



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: M.C. ALEJANDRO ESTEBAN PIMENTEL ALARCON

Asignatura: FUNDAMENTOS DE PROGRAMACION

Grupo: 3

No de Práctica(s): PRACTICA 12

Intearante(s): 1

*No. de Equipo de cómputo
empleado:*

Equipo 3

No. de Lista o Brigada: 420054913

Semestre: 2020-1

Fecha de entrega: 4 DE NOVIEMBRE 2019

Observaciones:

CALIFICACIÓN: _____

EN ESTA PRÁCTICA SE APRENDERÁ A USAR FUNCIONES, SI SE NECESITAN, PERO DECLARÁNDOLAS DESDE EL PRINCIPIO; EL CÓMO ESTAS SE USAN Y SE MANDAN A LLAMAR PARA SU USO POSTERIOR EN EL PROGRAMA.

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

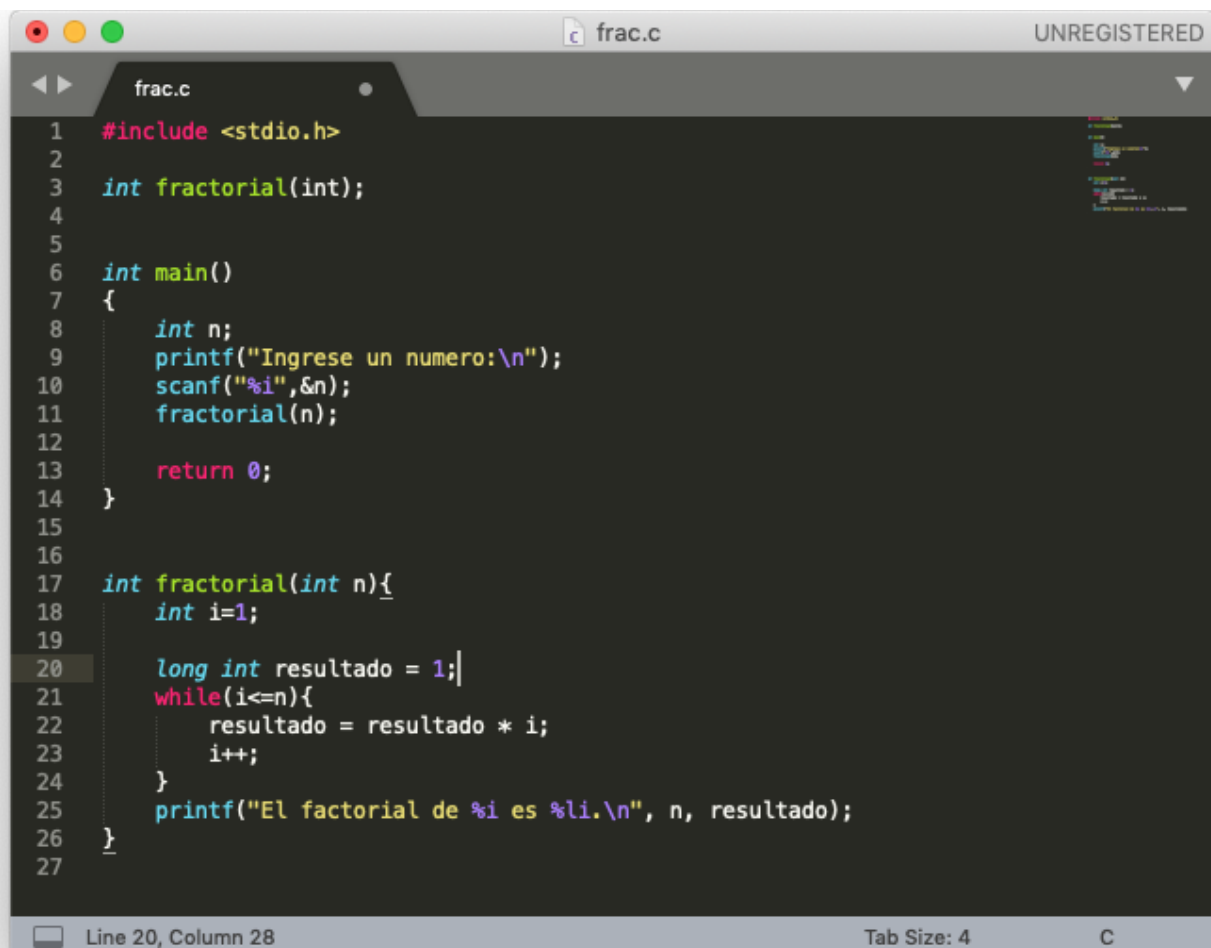
Las actividades deben tener los prototipos de sus funciones; y sus funciones implementadas después del main.

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

Primero se creará la función sin desarrollar, para que de esta manera se pueda desarrollar posteriormente el interior de esta.

En el main solo se necesita que se pida el valor a utilizar, y ahí se manda a llamar la función a usar, y el valor que usara dentro de ella.

Después ahora si se desarrolla la función que usaremos, en este caso la función realiza la operación para que regrese el factorial del número que pedimos.



```
1  #include <stdio.h>
2
3  int factorial(int);
4
5
6  int main()
7  {
8      int n;
9      printf("Ingrese un numero:\n");
10     scanf("%i",&n);
11     factorial(n);
12
13     return 0;
14 }
15
16
17 int factorial(int n){
18     int i=1;
19
20     long int resultado = 1;
21     while(i<=n){
22         resultado = resultado * i;
23         i++;
24     }
25     printf("El factorial de %i es %li.\n", n, resultado);
26 }
27
```

Line 20, Column 28 Tab Size: 4 C

```
fp03alu23 — -bash — 80x38
^
1 warning generated.
[Kenia01:~ fp03alu23$ gcc frac.c -o fra
frac.c:15:1: warning: control reaches end of non-void function [-Wreturn-type]
}
^
1 warning generated.
[Kenia01:~ fp03alu23$ gcc frac.c -o fra
frac.c:15:1: warning: control reaches end of non-void function [-Wreturn-type]
}
^
1 warning generated.
[Kenia01:~ fp03alu23$ ./fra
Ingrese un numero:
5
El factorial de 5 es 120.
[Kenia01:~ fp03alu23$ ./fra
Ingrese un numero:
10
El factorial de 10 es 3628800.
[Kenia01:~ fp03alu23$ ./fra
Ingrese un numero:
3
El factorial de 3 es 6.
[Kenia01:~ fp03alu23$ ./fra
Ingrese un numero:
4
El factorial de 4 es 24.
[Kenia01:~ fp03alu23$ ./fra
Ingrese un numero:
1
El factorial de 1 es 1.
[Kenia01:~ fp03alu23$ ./fra
Ingrese un numero:
2
El factorial de 2 es 2.
[Kenia01:~ fp03alu23$ ]
```

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

Como en la anterior actividad para su resolución se usarán funciones y de esta manera se resolverá.

Teniendo ya la función que realiza sacar el número factorial, se usara directamente en la función recorrer, esta lo que hace es realizar la operación pedida anteriormente, de numero en número, ¡primero con $1! / 1 + 2! / 2 + 3! / 3 + \dots + n! / n$, aquí lo que hará detener el programa es cuando el contador llegue al valor de n .

valoressuma.c

UNREGISTERED

valoressuma.c

```
1  #include <stdio.h>
2
3  int factorial(int);
4  long int recorrer(int);
5
6
7  int main(){
8      int n;
9      printf("valor x:");
10     scanf("%i",&n);
11     recorrer(n);
12
13 }
14
15
16 int factorial(int n){
17     int i=1;
18
19     long int resultado = 1;
20     while(i<=n){
21
22         resultado = resultado * i;
23         i++;
24     }
25     return resultado;
26 }
27
28 long int recorrer(int n){
29     long int x =0;
30     long int m;
31     for(int i=1; i<=n;i++){
32         m= (factorial(i)/i);
33         x = x + m;}
34
35     printf ("valor de la suma. de esto\n %li \n",x);
36
37
38 }
39
```

Line 12, Column 1

Tab Size: 4

C

```
fp03alu23 — -bash — 80x38
valor de la suma. de esto
[ 34Kenia01:~ fp03alu23$ ./ss
valor x:4
valor de la suma. de esto
[ 10Kenia01:~ fp03alu23$ ./ss
valor x:5
valor de la suma. de esto
[ 34Kenia01:~ fp03alu23$ gcc valoressuma.c -o ss
valoressuma.c:38:2: warning: control reaches end of non-void function
      [-Wreturn-type]
      }
      ^
1 warning generated.
[Kenia01:~ fp03alu23$ ./ss
valor x:7
valor de la suma. de esto
874
[Kenia01:~ fp03alu23$ ./ss
valor x:2
valor de la suma. de esto
2
[Kenia01:~ fp03alu23$ ./ss
valor x:3
valor de la suma. de esto
4
[Kenia01:~ fp03alu23$ ./ss
valor x:4
valor de la suma. de esto
10
[Kenia01:~ fp03alu23$ ./ss
valor x:5
valor de la suma. de esto
34
[Kenia01:~ fp03alu23$ ./ss
valor x:6
valor de la suma. de esto
154
Kenia01:~ fp03alu23$
```

Con esta práctica nos pudimos dar una idea general del cómo funcionan en el código el manejo de las funciones y su el reciclare en estas, para el uso en otros programas. De esta manera se sabe cómo se hace un código cualquiera, pero dividiendo sus acciones entre funciones y que este aun así muestra los resultados que se quieren.