	<b>Carátula para entrega de prácticas</b>	
Facultad de Ingeniería	Laboratorio de docencia	

# Laboratorios de computación salas A y B

*Profesor:* Karina García Morales

*Asignatura:* Fundamentos de programación

*Grupo:* 22

*No. de práctica(s):* Práctica No.11

*Integrante(s):* Freddy Beckham Cedillo Arias

*No. de lista o brigada:* No.11

*Semestre:* 1er Semestre

*Fecha de entrega:* 1 de noviembre del 2024

*Observaciones:*

**CALIFICACIÓN:** \_\_\_\_\_

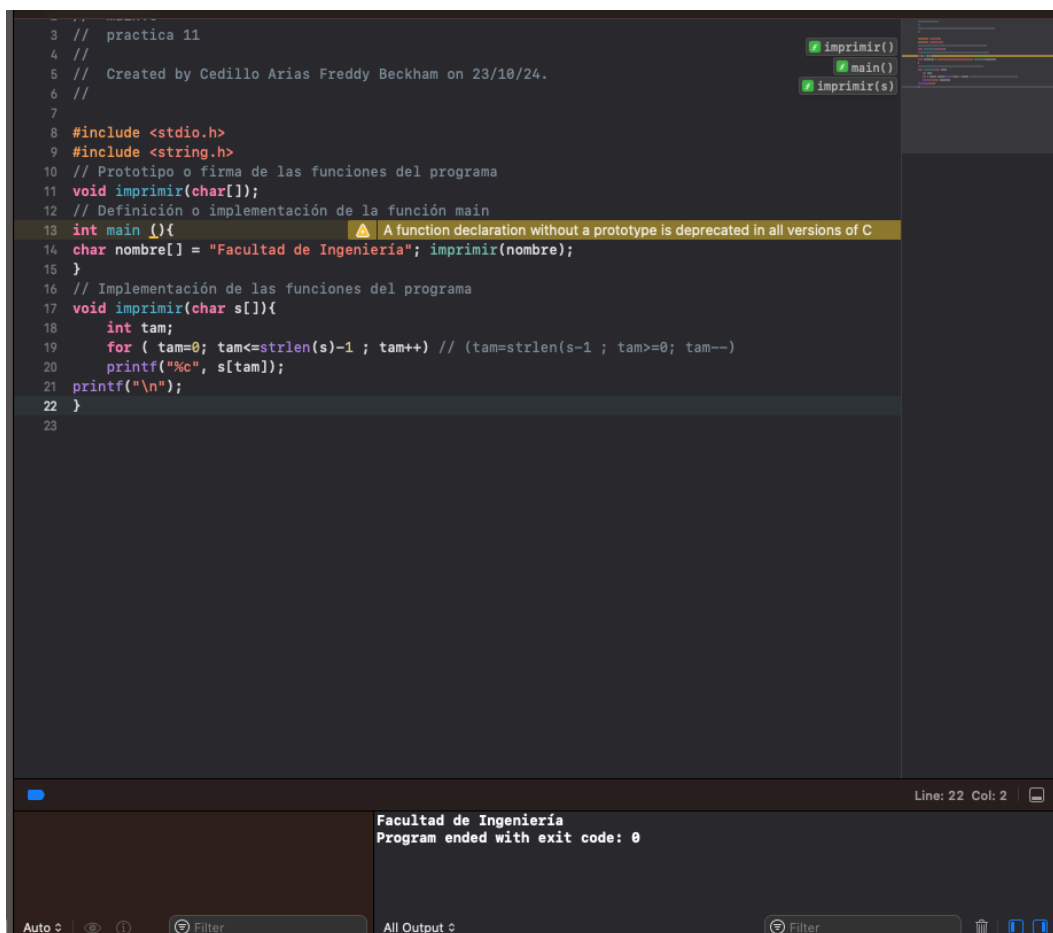
# Practica 11

## Funciones

### Objetivo:

El alumno elaborará 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.

### Desarrollo:



```
1 // practica 11
2 //
3 // Created by Cedillo Arias Freddy Beckham on 23/10/24.
4 //
5
6 #include <stdio.h>
7 #include <string.h>
8 // Prototipo o firma de las funciones del programa
9 void imprimir(char[]);
10 // Definición o implementación de la función main
11 int main () {
12     char nombre[] = "Facultad de Ingeniería"; imprimir(nombre);
13 }
14 // Implementación de las funciones del programa
15 void imprimir(char s[]){
16     int tam;
17     for ( tam=0; tam<=strlen(s)-1 ; tam++) // (tam=strlen(s)-1 ; tam>=0; tam--)
18         printf("%c", s[tam]);
19     printf("\n");
20 }
21
22
23
```

Facultad de Ingeniería  
Program ended with exit code: 0

Práctica 1:

```
1 //
2 // main.c
3 // practica 11
4 //
5 // Created by Cedillo Arias Freddy Beckham on 23/10/24.
6 //
7
8 #include <stdio.h>
9 void sumar(){
10     int x=5, y=10; // variables locales
11     int z;
12     z=x+y;
13     printf("%d \n", z);
14 }
15 int main()
16 {
17     sumar(); //llamado de la funcion
18 }
19
```

Facultad de Ingeniería  
Program ended with exit code: 0

Practica2:

```
1 //
2 // main.c
3 // practica 11
4 //
5 // Created by Cedillo Arias Freddy Beckham on 23/10/24.
6 //
7
8 #include <stdio.h>
9 int resultado; //variable global
10 void multiplicar() //función multiplicar ⚠ A function declaration without a prototype is deprecated in...
11 {
12     resultado = 5 * 4;
13     printf("%d",resultado);
14 }
15 int main() ⚠ A function declaration without a prototype is deprecated in all versions of C
16 {
17     multiplicar(); //llamado de la función multiplicar
18     return 0;
19 }
20
```

Facultad de Ingeniería  
Program ended with exit code: 0

Programa 3

```
1 //
2 // main.c
3 // practica 11
4 //
5 // Created by Cedillo Arias Freddy Beckham on 23/10/24.
6 //
7
8 #include <stdio.h>
9 #include <stdio.h>
10 void incremento();
11 /* La variable enteraGlobal es vista por todas
12 las funciones (main e incremento) */
13 int enteraGlobal;
14 void incremento()
15 {
16     // La variable enteraLocal es local a la función incremento
17     int enteraLocal = 5;
18     enteraGlobal += 2;
19     printf("global(%i) + local(%i) = %d\n", enteraGlobal, enteraLocal, enteraGlobal+enteraLocal);
20 }
21 int main()
22 {
23     int enteraGlobal;
24     // La variable cont es local a la función main
25     int cont;
26     enteraGlobal = 0; // La función main accede a la variable global
27     for(cont=0; cont<5; cont++)
28     {
29         incremento();
30     }
31     return 999;
32 }
33 |
```

Program ended with exit code: 231

Auto Filter All Output Filter

##### calculadora.c #####

```
#include <stdio.h>
int suma(int,int);
int resta(int,int);
int producto(int,int);
static int cociente (int,int);
int main() {
printf("5 + 7 = %i\n",suma(5,7));
printf("9 - 77 = %d\n",resta(9,77));
printf("6 * 8 = %i\n",producto(6,8));
printf("7 / 2 = %d\n",cociente(7,2));
}
static int cociente (int a, int b)
{
return (int)(a/b);
}
```

~  
~  
~  
~  
~  
~

"calculadora.c" 16L, 385B

Programa 4

```

//##### funcEstatica.c #####
#include <stdio.h>
int suma(int,int);
int resta(int,int);
int producto(int,int);
static int cociente (int,int);
int suma (int a, int b) {
return a + b; }
int resta (int a, int b) {
return a - b; }
int producto (int a, int b) {
return (int)(a*b); }
//static int cociente (int a, int b) {
//return (int)(a/b);
// }
~
~
~
~
~
~
~

```

(Lamentablemente no corrió el código , se hizo una inspección y comparación con otros códigos y no corría correctamente )

Programas iniciales en página de laboratorio práctica 11

```

1 ##### calculadora.c #####
2 #include <stdio.h>
3 int suma (int,int);
4 //static int resta(int,int),
5 int producto (int,int)
6 //static int cociente (int,int)
7 int main ()
8 {
9 printf("5 + 7= %i\n", suma (5,7));
10 //printf("9 - 77 = %d\n",resta(9,77));
11 printf("6 * 8 = %i\n",producto (6,8))
12 //printf("7 / 2 = %d\n",cociente(7,2));
13 }

```

```

1 ##### funcEstatica.c #####
2 #include <stdio.h>
3 int suma (int,int)
4 static int resta (int,int);
5 int producto (int,int);
6 static int cociente (int,int)
7 int suma (int a, int b)
8 {
9 return a b;
10 }
11 static int resta (int a, int b)
12 {
13 return a - b;
14 }
15 int producto (int a, int h)
16 {
17 return (int) (a*b);
18 }
19 static int cociente (int a, int b)
20 {
21 return (int) (a/b);
22 }

```

Las funciones resta y cociente se declaran como static, lo que significa que solo serán accesibles dentro del archivo en el que están definidas (en este caso, calculadora.c). Esto impide que cualquier otra parte del programa o módulo (en otros archivos) las llame.

```
##### funcEstatica.c #####
#include <stdio.h>
int suma(int,int);
static int resta(int,int)
int producto (int, int)
static int cociente (int,int)
int suma (int a, int h)
{
return a + b;
}
static int resta (int a, int b)
{
return a - b;
}
int producto (int a, int b)
{
return (int) (a*b);
}
static int cociente (int int b)
return (int) (a/b);
```

Si tienes la implementación de resta y cociente en otro archivo, y las has declarado como static, no podrás acceder a ellas desde calculadora.c. Deberás asegurarte de que estas funciones estén definidas en el mismo archivo o, si están en otro archivo, eliminarlas de la declaración static.

Resta permanece static en el mismo menú y se quita del programa de funcEstatica.c

```
1 ##### calculadora.c #####
2 #include <stdio.h>
3 int suma (int,int);
4 static int resta(int,int);
5 int producto (int,int);
6 //static int cociente (int,int);
7 static int resta (int a, int b)
8 {
9 return a - b;
10 }
11 int main ()
12 {
13 printf("5 + 7 = %i\n",suma(5,7));
14 printf("9 - 77 = %d\n",resta(9,77));
15 printf("6 * 8 = %i\n",producto (6,8));
16 printf("7 / 2 = %d\n",cociente(7,2));
17 }
```

Cociente deja de ser static y se puede llamar desde el menú

```
1 //##### funcEstatica.c #####
2 #include <stdio.h>
3 int suma (int,int);
4 int producto (int,int);
5 //static
6 int cociente (int,int);
7 int suma (int a, int b)
8 {
9 return a + b;
10 }
11 int producto (int a, int b)
12 {
13 return (int) (a*b)
14 }
15 //static
16 int cociente (int a, int b)
17 {
18 return (int) (a/b);
19 }
```

**Tarea:**

## Conceptos:

- ¿Qué es una función?

R= Es un bloque de código que realiza una tarea específica. Las funciones son una herramienta fundamental, ya que permiten modularizar el código, dividirlo en partes más pequeñas y reutilizar fragmentos de código. En C, las funciones se declaran con un tipo de retorno (o sin él), un nombre, y opcionalmente, parámetros de entrada.

- Describe los 4 tipos de funciones 😞 función sin tipo de dato de retorno y sin parámetros de entrada, etc.)

R=

-Función sin valor de retorno y sin parámetros: Estas funciones no devuelven un valor (su tipo de retorno es void) y no reciben parámetros. Son útiles para realizar tareas que no requieren datos de entrada ni un resultado de salida, como mostrar mensajes en pantalla.

```
1 void saludo() {  
2     printf("¡Hola, mundo!\n");  
3 }  
4  
5
```

-Función sin valor de retorno y con parámetros: Estas funciones no devuelven ningún valor, pero sí reciben parámetros que usan para ejecutar una acción. Son comunes en situaciones donde se necesita procesar datos de entrada, como realizar una suma e imprimir el resultado.

```
1 void imprimirNumero(int numero) {  
2     printf("El número es: %d\n", numero);  
3 }  
4  
5  
6
```

-Función con valor de retorno y sin parámetros: Este tipo de función devuelve un valor, pero no recibe parámetros. Un ejemplo es una función que genera y devuelve un número aleatorio, donde no se necesita entrada del usuario, pero se espera un resultado.

```
1 int obtenerCinco() {  
2     return 5;  
3 }  
4
```

-Función con valor de retorno y con parámetros: Estas funciones tanto reciben parámetros como devuelven un valor. Son ideales para operaciones de procesamiento de datos donde se espera un resultado específico, como calcular la potencia de un número usando los valores base y exponente proporcionados.

```
1 int suma(int a, int b) {  
2     return a + b;  
3 }  
4
```



- Sintaxis de la función

R=

```
1 tipo_de_retorno nombre_funcion(tipo_parametro parametro1, tipo_parametro parametro2, ...)  
2 {  
3     // Cuerpo de la función  
4     // Código a ejecutar  
5     return valor; // Opcional, depende del tipo de retorno  
6 }  
7
```

- ¿Qué es la firma de la función?

R= La firma de una función define su nombre y los tipos de sus parámetros, sin incluir el cuerpo de la función. Esto permite que el compilador reconozca las funciones antes de que estén completamente definidas en el código. La firma ayuda a diferenciar entre funciones con el mismo nombre pero diferentes parámetros (sobrecarga).

```
1 int suma(int a, int b);  
2
```

Del ejercicio proporcionado se debe realizar lo siguiente:

- Completar ejercicio.
- 1.- Completa pseudocódigo y codificación.
  - 2.- Genera el mismo programa aplicando funciones, utiliza un valor inicial de n=10.

```

1 #include <stdio.h>
2
3 // Función para obtener la altura de la figura
4 int obtenerAltura() {
5     int n;
6     printf("Dame la altura: ");
7     scanf("%d", &n);
8     return n;
9 }
10
11 // Función para imprimir la figura
12 void imprimirFigura(int n) {
13     if (n > 0) {
14         // Imprimir la primera línea de la figura
15         for (int k = 1; k <= n - 1; k++) printf(" ");
16         printf("*\n");
17
18         // Imprimir el cuerpo de la figura
19         for (int k = 2; k <= n - 1; k++) {
20             for (int j = 1; j <= n - k; j++) printf(" ");
21             printf("*");
22             for (int j = 1; j <= 2 * k - 3; j++) printf(" ");
23             printf("*\n");
24         }
25
26         // Imprimir la última línea de la figura, si n > 1
27         if (n > 1) {
28             printf("*");
29             for (int k = 1; k <= 2 * n - 3; k++) printf(" ");
30             printf("*\n");
31         }
32     }
33 }
34
35 // Función principal
36 int main() {
37     int altura = obtenerAltura();
38     imprimirFigura(altura);
39     return 0;
40 }
41

```

3.-Compara y completa las instrucciones en el pseudocódigo, escribe en la cuadrícula cual es la salida de los algoritmos para un valor de n=10

INICIO

n ← obtenerAltura()

SI n > 0 ENTONCES

imprimirFigura(n)

FIN SI

FIN

FUNC obtenerAltura

ESCRIBIR "Dame la altura:"

LEER n

RETORNAR n

FIN FUNC

FUNC imprimirFigura(n)

SI n > 0 ENTONCES

// Imprimir primera línea

PARA k ← 1 HASTA n-1 HACER

ESCRIBIR " "

FIN PARA

```

ESCRIBIR "*"
ESCRIBIR "\n"

```

```

// Imprimir el cuerpo de la figura
PARA k ← 2 HASTA n-1 HACER
  PARA j ← 1 HASTA n-k HACER
    ESCRIBIR " "
  FIN PARA
  ESCRIBIR "*"
  PARA j ← 1 HASTA 2*k-3 HACER
    ESCRIBIR " "
  FIN PARA
  ESCRIBIR "*"
  ESCRIBIR "\n"
FIN PARA

```

```

// Imprimir última línea si n > 1
SI n > 1 ENTONCES
  ESCRIBIR "*"
  PARA k ← 1 HASTA 2*n-3 HACER
    ESCRIBIR " "
  FIN PARA
  ESCRIBIR "*"
  ESCRIBIR "\n"
FIN SI

```

```

FIN SI
FIN FUNC

```

Entrada							S	A	L	I	D	A							
Línea1										*									
Linea2									*		*								
Linea3								*				*							
Linea4							*						*						
Linea5						*								*					
Linea6					*										*				
Linea7				*												*			

Linea8			*														*		
Linea9		*																*	
Linea 10	*																		*

Define si la siguiente frase es verdadera o falsa:

Esta frase es falsa.

- Mediante prueba de escritorio dibujar la salida en el recuadro.
- Fragmentar el programa para emplear 3 funciones, una de ellas la función principal (main).

```

1  #include <stdio.h>
2
3  // Función para obtener la altura de la figura
4  int obtenerAltura() {
5      int n;
6      printf("Dame la altura: ");
7      scanf("%d", &n);
8      return n;
9  }
10
11 // Función para imprimir una línea con espacios y asteriscos
12 void imprimirLinea(int espacios, int asteriscos) {
13     for (int i = 0; i < espacios; i++) printf(" ");
14     for (int j = 0; j < asteriscos; j++) {
15         printf("*");
16         if (j < asteriscos - 1) printf(" "); // Espacio entre asteriscos
17     }
18     printf("\n");
19 }
20
21 // Función para imprimir la figura completa
22 void imprimirFigura(int n) {
23     if (n > 0) {
24         // Imprimir la primera línea de la figura
25         imprimirLinea(n - 1, 1);
26
27         // Imprimir el cuerpo de la figura
28         for (int k = 2; k <= n - 1; k++) {
29             imprimirLinea(n - k, 2);
30             for (int j = 1; j <= 2 * k - 3; j++) printf(" ");
31             printf("*\n");
32         }
33
34         // Imprimir la última línea de la figura, si n > 1
35         if (n > 1) {
36             imprimirLinea(0, 2 * n - 1);
37         }
38     }
39 }
40
41 // Función principal
42 int main() {
43     int altura = obtenerAltura();
44     imprimirFigura(altura);
45     return 0;
46 }
47

```

- Del programa fragmentado generar su codificación.(Diagrama de flujo y pseudocódigo).

(Pseudocódigo)

Función obtenerAltura()

```
Declarar n como entero
Imprimir "Dame la altura: "
Leer n
Retornar n
Fin Función
```

```
Función imprimirLinea(espacios, asteriscos)
  Para i desde 0 hasta espacios - 1
    Imprimir " "
  Fin Para
  Para j desde 0 hasta asteriscos - 1
    Imprimir "*"
    Si j < asteriscos - 1 Entonces
      Imprimir " " // Espacio entre asteriscos
    Fin Si
  Fin Para
  Imprimir nueva línea
Fin Función
```

```
Función imprimirFigura(n)
  Si n > 0 Entonces
    // Imprimir la primera línea de la figura
    imprimirLinea(n - 1, 1)

    // Imprimir el cuerpo de la figura
    Para k desde 2 hasta n - 1
      imprimirLinea(n - k, 2)
      Para j desde 1 hasta 2 * k - 3
        Imprimir " "
      Fin Para
      Imprimir "*"
      Imprimir nueva línea
    Fin Para

    // Imprimir la última línea de la figura, si n > 1
    Si n > 1 Entonces
      imprimirLinea(0, 2 * n - 1)
    Fin Si
  Fin Si
Fin Función
```

```
Función principal
```

```
Declarar altura como entero
altura ← obtenerAltura()
imprimirFigura(altura)
Fin Función
```

### **Conclusiones:**

En conclusión las funciones son una gran herramienta para crear un orden haciendo que un bloque de código realiza específica qué puede ser llamado desde cualquier parte del código lo que nos ayudará a hacer una reducción del trabajo y poco a poco crear un código de una manera eficiente para nuestro futuro en la ingeniería.

### **Bibliografía :**

Programiz. (s.f.). C Functions. Recuperado el 1 de noviembre de 2024, recuperado de <https://www.programiz.com/c-programming/c-functions>

Tutorial Gateway. (s.f.). Types of Functions in C Programming. Recuperado el 1 de noviembre de 2024, recuperado de <https://www.tutorialgateway.org/types-of-functions-in-c/>

Codeforwin. (s.f.). Types of functions in C. Recuperado el 1 de noviembre de 2024, recuperando de <https://codeforwin.org>