



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación

Salas A y B

Profesor: Marco Antonio Martínez Quintana

Asignatura: Fundamentos de programación

Grupo: 3

No. De Práctica(s): 12

Integrante(s): Emiliano Guevara Chávez

*No. de Equipo de
Cómputo empleado:* No aplica

No. De Lista o Brigada: 20

Semestre: Ciclo 2021-1

Fecha de entrega:

Observaciones:

CALIFICACIÓN: _____

Práctica #12:

Funciones

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.

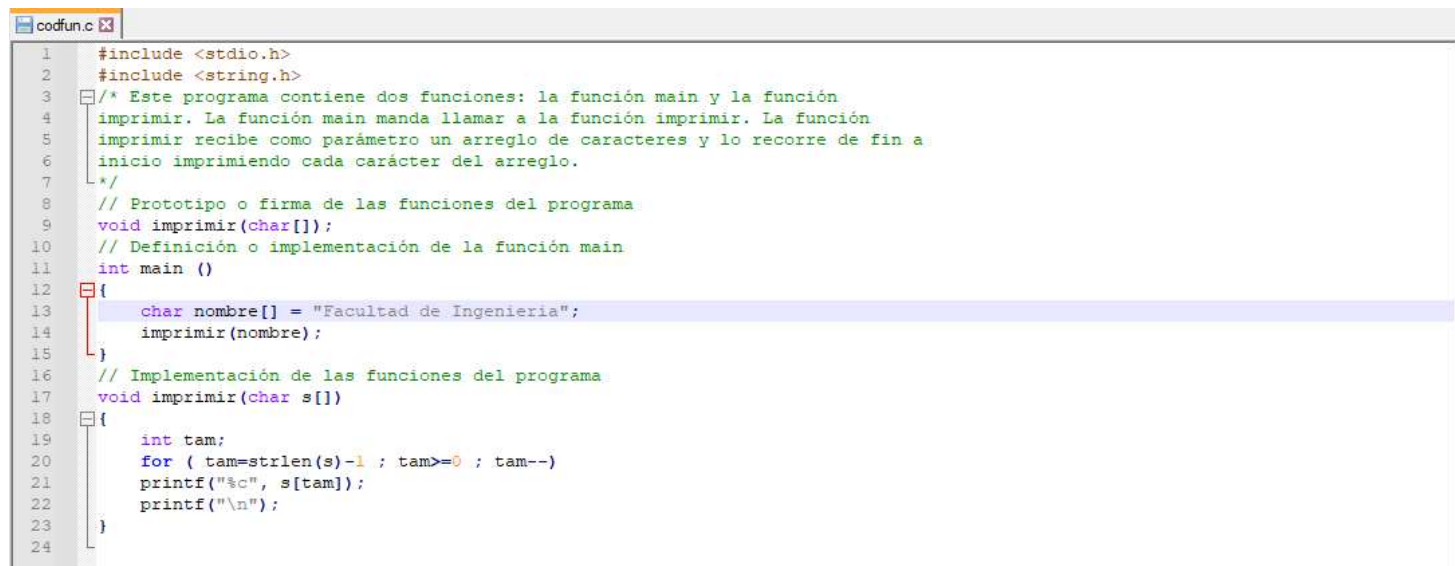
Introducción:

Un programa en lenguaje C consiste en una o más funciones. C permite tener dentro de un archivo fuente varias funciones, esto con el fin de dividir las tareas y que sea más fácil la depuración, la mejora y el entendimiento del código. En lenguaje C la función principal se llama main. Cuando se ordena la ejecución del programa, se inicia con la ejecución de las instrucciones que se encuentran dentro de la función main, y ésta puede llamar a ejecutar otras funciones, que a su vez éstas pueden llamar a ejecutar a otras funciones, y así sucesivamente.

Desarrollo de actividades

Funciones

Código (funciones)



```
1  #include <stdio.h>
2  #include <string.h>
3  /* Este programa contiene dos funciones: la función main y la función
4  imprimir. La función main manda llamar a la función imprimir. La función
5  imprimir recibe como parámetro un arreglo de caracteres y lo recorre de fin a
6  inicio imprimiendo cada carácter del arreglo.
7  */
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 {
13     char nombre[] = "Facultad de Ingenieria";
14     imprimir(nombre);
15 }
16 // Implementación de las funciones del programa
17 void imprimir(char s[])
18 {
19     int tam;
20     for ( tam=strlen(s)-1 ; tam>=0 ; tam--)
21         printf("%c", s[tam]);
22     printf("\n");
23 }
24
```

```
C:\Users\user\Desktop\Lenguaje C\Ejemplos>gcc codfun.c -o codfun.exe
C:\Users\user\Desktop\Lenguaje C\Ejemplos>codfun.exe
aireinegnI ed datlucaF
C:\Users\user\Desktop\Lenguaje C\Ejemplos>_
```

Ámbito o alcance de las variables

Código (ámbito de las variables)

```
1 #include <stdio.h>
2 /*
3  Este programa contiene dos funciones: la función main y la función incremento. La
4  función main manda llamar a la función incremento dentro de un ciclo for. La función
5  incremento aumenta el valor de la variable enteraGlobal cada vez que es invocada.
6 */
7 void incremento();
8 // La variable enteraGlobal es vista por todas
9 // las funciones (main e incremento)
10 int enteraGlobal = 0;
11 int main()
12 {
13     // La variable cont es local a la función main
14     for (int cont=0 ; cont<5 ; cont++)
15     {
16         incremento();
17     }
18     return 999;
19 }
20 void incremento()
21 {
22     // La variable enteraLocal es local a la función incremento
23     int enteraLocal = 5;
24     enteraGlobal += 2;
25     printf("global(%i) + local(%i) = %d\n", enteraGlobal, enteraLocal,
26           enteraGlobal+enteraLocal);
27 }
28 }
```

```
C:\Users\user\Desktop\Lenguaje C\Ejemplos>gcc codambiva.c -o codambiva.exe
```

```
C:\Users\user\Desktop\Lenguaje C\Ejemplos>codambiva.exe
```

```
global(2) + local(5) = 7
global(4) + local(5) = 9
global(6) + local(5) = 11
global(8) + local(5) = 13
global(10) + local(5) = 15
```

```
C:\Users\user\Desktop\Lenguaje C\Ejemplos>
```

Argumentos para la función main

Código (argumentos función main)

```
1 #include <stdio.h>
2 #include <string.h>
3 /*
4  Este programa permite manejar los argumentos enviados al ejecutarlo.
5 */
6 int main (int argc, char** argv)
7 {
8     if (argc == 1)
9     {
10         printf("El programa no contiene argumentos.\n");
11         return 88;
12     }
13
14     printf("Los elementos del arreglo argv son:\n");
15     for (int cont = 0 ; cont < argc ; cont++ )
16     {
17         printf("argv[%d] = %s\n", cont, argv[cont]);
18     }
19
20     return 88;
21 }
```

```
C:\Users\user\Desktop\Lenguaje C\Ejemplos>gcc cofuma.c -o cofuma.exe
```

```
C:\Users\user\Desktop\Lenguaje C\Ejemplos>cofuma.exe
El programa no contiene argumentos.
```

```
C:\Users\user\Desktop\Lenguaje C\Ejemplos>cofuma.exe 4
Los elementos del arreglo argv son:
argv[0] = cofuma.exe
argv[1] = 4
```

```
C:\Users\user\Desktop\Lenguaje C\Ejemplos>cofuma.exe 80
Los elementos del arreglo argv son:
argv[0] = cofuma.exe
argv[1] = 80
```

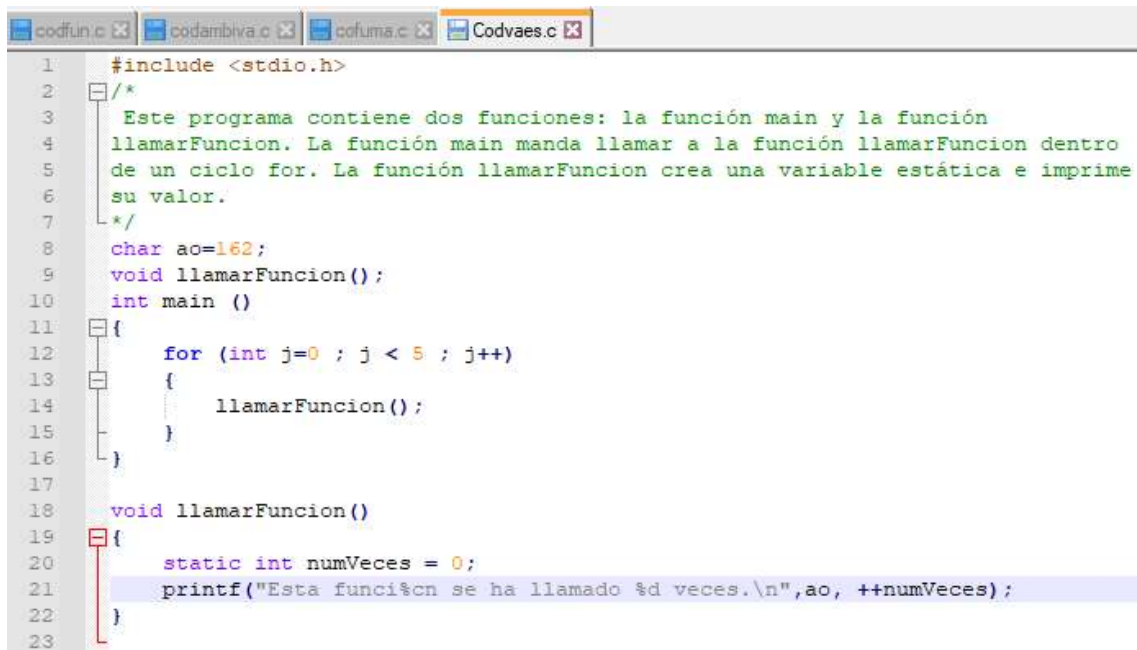
```
C:\Users\user\Desktop\Lenguaje C\Ejemplos>cofuma.exe 1
Los elementos del arreglo argv son:
argv[0] = cofuma.exe
argv[1] = 1
```

```
C:\Users\user\Desktop\Lenguaje C\Ejemplos>6
"6" no se reconoce como un comando interno o externo,
programa o archivo por lotes ejecutable.
```

```
C:\Users\user\Desktop\Lenguaje C\Ejemplos>
```

Estático

Código (variable estática)



```
1  #include <stdio.h>
2  /*
3   Este programa contiene dos funciones: la función main y la función
4   llamarFuncion. La función main manda llamar a la función llamarFuncion dentro
5   de un ciclo for. La función llamarFuncion crea una variable estática e imprime
6   su valor.
7  */
8  char ao=162;
9  void llamarFuncion();
10 int main ()
11 {
12     for (int j=0 ; j < 5 ; j++)
13     {
14         llamarFuncion();
15     }
16 }
17
18 void llamarFuncion()
19 {
20     static int numVeces = 0;
21     printf("Esta funci%cn se ha llamado %d veces.\n",ao, ++numVeces);
22 }
23
```

```

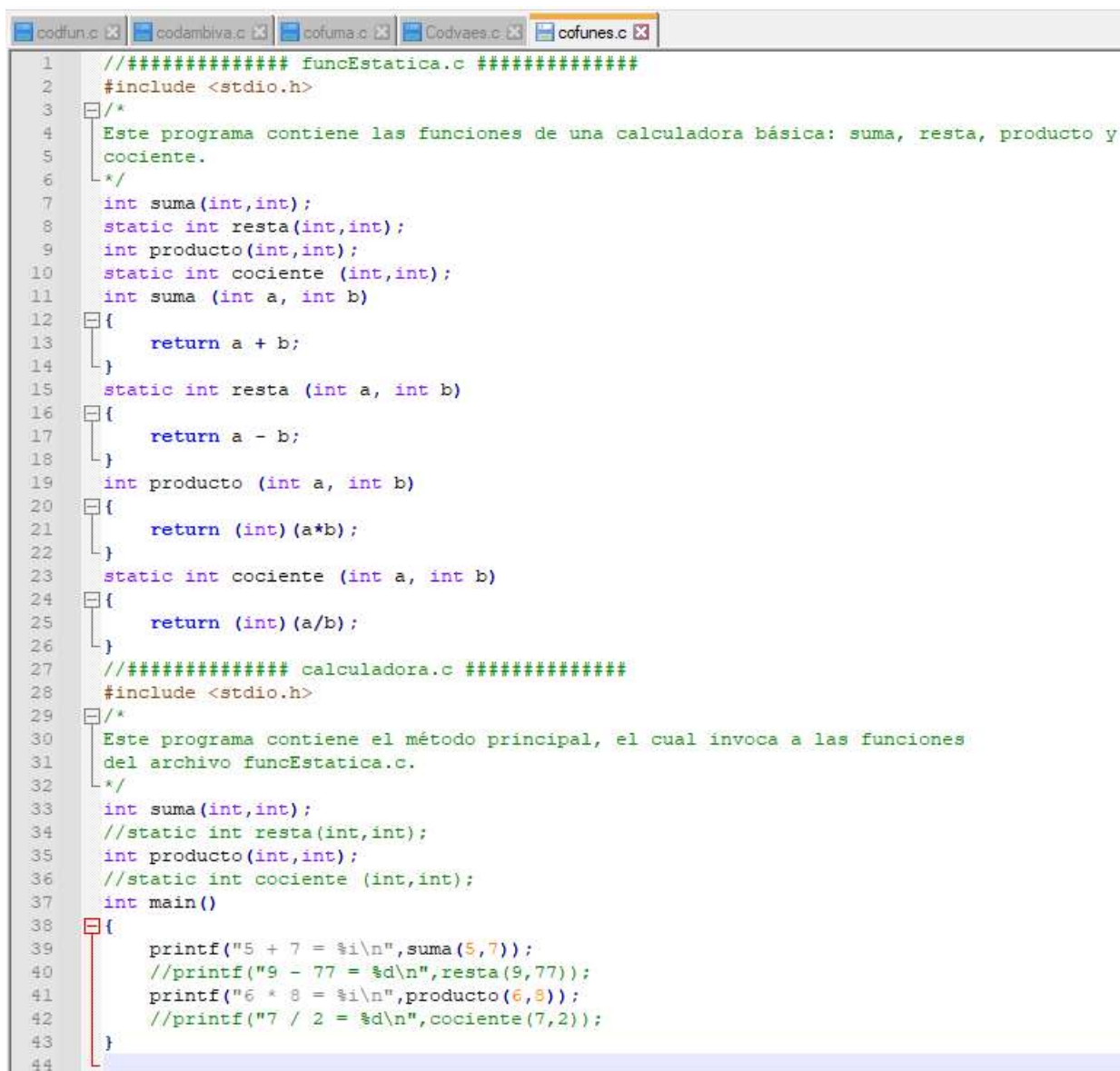
C:\Users\user\Desktop\Lenguaje C\Ejemplos>gcc Codvaes.c -o Codvaes.exe

C:\Users\user\Desktop\Lenguaje C\Ejemplos>Codvaes.exe
Esta función se ha llamado 1 veces.
Esta función se ha llamado 2 veces.
Esta función se ha llamado 3 veces.
Esta función se ha llamado 4 veces.
Esta función se ha llamado 5 veces.

C:\Users\user\Desktop\Lenguaje C\Ejemplos>_

```

Código (función estática)



```

1  //##### funcEstatica.c #####
2  #include <stdio.h>
3  /*
4  Este programa contiene las funciones de una calculadora básica: suma, resta, producto y
5  cociente.
6  */
7  int suma(int,int);
8  static int resta(int,int);
9  int producto(int,int);
10 static int cociente (int,int);
11 int suma (int a, int b)
12 {
13     return a + b;
14 }
15 static int resta (int a, int b)
16 {
17     return a - b;
18 }
19 int producto (int a, int b)
20 {
21     return (int) (a*b);
22 }
23 static int cociente (int a, int b)
24 {
25     return (int) (a/b);
26 }
27 //##### calculadora.c #####
28 #include <stdio.h>
29 /*
30 Este programa contiene el método principal, el cual invoca a las funciones
31 del archivo funcEstatica.c.
32 */
33 int suma(int,int);
34 //static int resta(int,int);
35 int producto(int,int);
36 //static int cociente (int,int);
37 int main()
38 {
39     printf("5 + 7 = %i\n",suma(5,7));
40     //printf("9 - 77 = %d\n",resta(9,77));
41     printf("6 * 8 = %i\n",producto(6,8));
42     //printf("7 / 2 = %d\n",cociente(7,2));
43 }
44

```

```
C:\Users\user\Desktop\Lenguaje C\Ejemplos>gcc cofunes.c -o cofunes.exe  
C:\Users\user\Desktop\Lenguaje C\Ejemplos>cofunes.exe  
5 + 7 = 12  
6 * 8 = 48  
C:\Users\user\Desktop\Lenguaje C\Ejemplos>
```

Conclusión

En conclusión me di cuenta de que, la función es una parte muy importante, ya que puedes llevar acabo líneas de código muy complejas, pero que a la vez te puedan dar algo muy específico o hacerte un proceso que se repite en cierto proceso de manera independiente.