



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: César Fabián Domínguez Velasco

Asignatura: Fundamentos de Programación

No. de práctica(s): 11

Integrante(s): 13_García_Sánchez_Alejandro
18_Lopez_Castro_Anastacia
34_Ramirez_Rivas_Gael
39_Ruiz_Hernandez_Ruben_Antonio

No. de lista o brigada: 1A

Semestre: 2024-2

Fecha de entrega: 22 de Mayo del 2024

Observaciones:

CALIFICACIÓN: _____

PRÁCTICA 11: FUNCIONES

1) 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.

2) Introducción:

Un programa en lenguaje C consiste en una o más funciones, donde C permite tener dentro de un archivo fuente varias funciones, esto con el fin de dividir las tareas con el fin de que sea más fácil la depuración, la mejora y el entendimiento del código; por ejemplo, en lenguaje C la función principal se llama main. En cambio, cuando se ordena la ejecución del programa, se inicia con la ejecución de las instrucciones que se encuentran dentro de la función main, ya que puede llamar a ejecutar otras funciones, que a su vez éstas pueden llamar a ejecutar a otras funciones, y así sucesivamente. A continuación, realizaremos la construcción, pero sobre todo hacer tanto la ejecución como compilación.

3) Desarrollo (Capturas de pantalla de los programas en C):

→ Programa 1.c

```
Restricted Mode is intended for safe code browsing. Trust this folder to enable all features. Manage Learn More

Welcome C programa1a.c

C programa1a.c
1 //Programa 1.c: Funciones//
2
3 #include <stdio.h>
4 #include <string.h>
5
6 printf("Programa 1.c: Funciones.\n");
7
8 printf("\nEste programa contiene dos funciones: la función main y la función imprimir.\n");
9 printf("\nLa función main manda llamar a la función imprimir. La función imprimir recibe como\n");
10 printf("\nparámetro un arreglo de caracteres y lo recorre de fin a inicio imprimiendo\n");
11 printf("\ncada carácter del arreglo\n");
12
13 // Prototipo o firma de las funciones del programa
14
15 void imprimir(char[]);
16
17 // Definición o implementación de la función main
18
19 int main ()
20 {
21
22
23     {
24
25         char nombre[] = "Facultad de Ingeniería";
26         imprimir(nombre);
27     }
28
29 }
30
31 // Implementación de las funciones del programa
32
33 void imprimir(char s[])
34 {
35     {
36
37         int tam;
38
39         for ( tam=strlen(s)-1 ; tam>=0 ; tam--)
40
41             printf("%c", s[tam]);
42
43             printf("\n");
44
45         }
46
47     }
48
49     return 0;
50 }
```



ANSI C

| main.c | Console |
|-----------------------|---|
| 1 //Programa 1.c: Fu | Programa 1.c: Funciones. |
| 2 | |
| 3 #include <stdio.h> | Este programa contiene dos funciones: la función main y la función imprimir. |
| 4 #include <string.h> | |
| 5 | |
| 6 // Prototipo o fir | La función main manda llamar a la función imprimir. La función imprimir recibe como |
| 7 void imprimir(char | parámetro un arreglo de caracteres y lo recorre de fin a inicio imprimiendo |
| 8 | |
| 9 // Definición o im | |
| 10 int main (){ | cada carácter del arreglo |
| 11 | a |
| 12 printf("Progra | ad |
| 13 | |

→ Programa 2.c:

```
main.c
1  #include <stdio.h>
2
3  void sumar() // función suma
4  {
5      int x=5, y=10, z; //variables locales
6      z=x+y;
7      printf("%i",z);
8  }
9
10 int main()
11 {
12     sumar(); // llamado de la función suma
13 }
```

input

15

...Program finished with exit code 0
Press ENTER to exit console. □

→ Programa 3.c:

```
OPENEDG Run ANSI C
main.c Console
1 //Programa 3.c: Multiplicación de Función//
2
3 #include <stdio.h>
4
5 int resultado; //variable global
6
7 void multiplicar() //función multiplicar
8 {
9 {
10
11     resultado = 5 * 4;
12 }
13 }
14
15 int main()
16 {
17 {
18     printf("Programa 3.c: Multiplicación de Función.\n");
19
20     printf("\nEste programa muestra la declaración de la variable global resultado,\n");
21     printf("\nla cual es utilizada en ambas funciones.\n");
22
23     multiplicar(); //llamado de la función multiplicar
24
25     printf("%i\n",resultado);
26
27     return 0;
28 }
29 }
30 }
```

Programa 3.c: Multiplicación de Función.

Este programa muestra la declaración de la variable global resultado,
la cual es utilizada en ambas funciones.
20

→ Programa 4.c:

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

```
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

...Program finished with exit code 231
Press ENTER to exit console.
```

→ Programa 5.c:

```
OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
Learn Python with
KodeKloud

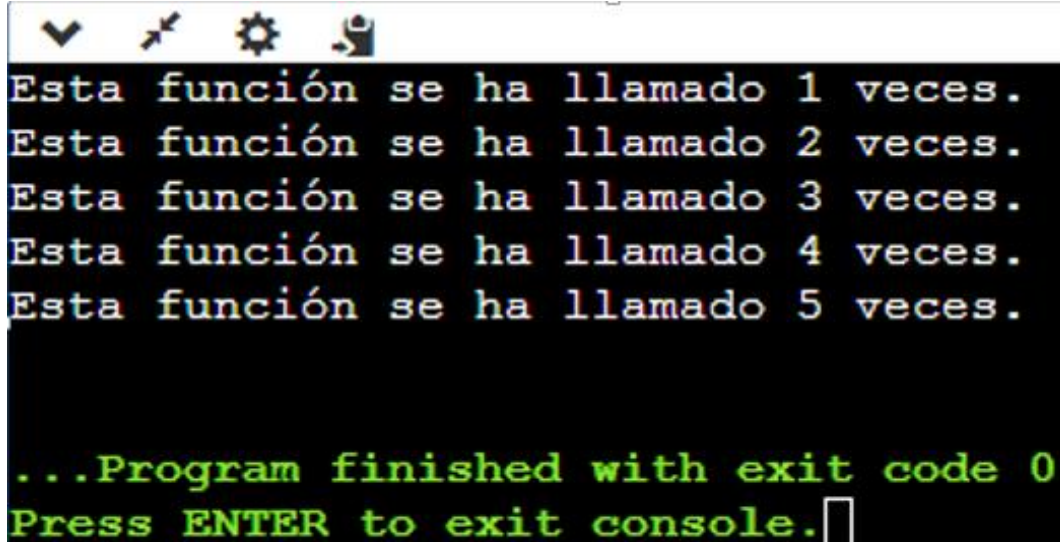
main.c
1  #include <stdio.h>
2  #include <string.h>
3  int main (int argc, char** argv)
4  {
5      if (argc == 1)
6      {
7          printf("El programa no contiene argumentos.\n");
8          return 88;
9      }
10
11     printf("Los elementos del arreglo argv son:\n");
12     for (int cont = 0 ;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

```
El programa no contiene argumentos.

...Program finished with exit code 88
Press ENTER to exit console.
```

→ Programa 6.c:

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



```
✓ ↗ ⚙ 🖨
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.

...Program finished with exit code 0
Press ENTER to exit console. □
```

```
#include <stdio.h>
void llamarFuncion();
int main ()
{
    for (int j=0 ; j < 5 ; j++)
    {
        llamarFuncion();
    }
}
void llamarFuncion()
{
    /* Solo la primera vez que se llame a esta función se creará y se le asignará
    el valor de 0 a la variable estática numVeces */
    static int numVeces = 0;
    printf("Esta función se ha llamado %d veces.\n", ++numVeces);
}
```

✓ ✂ ⚙ 📄

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.

...Program finished with exit code 0
Press ENTER to exit console.

```
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 b)
16 {
17     return (int)(a*b);
18 }
19 static int cociente (int a, int b)
20 {
21     return (int)(a/b);
22 }
```

```
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 }
14
```

$$5 + 7 = 12$$

$$6 * 8 = 48$$

4) Conclusiones (Individuales):

- ❖ **García Sánchez Alejandro:** El programa 6 consistió en la aplicación de dos funciones, la función “main” y “llamar función”, básicamente para llamar una función que genera una variable estática dentro del ciclo for, comenzando con un determinado valor y a partir de la iteración ir incrementando ese valor.
- ❖ **López Castro Anastacia:** Las funciones static en C tienen un alcance de archivo, lo que significa que solo son visibles y utilizables dentro del archivo en el que se definen. Esto es útil para encapsular la funcionalidad y prevenir que otras partes del programa accedan directamente a estas funciones. En este caso, resta y cociente son funciones que solo se pueden utilizar dentro de funcEstatica.c
- ❖ **Ramírez Rivas Gael:** El estudio de las funciones en el lenguaje de programación C es fundamental para la resolución de problemas y la creación de algoritmos eficientes. A lo largo de este trabajo, hemos analizado diferentes tipos de funciones en C, como las funciones sin parámetros, con parámetros, con valores de retorno y sin valores de retorno.
- ❖ **Ruíz Hernández Rubén Antonio:** Desde mi punto de vista aprendí para el programa 1a.c usar adecuadamente el manejo de la función void imprimir() en el lenguaje C se refiere a una función que no devuelve un valor y se utiliza para realizar una acción específica, en este caso, imprimir caracteres formateados en la salida estándar, mientras que para el programa 3a.c si deseo utilizar el resultado de la multiplicación en otro lugar de tu código, debo modificar la función void multiplicar() para que devuelva un valor en lugar de imprimirlo.

5) Retroalimentación (Equipo):

Se cumplieron al pie de la letra los objetivos de esta última práctica ya que elaboramos programas en C donde la solución del problema se dividió en funciones, donde distinguimos lo que es el prototipo o firma de una función e implementación de ella, así como manipular parámetros tanto en la función principal como en otras. Concluimos que las funciones son una parte esencial del lenguaje de programación C incluyendo la ayuda para escribir código más organizado, modular y reutilizable, de igual manera al dividir el código en partes más pequeñas, puede hacerlo más fácil de entender, probar y mantener.

6) Fuentes en APA:

- ★ Laboratorio Salas A y B. (s.f.). *Manual de Prácticas de la Asignatura Fundamentos de Programación (Guía práctica de estudio 11: Funciones, pág. 164 - 174)*. Recuperado el 17 de Mayo del 2024, de Laboratorio de Computación Salas A y B: <http://lcp02.fi-b.unam.mx/>
- ★ RuyAntonio. (s.f.). *GitHub - RuyAntonio/practica11_fdp: Práctica 11: Funciones*, de GitHub. Recuperado el 17 Mayo del 2024, de GitHub: https://github.com/RuyAntonio/practica11_fdp.git