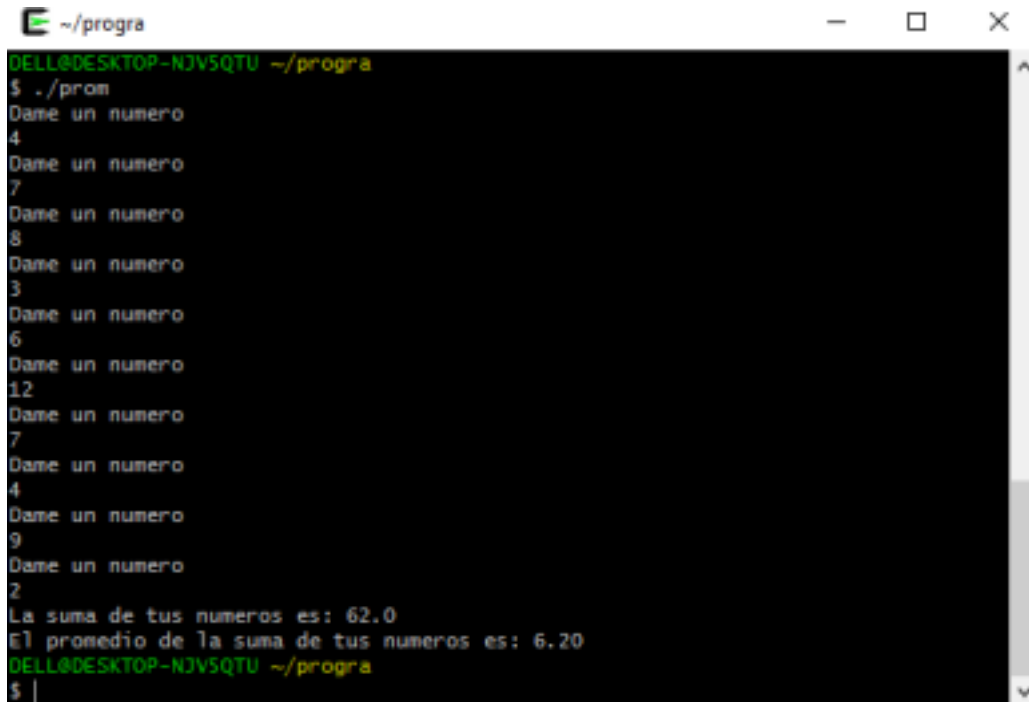
- ✚ Hacer un programa que pida y lea 10 números y muestre su suma y su promedio.



```
D:\Fundamentos de progra\Practicas hechas\programas en c\promedio.c - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

promedio.c
1 #include <stdio.h>
2 #define DIEZ 10
3 int main(){
4     int num, x;
5     double prom, suma;
6     x = 1;
7     suma = 0;
8     do {
9         printf("Dame un numero\n");
10        scanf("%i", &num);
11        suma = (suma+num);
12        x = (x+1);
13    }while (x <= DIEZ);
14    prom = (suma/DIEZ);
15    printf("La suma de tus numeros es: %.1f\n", suma);
16    printf("El promedio de la suma de tus numeros es: %.2f", prom );
17    return 0;
18 }
```

Al hacer la compilación de este programa veremos como resultado la suma de 10 números dados y el promedio de estos. Para este utilice el ciclo do-while. En esta también ocupe la directiva define.



```
DELL@DESKTOP-NJV5QTU ~/progra
$ ./prom
Dame un numero
4
Dame un numero
7
Dame un numero
8
Dame un numero
3
Dame un numero
6
Dame un numero
12
Dame un numero
7
Dame un numero
4
Dame un numero
9
Dame un numero
2
La suma de tus numeros es: 62.0
El promedio de la suma de tus numeros es: 6.20
DELL@DESKTOP-NJV5QTU ~/progra
$
```

Programa 3.

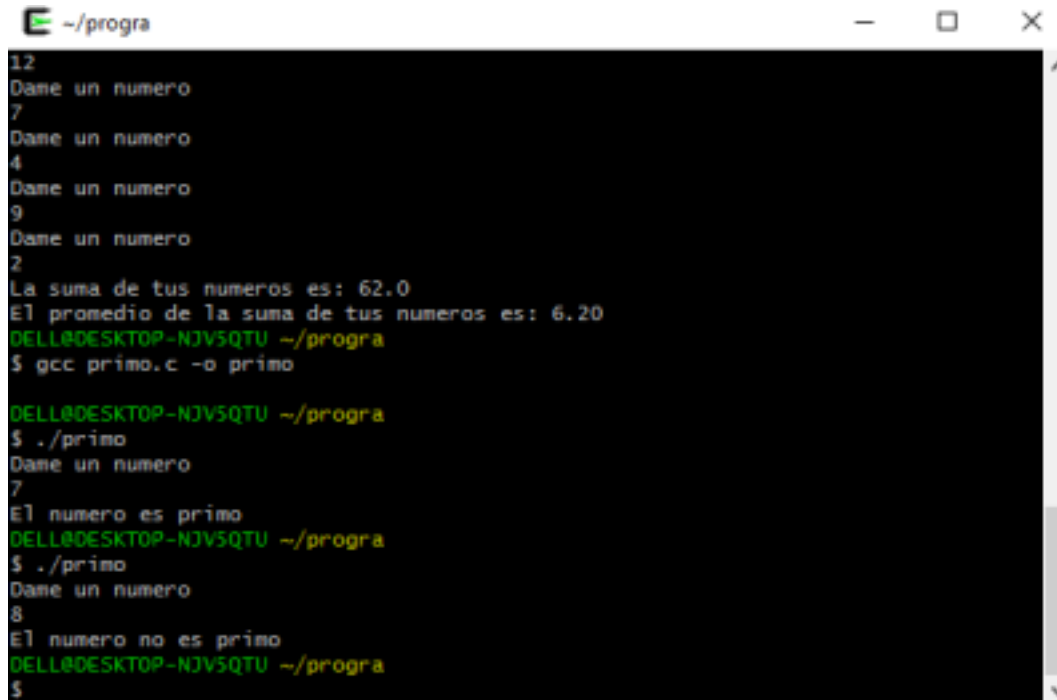
- 🔧 Hacer un programa que pida un número e indique si es primo o no.



```
D:\Fundamentos de progra\Practicas hechas\programas en c\primos.c - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

1 #include <stdio.h>
2 int main(){
3     int num, a, x;
4     a = 1;
5     x = 0;
6     printf("Dame un numero\n");
7     scanf("%i", &num);
8     if(num != 1 || num != 0){
9         while (a<=num && num%a!=0){
10
11             if((num%a) == 0){
12                 x = (x+1);
13             }
14             a = (a+1);
15         }
16         if (x == 2){
17             printf("El numero es primo");
18         }
19         else{
20             printf("El numero no es primo");
21         }
22     }
23     else{
24         printf("El numero no es primo");
25     }
26     return 0;
27 }
```

Al compilar el programa vemos que al darle un número este nos indicara si es primo o no lo es. Para este programa utilice el ciclo de while, y dentro de este while estará un if que pondrá las condiciones necesarias para identificar un número primo.



```
~/progra
12
Dame un numero
7
Dame un numero
4
Dame un numero
9
Dame un numero
2
La suma de tus numeros es: 62.0
El promedio de la suma de tus numeros es: 6.20
DELL@DESKTOP-NJVSQTU ~/progra
$ gcc primo.c -o primo

DELL@DESKTOP-NJVSQTU ~/progra
$ ./primo
Dame un numero
7
El numero es primo
DELL@DESKTOP-NJVSQTU ~/progra
$ ./primo
Dame un numero
8
El numero no es primo
DELL@DESKTOP-NJVSQTU ~/progra
$
```

Conclusión.

Los tres ciclos que se utilizaron en esta práctica, tienen una función similar que es la de repetir, cada uno de ellos es adaptable para cualquier problema que necesite hacer una iteración de un proceso, y el ocupar cada uno de ellos en algún problema en específico dependerá entonces de la opción que elija ocupar la persona en su programa.

La directiva define es de gran ayuda si se desea ocupar un valor constante durante todo el proceso del programa. Me parece que el único inconveniente que tuve en esta práctica fue el poner la condición en el último programa, dado que todos los números son divisibles entre uno y sí mismo, y el encontrar la forma en que el programa me indicara que el número primo solo tuviera esta especificación fue difícil.

Cada uno de los ciclos de iteración son fáciles de usar si se comprende bien el objetivo de lo que queremos que realice nuestro programa. Aunque para mí el ciclo más practico es for dado que es su estructura es corta y el proceso que realiza es igual de eficiente que cualquiera de los otros dos ciclos.