



## Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

# Laboratorios de computación salas A y B

Alejandro E. Pimentel Alarcón

*Profesor:*

Fundamentos de Programación

*Asignatura:*

3; Bloque:135

*Grupo:*

7

*No de Práctica(s):*

Bazaldúa Morales Vanessa

*Integrante(s):*

*No. de Equipo de  
cómputo*

47-Tailandia

#6

*No. de Lista o*

2020-1

*Semestre:*

03/Oct/19

*Fecha de entrega:*

*Observaciones:*

**CALIFICACIÓN:** \_\_\_\_\_

# FUNDAMENTOS DE LENGUAJE C

## OBJETIVO:

Elaborar programas en lenguaje C utilizando las instrucciones de control de tipo de secuencia, para realizar la declaración de variables de diferentes tipos de datos, así como efectuar llamadas a funciones externas de entrada y salida para asignar y mostrar valores de valores y expresiones.

## Tipos de variables

A continuación se muestra una tabla con los tipos de variables y la cantidad de espacio (en Bits) que utilizan ya que, en lenguaje C se definen por el espacio que utilizan (dependiendo que tan grande o pequeño sea tu variable).

DATA TYPE	MEMORY (BYTES)	RANGE
short int	2	-32,768 to 32,767
unsigned short int	2	0 to 65,535
unsigned int	4	0 to 4,294,967,295
int	4	-2,147,483,648 to 2,147,483,647
long int	4	-2,147,483,648 to 2,147,483,647
unsigned long int	4	0 to 4,294,967,295
long long int	8	-(2 <sup>63</sup> ) to (2 <sup>63</sup> )-1
unsigned long long int	8	0 to 18,446,744,073,709,551,615

Para los reales, se tienen también diferentes tipos de variables que asignan más bits para tener mayor rango y mayor precisión. Las variables reales siempre poseen signo.

Tipo	Bits	Valor Mínimo	Valor Máximo
<i>float</i>	32	3.4 E-38	3.4 E38
<i>double</i>	64	1.7 E-308	1.7 E308
<i>long double</i>	80	3.4 E-4932	3.4 E4932

En esta actividad de laboratorio al principio fue un poco de teoría acerca de como se manejaban los valores y las variables en este tipo de lenguaje, sin embargo, con las que trabajaremos de forma más común es con float y double. Aunque dependiendo el caso será el tipo de variable que utilizaremos. Iniciamos los ejercicios prácticos solamente “jugando” en un programa llamado «sublime» text para ver cómo funcionaba y que nos sintiéramos cómodos utilizándolo.

```

primero.c
UNREGISTERED

primero.c
1 int main() {
2
3     // Variables enteras
4     short numeroEntero1;
5     signed int numeroEntero2;
6     unsigned long numeroEntero3;
7
8     // Caracter
9     char caracter;
10
11    //Variables reales
12    float puntoFlotante1;
13    double puntoFlotante2;
14
15    return 0;
16 }

```

## Mostrar y leer

Para especificar el tipo de variable que vas a utilizar dentro de tu programa lo debes expresar de la siguiente manera, igual dependiendo de cual vayas a utilizar.

Tipo de dato	Especificador de formato
Entero	%d, %i, %ld, %li, %o, %x
Flotante	%f, %lf, %e, %g
Carácter	%c, %d, %i, %o, %x
Cadena de caracteres	%s

```

1 #include <stdio.h>
2 int main () {
3     //Declaramos variables a leer
4     int numeroEntrada;
5     double realEntrada;
6
7     // Asignamos variables
8     int numeroEntero = 32768;
9     char caracter = 'B';
10    float numeroReal = 89.8;
11
12    // Mostramos texto y valores
13    printf ("Primero texto solo\n");
14    printf("Luego podemos poner un entero: %i/n", numeroEntero);
15    printf("Y un numero real: %.2f\n", numeroReal);
16
17    //Leemos valores
18    scanf("%i", &numeroEntrada);
19    scanf("%If", &realEntrada);
20
21    // Y ahora podemos mostrarlos también
22    printf ("Tu entero: %i\n", numeroEntrada);
23    printf("Tu real: %.3If\n", realEntrada);
24
25    return 0;
26 }
27

```

Después de jugar un poco con «sublime text» entonces empezamos con algo más interesante, ahora ya teníamos que compilar y correr el programa(para ver que en verdad funcione); igual de inicio son ejercicios sencillos y solo teníamos que seguir los pasos tal como lo indicaba el profesor para que todo saliera correcto, este fue el primero que compilamos y luego corrimos.

Para compilar se realiza el comando:

- gcc main.c -o main

y para correr el programa es con:

- ./main

También nos menciono que la línea al principio se trata de una librería básica (existen una gran variedad de librerías, dependiendo del programa que vayas a realizar), debido a que es necesario para poder realizar las operaciones.

```
Last login: Mon Sep 30 09:32:53 on console
[Tailandia47:~ fp03alu06$ gcc vane.c -o main
clang: error: no such file or directory: 'vane.c'
clang: error: no input files
[Tailandia47:~ fp03alu06$ ls
Desktop      Downloads    Movies       Pictures
Documents    Library      Music        Public
[Tailandia47:~ fp03alu06$ cd Desktop
[Tailandia47:Desktop fp03alu06$ cd
[Tailandia47:Desktop fp03alu06$ ls
Desktop      Downloads    Movies       Pictures
Documents    Library      Music        Public
[Tailandia47:~ fp03alu06$ cd Desktop
[Tailandia47:Desktop fp03alu06$ ls
#include <stdio.h>
[Tailandia47:Desktop fp03alu06$ ls
vane.c
[Tailandia47:Desktop fp03alu06$ gcc vane.c -o main
vane.c:19:10: warning: length modifier 'I' results in undefined behavior or no
effect with 'f' conversion specifier [-Wformat]
scanf("%If", &realEntrada);
      ~~~~^
vane.c:23:22: warning: length modifier 'I' results in undefined behavior or no
effect with 'f' conversion specifier [-Wformat]
printf("Tu real: %.3If\n", realEntrada);
                   ~~~~^

2 warnings generated.
[Tailandia47:Desktop fp03alu06$ ./main
Primer texto solo
Luego podemos poner un entero: 32768/nY un numero real: 89.80
█
```

Operador	Operación	Uso	Resultado
+	Suma	125.78 + 62.5	188.28
-	Resta	65.3 - 32.33	32.97
*	Multiplicación	8.27 * 7	57.75
/	División	15 / 4	3.75
%	Módulo	4 % 2	0

## Operadores:

Estos como su nombre lo indica son los símbolos que utilizaremos para especificar qué operación queremos hacer, puede ser una suma, resta, división, etc.

Para comprobarlos escribimos lo siguiente en nuestro editor de código. Y luego lo ejecutamos en nuestra terminal:

```
~/Escritorio/operadores.c - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

operadores.c
1 #include <stdio.h>
2
3 int main() {
4
5     int dos, tres, cuatro, cinco;
6     double resultado;
7
8     dos = 2;
9     tres = 3;
10    cuatro = 4;
11    cinco = 5;
12
13    resultado = cinco/dos;
14    printf("5/2 = %.1lf\n", resultado);
15
16    resultado = (double)cinco/dos;
17    printf("5/2 = %.1lf\n", resultado);
18
19    return 0;
20 }
```

```
vanessa@Titan: ~/Escritorio
Archivo Editar Ver Buscar Terminal Ayuda
Descargas examples.desktop Música 'Sin título 2.odt'
Documentos FP_2020-1_8166 Plantillas snap
Escritorio Imágenes Público Vídeos

vanessa@Titan:~$ cd Escritorio
vanessa@Titan:~/Escritorio$ ls}

Orden «ls» no encontrada. Quizá quiso decir:

la orden «lsd» del paquete snap «lsd (0.16.0)»
la orden «lsx» del paquete deb «suckless-tools»
la orden «lsh» del paquete deb «lsh-client»
la orden «ls» del paquete deb «coreutils»
la orden «lsm» del paquete deb «lsm»
la orden «lsc» del paquete deb «livescript»

Consulte «snap info <nombre del snap>» para ver más versiones.

vanessa@Titan:~/Escritorio$ ls
FP_2020-1_8166 main operadores.c PRÁCTICA_6FDP.pdf primero.c
vanessa@Titan:~/Escritorio$ gcc operadores.c -o operadores
vanessa@Titan:~/Escritorio$ ./operadores
5/2 = 2.0
5/2 = 2.5
vanessa@Titan:~/Escritorio$
```

## Comparaciones:

De igual manera tenemos las comparaciones que, en realidad no son más que operadores como veremos a continuación:

Operador	Operación	Uso	Resultado
==	Igual que	'h' == 'H'	Falso
!=	Diferente a	'a' != 'b'	Verdadero
<	Menor que	7 < 15	Verdadero
>	Mayor que	11 > 22	Falso
<=	Menor o igual	15 <= 22	Verdadero
>=	Mayor o igual	20 >= 35	Falso

## Operadores lógicos:

Así como los operadores lógicos.

Operador	Operación
!	No
&&	Y
	O

Para ver mejor su funcionalidad realizamos la siguiente actividad:

```
~/Escritorio/logicos.c - Sublin
File Edit Selection Find View Goto Tools Project Preferences Help
operadores.c x logicos.c x
1 #include <stdio.h>
2
3 int main() {
4
5     int num1, num2, res;
6     char c1, c2;
7
8     num1 = 7;
9     num2 = 15;
10    c1 = 'h';
11    c2 = 'H';
12
13    printf("¿ num1 es menor a num2 ? -> %d\n", num1<num2);
14    printf("¿ c1 es igual a c2 ? -> %d\n", c1==c2);
15    printf("¿ c1 es diferente a c2 ? -> %d\n", c1!=c2);
16
17    res = num1 < num2 && c1 == 'h';
18    printf("¿ num 1 < num2 Y c1 es igual a 'h' ? -> %d\n", res);
19
20    res = c1 == 's' || c2 == 'H';
21    printf("¿ c1 es igual a 's' O c2 a 'H' ? -> %d\n", res);
22
23    return 0;
24 }
```

```
vanessa@Titan: ~/Escritorio
Archivo Editar Ver Buscar Terminal Ayuda
vanessa@Titan:~$ ls
Descargas  examples.desktop  Música  'Sin titulo 2.odt'
Documentos FP_2020-1_8166   Plantillas snap
Escritorio Imágenes        Público  Videos
vanessa@Titan:~$ cd Escritorio
vanessa@Titan:~/Escritorio$ ls
FP_2020-1_8166 main operadores.c primero.c
logicos.c      operadores PRÁCTICA_6FDP.pdf
vanessa@Titan:~/Escritorio$ gcc logicos.c -o logicos
vanessa@Titan:~/Escritorio$ ./logicos
¿ num1 es menor a num2 ? -> 1
¿ c1 es igual a c2 ? -> 0
¿ c1 es diferente a c2 ? -> 1
¿ num 1 < num2 Y c1 es igual a 'h' ? -> 1
¿ c1 es igual a 's' O c2 a 'H' ? -> 1
vanessa@Titan:~/Escritorio$
```

## Conclusión:

En conclusión podría decir que, es increíble la cantidad de operaciones y razonamientos lógicos que tienes que arreglar y analizar para ejecutar algo tan simple como una suma por ejemplo, pero aun así se me hace bastante interesante, sin duda me ha costado por no saber nada acerca de programación, peor ahora que ya voy conociendo más acerca del mundo virtual, me ha llamado mucho la atención. No deja de sorprenderme cada día con la cantidad de datos, conceptos, operadores, etc con los que tienes que lidiar, trato de entender todo lo que vemos y realizamos, sin embargo, hay algunas cosas que se me han dificultado más pero todo con tal de seguir aprendiendo. Y en un futuro ser una buena ingeniera.