



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):

12

Integrante(s):

Nava Corona Nadia Erandeni

*No. de Equipo de cómputo
empleado:*

No. de Lista o Brigada:

6948

Semestre:

2020-1

Fecha de entrega:

Lunes 04 de noviembre

Observaciones:

Muy bien

CALIFICACIÓN: 10

"Funciones"

Introducción.

Las funciones son usadas por los programadores para hacer sus códigos más cortos, ya que consiste en reducir un gran problema complejo, en pequeños problemitas más sencillos, concentrándose en la solución por separado, de cada uno de ellos.

En C, se conocen como funciones aquellos trozos de códigos utilizados para dividir un programa con el objetivo que, cada bloque realice una tarea determinada.

Podemos declarar estas funciones antes del *main* y cuando se necesiten, simplemente las mandamos a llamar. Algo que demostraremos en esta práctica.

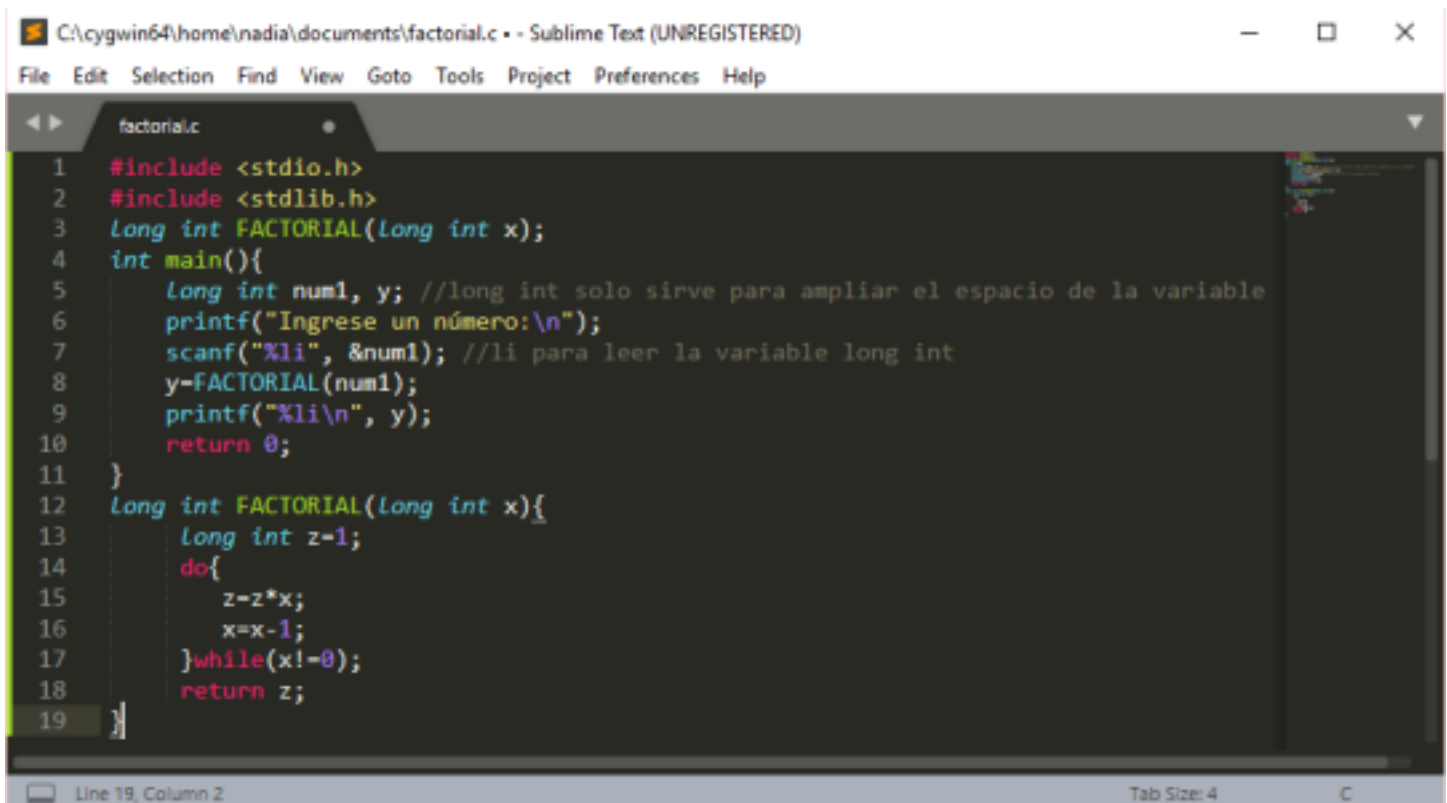
Objetivo.

Elaborar programas en C donde la solución del problema se divida en funciones. Distinguir lo que es el prototipo o firma de una función y la implementación de ella, así como manipular parámetros tanto en la función principal como en otras.

Actividades

Las actividades deben tener los prototipos de sus funciones, y sus funciones implementadas después del main.

- Crear un programa que tenga una función que regrese el factorial de un número de entrada.



```
C:\cygwin64\home\nadia\documents\factorial.c - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

factorial.c
1  #include <stdio.h>
2  #include <stdlib.h>
3  Long int FACTORIAL(Long int x);
4  int main(){
5      Long int num1, y; //long int solo sirve para ampliar el espacio de la variable
6      printf("Ingrese un número:\n");
7      scanf("%li", &num1); //li para leer la variable long int
8      y=FACTORIAL(num1);
9      printf("%li\n", y);
10     return 0;
11 }
12 Long int FACTORIAL(Long int x){
13     Long int z=1;
14     do{
15         z=z*x;
16         x=x-1;
17     }while(x!=0);
18     return z;
19 }
```

Line 19, Column 2 Tab Size: 4 C

```
nadia@LAPTOP-VR22P14N ~/documents
$ gcc factorial.c -o factorial

nadia@LAPTOP-VR22P14N ~/documents
$ ./factorial 5
Ingrese un número:
5
120

nadia@LAPTOP-VR22P14N ~/documents
$ ./factorial
Ingrese un número:
7
5040

nadia@LAPTOP-VR22P14N ~/documents
$ ./factorial
Ingrese un número:
3
6

nadia@LAPTOP-VR22P14N ~/documents
$ |
```

- Crear un programa que tenga una función que regrese el resultado de la serie:

$$\sum_{x=1}^n \frac{x!}{x}$$

Para un número n de entrada. Utilizar la función de factorial de la primera actividad.

```
factorial.c  funcion.c
1  #include <stdlib.h>
2  #include <stdio.h>
3  Long int FACTORIAL(Long int x);
4  Long int SERIE(Long int x);
5  int main(){
6      Long int num1, res;
7      printf("Ingrese un número\n");
8      scanf("%li", &num1);
9      res=SERIE(num1);
10     printf("%li", res);
11     return 0;
12 }
13 Long int FACTORIAL(Long int x){
14     Long int z=1;
15     do{
16         z=z*x;
17         x=x-1;
18     }while(x!=0);
19     return z;
20 }
21 Long int SERIE(Long int x){
22     Long int y;
23     Long int s=1;
24     Long int w=0;
25     do{
26         y=FACTORIAL(s)/s;
27         w=w+y;
28         s++;
29     }while(s<=x);
30     return w;
31 }
```

~/documents

\$ gcc funcion.c -o funcion

nadia@LAPTOP-VR22P14N ~/documents

\$./funcion

Ingrese un número

5

34

nadia@LAPTOP-VR22P14N ~/documents

\$./funcion

Ingrese un número

4

10

nadia@LAPTOP-VR22P14N ~/documents

\$./funcion

Ingrese un número

9

46234

nadia@LAPTOP-VR22P14N ~/documents

\$./funcion

Ingrese un número

7

874

nadia@LAPTOP-VR22P14N ~/documents

\$

Conclusiones.

Las funciones ayudan a tener un programa más corto y organizado, pues en vez de escribir el mismo bloque de código a ejecutar a lo largo de tu programa, con las funciones no debemos escribirlas nuevamente, sino solamente mandarlas a llamar. Además de que nos ayudan a reducir el tiempo de codificación, y disminuir los errores en nuestro programa.