	Carátula para entrega de prácticas	
Facultad de Ingeniería		Laboratorio de docencia

Laboratorios de computación

Salas A y B

<i>Profesor:</i>	Alejandro Esteban Pimentel Alarcon
<i>Asignatura:</i>	Fundamentos de Programación
<i>Grupo:</i>	135
<i>No de Práctica(s):</i>	12
<i>Integrante(s):</i>	Godínez Juárez Alondra Itati
<i>No. de Equipo de cómputo empleado:</i>	
<i>No. de Lista o Brigada:</i>	316146153
<i>Semestre:</i>	2020-1
<i>Fecha de entrega:</i>	Lunes 4 de noviembre del 2019
<i>Observaciones:</i>	No se cumple el objetivo de utilizar los prototipos de las funciones. Además, la segunda actividad no cumple con lo establecido, el resultado de la sumatoria es incorrecto, solo se obtiene el último término de la suma.

CALIFICACIÓN: 7

Funciones

Introducción. En la presente práctica veremos la importancia y funcionalidad de usar funciones en c y cómo nos puede salvar de escribir largas líneas de código de una manera sencilla.

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.

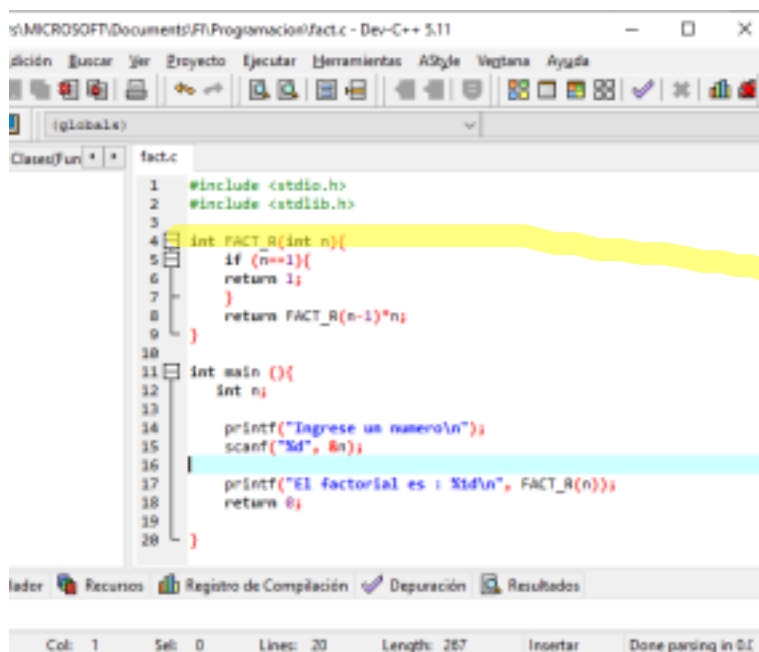
Las funciones permiten a un programador modularizar un programa. Todas las variables declaradas en las definiciones de función son variables locales (son conocidas solo en la función en la cual están definidas)

La mayor parte de las funciones tienen una lista de parámetros. Los parámetros proporcionan la forma de comunicar información entre funciones, siendo también variables locales. Cada función deberá limitarse a ejecutar una tarea sencilla y bien definida. El nombre deberá expresar claramente dicha tarea.

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

Actividad

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



```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int FACT_R(int n){
5     if (n==1){
6         return 1;
7     }
8     return FACT_R(n-1)*n;
9 }
10
11 int main (){
12     int n;
13
14     printf("Ingrese un numero\n");
15     scanf("%d", &n);
16
17     printf("El factorial es : %d\n", FACT_R(n));
18     return 0;
19 }
20
```

Falta el
prototipo

```
C:\Users\MICROSOFT\Documents\FI\Programacion\fact.exe
Ingrese un numero
4
El factorial es : 24
-----
Process exited after 27.55 seconds with return value 0
Press any key to continue . . .
```

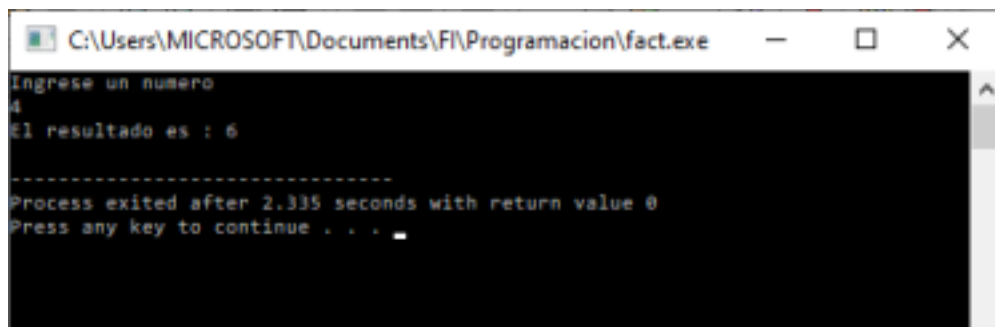
- Crear un programa que tenga una función que regrese el resultado de la serie:

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

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

Esto no cumple con la sumatoria solo obtiene el último término

```
fact.c
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int FACT_R(int n){
5      if (n==1){
6          return 1;
7      }
8      return FACT_R(n-1)*n;
9  }
10 int y(int n){
11     if (n==1){
12         return 1;
13     }
14     return FACT_R(n)/n;
15 }
16
17 int main (){
18     int n;
19
20     printf("Ingrese un numero\n");
21     scanf("%d", &n);
22
23     printf("El resultado es : %d\n", y(n));
24     return 0;
25 }
26
```



```
C:\Users\MICROSOFT\Documents\FI\Programacion\fact.exe
Ingrese un numero
4
El resultado es : 6

-----
Process exited after 2.335 seconds with return value 0
Press any key to continue . . .
```

Conclusión. Vimos y probamos a través de encontrar el factorial de un número y el resultado de la serie entre el factorial de un número y el mismo. Como si hubiéramos utilizado otro método nos hubiéramos que tenido que mezclar diferentes herramientas como estructuras de repetición, con largas líneas de código, pero con funciones todo se reduce y optimiza.