
 GOBIERNO DEL ESTADO DE MÉXICO	MANUAL DE PRÁCTICAS FO-TESJI-11100-12			 TECNOLÓGICO DE ESTUDIOS SUPERIORES JILOTEPEC	
Nombre de la práctica	CADENAS Y FUNCIONES			No.	13
Asignatura:	METODOS NUMERICOS	Carrera:	ING. SISTEMAS COMPUTACIONALES	Duración de la práctica (Hrs)	10

Nombre del alumno: Lissette Garcia Nolasco

I. Competencia(s) específica(s):

II. Lugar de realización de la práctica (laboratorio, taller, aula u otro):

Aula

III. Material empleado:

Dev C++

Computadora

IV. Desarrollo de la práctica:

FUNCIONES

Definición de función matemática

Matemáticamente una función es una operación que toma uno o más valores llamados argumentos y produce un valor llamado resultado.

Definición de función

- Una función es un bloque de código reconocido por un identificador que realiza un trabajo específico.
- Su propósito es dividir los programas en módulos manejables separados (divide y vencerás).

Ventajas

1. Facilita el diseño descendente.
2. Los procedimientos dentro de ellas se pueden ejecutar varias veces.
3. Facilita la división de tareas.
4. Se pueden probar individualmente

5. Con funciones apropiadamente diseñadas, es posible ignorar como se realiza una tarea, sabiendo qué es lo que hacen.

Modo de uso

1. Funciones diseñadas para realizar operaciones a partir de sus argumentos y devolver un valor basado en sus cálculos.
2. Funciones que no reciben argumentos, realizan un proceso y devuelven un valor .
3. Funciones que no tienen argumentos ni valor de retorno explícito, operan sobre el entorno de variables globales o atributos del sistema operativo.

Llamadas a funciones

- Para llamar a una función se especifica su nombre y la lista de argumentos sin poner el tipo de dato.
- En una llamada habrá un argumento real por cada argumento formal, respetando el orden de declaración.
- argumento formal: Los que aparecen en la definición de la función.
- argumento real: Los que se pasan en la llamada a la función.

Paso de parámetros por valor

- Se hace una copia del valor del argumento en el parámetro formal.
- La función opera internamente con estos últimos.
- Los parámetros formales se crean al entrar a la función y se destruyen al salir de ella, cualquier cambio realizado por la función en los parámetros formales no tienen ningún efecto sobre los argumentos.

EJEMPLO 1

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  float promedio (float a, float b);
4  int main(){
5      float a=5, b=10, prom;
6      prom=promedio (a, b);
7      printf ("El promedio es:%2.1f\n"
8      system ("Pause" );
9      return 0;
10 }
11 float promedio (float a, float b){
12     float prom;
13     a =a+3;
14     b =b+3;
15     prom =( a+b) / 2;
16
17     return prom;
18 }
```

```
El promedio es:10.5
Presione una tecla para continuar . . .
```

Variables locales y globales

☐ Variables Locales:

☐ Se declaran dentro de la función y sólo están disponibles durante su ejecución.

☐ Se crean cuando se entra en ejecución una función y se destruyen cuando se termina.

☐ Variables globales:

☐ Se declaran fuera de las funciones. Pueden ser utilizadas por todas las funciones.

☐ Existen durante toda la vida del programa.

Ejercicio

- Escribir una función que se llame maximo que reciba dos número por parámetros y que regrese el mayor de ellos.
- Escribir una función que reciba caracteres del teclado hasta recibir un espacio o un salto de línea (enter) y a continuación mostrar todos los caracteres en orden inverso.

Ejemplo:

- ☐ Entrada: Hola
- ☐ Salida: aloH

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  int main(){
5      char a[200];
6      int len,i=0,j;
7      puts("INGRESE UNA PALABRA:\n");
8      scanf("%s",a);
9      printf("LA PALABRA INTRODUCIDA ES: %s\n",a);
10     len=strlen(a);
11     while(a[i]!='\0'){
12         i++;
13     }
14     printf("LA PALABRA A LA REVES ES: ");
15     for(j=i-1;j>=0;j--){
16         printf("%c",a[j]);
17     }
18     puts("");
19     return 0;
20 }
```

INGRESE UNA PALABRA:

Hola

LA PALABRA INTRODUCIDA ES: Hola

LA PALABRA A LA REVES ES: aloH

Process exited after 4.021 seconds with return value 0
Presione una tecla para continuar . . .

Ejercicio 3:

Escribir una función que tome como parámetros las longitudes de los tres lados de un triángulo (a, b, c) y devuelva el área del triángulo.

$$A = \sqrt{p(p-a)(p-b)(p-c)} \text{ Donde } p = \frac{a+b+c}{2}$$

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <math.h>
4
5 float calcular(float a, float b, float c){
6     if(a!=0&b!=0&c!=0){
7         float s=(float)(a+b+c)/2;
8         return s;
9     }
10 }
11 float area(float a, float b, float c){
12     if(a!=0&b!=0&c!=0){
13         float s=calcular(a,b,c);
14         float area=0;
15         if(s>a&s>b&s>c){
16             area=sqrt(s*(s-a)*(s-b)*(s-c));
17         }
18         return area;
19     }
20 }
21 int main(){
22     printf("AREA DEL TRIANGULO\n");
23     int i=0;
24     float lados[3];
25     for(i=0;i<3;i++){
26         printf("INGRESA LA LONGIUD DEL LADO %i:\n",i+1);
27         scanf("%f",&lados[i]);
28     }
29     float area=calcular(lados[0],lados[1],lados[2]);
30     if(area==0){
31         printf("LOS DATOS INGRESADOS NO CORRESPONDEN");
32     }else{
33         printf("EL AREA DEL TRIANGULO ES:%.3f",area);
34     }
35     getch();
36     return 0;
37 }
```

```
AREA DEL TRIANGULO
INGRESA LA LONGIUD DEL LADO 1:
5
INGRESA LA LONGIUD DEL LADO 2:
4
INGRESA LA LONGIUD DEL LADO 3:
3
EL AREA DEL TRIANGULO ES:6.000
```

Funciones recursivas

- ° Se llaman funciones recursivas a aquellas que se llaman a sus mismas de forma repetida hasta que se cumpla alguna condición.
- ° Cada llamada implica el almacenamiento de variables de estado y otros parámetros.

Ejemplo 3:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 int main(){
5     int x, y, nax;
6     x=2;
7     y=3;
8     nax=potencia(x,y);
9     printf("LA POTENCIA ES: %d\n", nax);
10    system("Pause");
11    return 0;
12 }
13 int potencia (a,b){
14     if(b<1)
15         return 1;
16     return a*potencia(a,b-1);
17 }
```

```
LA POTENCIA ES: 8
Presione una tecla para continuar . . .
```

Ejercicio 4:

Haz un programa con funciones recursivas que calcule el factorial de un número n ingresado desde teclado.

Ej. $N = 5$

$5! = 4! * 5$

$4! = 3! * 4$

$3! = 2! * 3$

$2! = 1! * 2$

$1! = 0! * 1$

$0! = 1$



GOBIERNO DEL
ESTADO DE MÉXICO

MANUAL DE PRÁCTICAS

FO-TESJI-11100-12



TECNOLÓGICO DE ESTUDIOS SUPERIORES
JILOTEPEC

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 int factorial(int numero);
5 int main(){
6     int numero, resul;
7     puts("INTRODUCE UN NUMERO");
8     scanf("%d", &numero);
9     resul=factorial(numero);
10    printf("EL FATORIAL DE %d ES: %d\n", numero, resul);
11    return 0;
12 }
13 int factorial(int numero){
14     int resul;
15     resul=1;
16     while(numero>1){
17         resul*=numero;
18         numero--;
19     }
20     return resul;
21 }
```

INTRODUCE UN NUMERO

8

EL FATORIAL DE 8 ES: 40320

Process exited after 3.126 seconds with return value 0

Presione una tecla para continuar . . .