	<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:

No. de Práctica: 6

Integrantes: Ever Ivan Rosales Gómez

No de Lista o brigade:

Semestre: 2023-2

Fecha de entrega: 12/04/23

Observaciones:

Calificación

## ENTORNO Y FUNDAMENTOS DEL LENGUAJE C

### OBJETIVOS:

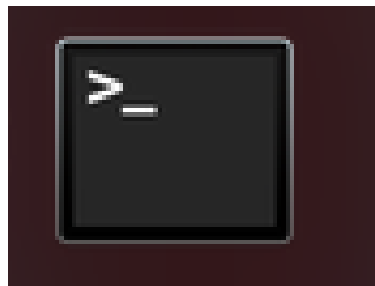
Elaborar programas en lenguaje C utilizando instrucciones de control de secuencia para la resolución de estos.

### DESARROLLO:

Para iniciar es importante mencionar qué es un editor de texto. Un editor de texto es aquel que únicamente pone textos sin formatos, comúnmente son conocidos como archivos de texto y el más conocido es BLOC DE NOTAS.

Se utiliza un editor de texto porque es fácil abrirlo en otra computadora o verlos en otro sitio de manera sencilla.

En esta práctica el editor de texto que se usará es la TERMINAL.

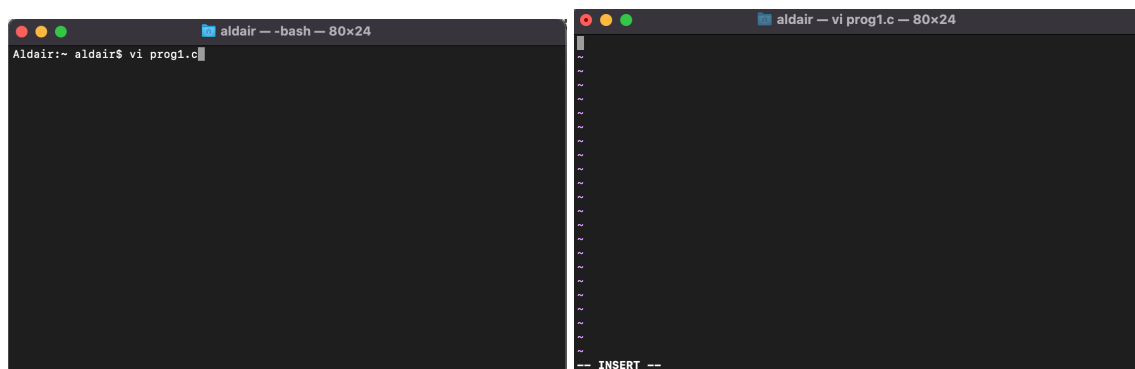


Se va a crear un archivo para poder modificarlo en Lenguaje C. Para crear un archivo en este lenguaje es importante considerar dos cosas, cómo inicia y cómo termina:

INICIA	NOMBRE	FIN
vi	nombre que queramos	.c





Como se observa en la tabla todo formato en lenguaje C al usar la terminal comienza con vi y finaliza con .c. En este caso se van a crear 3 archivos para poder hacer 3 ejercicios en cada uno.

El nombre del primer archivo será “prog1”.



Al dar enter la terminal arroja directamente al archivo para poder editarlo.

Antes de comenzar a escribir los primeros programas es importante saber los comandos que se pueden usar y que el pseudocódigo sea más fácil de manejar. Es importante mencionar que para usar los comandos debes teclear primero la tecla “esc” como sistema de bloqueo para escribir.

k o 	Mueve el cursor arriba, sirve para moverse en el eje “Y” positivo
j o 	Mueve el cursor abajo, sirve para moverse en el “Y” negativo
o o 	Mueve el cursor al lado izquierdo
l o 	Mueve el cursor al lado derecho
1G	Sirve para regresar al inicio de la primera línea
G	Sirve para ir a la última línea
x	Es como usar control + x, borra lo que marca la letra del cursor
dd	Borra toda la fila del cursor
ndd	N = Número de filas que se desean eliminar, estas se eliminarán a partir del cursor
yy	Sirve para copiar la fila donde está el curso
p	Pega o borra un bloque de contenido
u	Deshace el último cambio

Cuando se quiera salir el archivo se realizan los siguientes pasos:

-Presionar la tecla esc que está arriba del tabulador.

-Teclear : (dos puntos) y agregar wq.

La función :wq realiza dos procesos, guardar y salir el proceso.

Por último, un proceso de gran importancia es **compilar**, para entender esta palabra cuando una persona está realizando un pseudocódigo lo entiende y lo explica de manera que otras personas lo entiende, pero una computadora lo entiende a su manera, de manera muy diferente a la de una persona. La forma en que entiende la información es con ayuda del código binario (con números 1 o 0). El **compilador** se encarga de pasar el pseudocódigo a código binario para que la computadora pueda realizar las instrucciones y pueda funcionar y ejecutarse. La compilación debe tener un orden así como cuando se crea una archivo.

En la terminal se agrega lo siguiente:

INICIA	NOMBRE	FIN	
gcc	nombre del archivo	(tecla de espacio)- o nombre del archivo.out	./nombre del archivo



En la imagen se muestran 3 pasos fundamentales para la resolución de un problema de programación. Es importante tener un panorama para resolver el problema, posteriormente armar el pseudocódigo y finalmente compilador para hacer las pruebas de escritorio para finalizar con saber si programa funciona perfectamente.

## BASES PARA EMPEZAR CON LENGUAJE C

Es importante conocer qué es una librería y su función. Una librería es un conjunto de información que tiene como función realizar una tarea en específico, por lo que existe más de una librería. Cuando se agrega una librería se debe tener la siguiente estructura:

#	nombre de la librería
---	-----------------------

La librería que se usará en esta práctica es **stdio.h**. Esta librería tiene la función de arrojar los datos de escritura y leer. A continuación la estructura que debe contener

Estructura de escribir		Texto	Fin
printf	(" ")	En medio de los corchetes y comillas va el texto que queremos poner	Finalizando con ;

Para leer los es similar el orden, a continuación el orden.

Estructura de leer		Leer	Fin
scanf	("%" ",&)	-A un lado del signo de porcentaje va la letra dependiendo la variable usada. -A un lado del signo aspersan va el nombre de la variable	Finalizando con ;

Es muy usual que al ver pseudocódigos se encuentre texto sin intervenir en el programa y no de la forma printf. Ese tipo de texto son comentarios para indicar al lector que es lo que se está realizando en ese paso, para identificar qué se trata de texto siempre comienza con dos diagonales en caso de ser una comentario corto de una fila, al existir un bloque de texto se usa /\* al inicio, al finalizar es orden contrario \*/.

Ese texto es de mucha importancia pues el lector puede identificar con mayor facilidad el orden del pseudocódigo.

```
//Programa que escriba tu edad
#include <stdio.h> //LIBRERIA
int edad; //DECLARACIÓN DE LA VARIABLE

printf("Ingresa tu edad: "); //IMPRIMIR EN PANTALLA
scanf("%d",&edad); //GUARDA EL NOMBRE DE LA VARIABLE
printf("\nTu edad es %d",edad);
```

## EJERCICIOS EN CLASE

Los siguientes códigos están declarados mal por lo que se deben modificar para poder compilarse de manera correcta.

```
#include <stdio.h>
int main() {
short enteroNumero1 = 115;
signed int enteroNumero2 = 55;
unsigned long enteroNumero3 = 789;
char caracterA = 65;
char caracterB = 'B';
float puntoFlotanteNumero1 = 89.8;
unsigned double puntoFlotanteNumero2 = 238.2236;
return 0;
}
```

```
Aldair:~ aldair$ vi prac1.c
Aldair:~ aldair$ gcc prac1.c -o prac1.out
prac1.c:7:18: error: unexpected character <U+2018>
char caracterB = 'B';
                  ^
prac1.c:7:22: error: character <U+2019> not allowed in an identifier
char caracterB = 'B';
                  ^
prac1.c:7:21: error: use of undeclared identifier 'B'
char caracterB = 'B';
                  ^
prac1.c:9:1: error: 'double' cannot be signed or unsigned
unsigned double puntoFlotanteNumero2 = 238.2236;
^
4 errors generated.
Aldair:~ aldair$
```

Como se puede ver en el momento de compilar la terminal arroja error en la declaración de la letra B, las comillas no son las adecuadas.

```
#include <stdio.h>
int main() {
short enteroNumero1 = 115;
signed int enteroNumero2 = 55;
unsigned long enteroNumero3 = 789;
char caracterA = 65;
char caracterB = 'B';
float puntoFlotanteNumero1 = 89.8;
return 0;
}
```

```
[Aldair:~ aldair$ gcc prac1.c -o prac1.out
[Aldair:~ aldair$ ./prac1
-bash: ./prac1: No such file or directory
Aldair:~ aldair$
```

Se logró compilar, solo que el programa no contiene ningún dato de imprimir en pantalla.

```
#include <stdio.h>
int main() {
//Declaración de variables
int entero;
float flotante;
double doble;
char caracter;
//Asignación de variables
entero = 14;
flotante = 3.5f;
doble = 6.8e10;
caracter = 'A';
//Funciones de salida de datos en pantalla
printf("La variable entera tiene valor: %i \n", entero);
printf("La variable flotante tiene valor: %f \n", flotante);
printf("La variable doble tiene valor: %f \n", doble);
printf("La variable caracter tiene valor: %c \n", caracter);
printf("Entero como octal: %o \n Como Hexadecimal %X \n", entero,
entero);
printf("Flotante con precisión: %5.2f \n", flotante);
printf("Doble con precisión: %5.2f \n", doble);
printf("Carácter como entero: %d \n", caracter);
return 0;
-- INSERT --
```

```
[Aldair:~ aldair$ gcc prac1.c -o prac1.out
[Aldair:~ aldair$ ./prac1.c
-bash: ./prac1.c: Permission denied
```

## SECUENCIAS DE CARÁCTER

<code>\n</code>	Funciona para dar una salta entre línea
<code>\t</code>	Funciona para dar un tabulador

```
C prog1.c > main()
1  #include <stdio.h>
2
3  int entero;
4  float flotante;
5
6  int main () {
7      printf("Ingresa el valor entero: ");
8      scanf("%d",&entero);
9      printf("\n\tEl valor ingresado es %d",entero);
10     printf("\nIngresa el valor float: ");
11     scanf("%f",&flotante);
12     printf("\n\tEl valor ingresado es %f",flotante);
13     return 0;
14 }
```

PROBLEMAS   SALIDA   TERMINAL   CONSOLA DE DEPURACIÓN

- Aldair:aprendiendo en C aldair\$ gcc prog1.c -o prog1.out
- Aldair:aprendiendo en C aldair\$ ./prog1
- Aldair:aprendiendo en C aldair\$ gcc prog1.c -o prog1.out
- Aldair:aprendiendo en C aldair\$ ./prog1

```
Ingresa el valor entero: 30

        El valor ingresado es 30
Ingresa el valor float: 59.245
```

```
C prog1.c > main()
1  #include <stdio.h>
2  int main()
3  {
4      int enteroNumero;
5      char caracterA = 65; // Convierte el entero a carácter ASCII.
6      double puntoFlotanteNumero;
7      // Asignar valor de teclado a una variable.
8      printf("Escriba un valor entero: ");
9      scanf("%i", &enteroNumero);
10     printf("Escriba un valor real: ");
11     scanf("%lf", &puntoFlotanteNumero);
12     // Imprimir valores con formato.
13     printf("\nImprimiendo las variables enteras \a\n");
14     printf("\t Valor de enteroNumero = %i \a\n", enteroNumero);
15     printf("\t Valor de caracterA = %c \a\n", caracterA);
16     printf("\t Valor de puntoFlotanteNumero = %lf \a\n",
17     puntoFlotanteNumero);
18     printf("\t Valor de enteroNumero en base 16 = %x \a\n", enteroNumero);
19     printf("\t Valor de caracterA en código hexadecimal = %x\n", caracterA);
20     printf("\t Valor de puntoFlotanteNumero\n");
21     printf("en notación científica = %e\n", puntoFlotanteNumero);
22     return 0;
23 }
```

- Aldair:aprendiendo en C aldair\$ gcc prog1.c -o prog1
- Aldair:aprendiendo en C aldair\$ ./prog1

```
Escriba un valor entero: 5
Escriba un valor real: -3

Imprimiendo las variables enteras
Valor de enteroNumero = 5
Valor de caracterA = A
Valor de puntoFlotanteNumero = -3.000000
Valor de enteroNumero en base 16 = 5
Valor de caracterA en código hexadecimal = 41
Valor de puntoFlotanteNumero
en notación científica = -3.000000e+00
```

```

C prog2.c > main()
1  #include <stdio.h>
2  int main()
3  {
4      short ocho, cinco, cuatro, tres, dos, uno;
5
6      // 8 en binario: 0000 0000 0000 1000
7      ocho = 8;
8      // 5 en binario: 0000 0000 0000 0101
9      cinco = 5;
10     // 4 en binario: 0000 0000 0000 0100
11     cuatro = 4;
12     // 3 en binario: 0000 0000 0000 0011
13     tres = 3;
14     // 2 en binario: 0000 0000 0000 0010
15     dos = 2;
16     // 1 en binario: 0000 0000 0000 0001
17     uno = 1;
18     printf("Operadores aritméticos\n");
19     printf("5 modulo 2 = %d\n", cinco%dos);
20     printf("Operadores lógicos\n");
21     printf("8 >> 2 = %d\n", ocho>>dos);
22     printf("8 << 1 = %d\n", ocho<<1);
23     printf("5 & 4 = %d\n", cinco&cuatro);
24     printf("3 | 2 = %d\n", tres|dos);
25
26     printf("\n");
27     return 0;
28 }

```

- Aldair:aprendiendo en C aldair\$ gcc prog2.c -o prog2.c
- Aldair:aprendiendo en C aldair\$ ./prog2.c

```

Operadores aritméticos
5 modulo 2 = 1
Operadores lógicos
8 >> 2 = 2
8 << 1 = 16
5 & 4 = 4
3 | 2 = 3

```

## EJERCICIOS DE TAREA

1. Investigar cual es el dato que se encuentra por default en el lenguaje C (signed o unsigned)

Signed es el dato que aparece por default ya que en primer instante se permite usar valores positivos

2. Indicar que sucede cuando en una variable tipo caracter se emplea el formato %d, %i, %o, %x

Estos son caracteres de conversión, antes se debe tomar en cuenta el tipo de variable que se va a declarar, en el caso de d se refiere a un valor entero, Cuando es i se habla de entero decimal, octal o hexadecimal. Cuando es o se refiere a entero octal. Por último, cuando es x es un entero hexadecimal



### 3. Mencionar las características con las que debe crearse una variable

Considerar el funcionamiento que va existir a lo largo del programa, considerando el resultado final.

### 4. ¿Cuál es la diferencia entre variable estática y constante?

Una variable estática es aquella que no cambia su valor a lo largo del programa, se declara el nombre junto con el valor, por otro lado, una variable constante es aquella que puede cambiar su valor a lo largo del programa

### 5. Menciona en qué momento empleamos los dos tipos de diferentes ( < > != )

Estas son expresiones lógicas que son utilizadas mayormente en expresiones de condición.

< significa que un valor es menor a otro.

> significa que un valor es mayor a otro.

!= significa que un valor es diferente a otro.

### 6. Crea un programa en el que declares 4 variables haciendo uso de las reglas signed/unsigned, las cuatro variables deben ser solicitadas al usuario(se emplea scanf) y deben mostrarse en pantalla (emplear printf)

```
C prog2.c > main()
1 //EJERCICIO 4
2 #include <stdio.h>
3 int signed var1, var2;
4 unsigned var3, var4;
5
6 int main ()
7 {
8     printf("Ingresa el valor de tu primer variable: ");
9     scanf("%d",&var1);
10    printf("\nIngresa el valor de tu segunda variable: ");
11    scanf("%d",&var2);
12    printf("\nIngresa el valor de tu tercer variable: ");
13    scanf("%d",&var3);
14    printf("\nIngresa el valor de tu cuarta variable: ");
15    scanf("%d",&var4);
16    printf("\n\tEl valor de tu primer variable es: %d",var1);
17    printf("\n\tEl valor de tu segunda variable es: %d",var2);
18    printf("\n\tEl valor de tu tercer variable es: %d",var3);
19    printf("\n\tEl valor de tu cuarta variable es: %d",var4);
20    printf("\n");
21 }
```

```
● Aldair:aprendiendo en C aldair$ gcc prog2.c -o prog2.c
```

```
● Aldair:aprendiendo en C aldair$ ./prog2.c
```

```
Ingresa el valor de tu primer variable: 134
```

```
Ingresa el valor de tu segunda variable: 143
```

```
Ingresa el valor de tu tercer variable: -345
```

```
Ingresa el valor de tu cuarta variable: -678
```

```
El valor de tu primer variable es: 134
```

```
El valor de tu segunda variable es: 143
```

```
El valor de tu tercer variable es: -345
```

```
El valor de tu cuarta variable es: -678
```

7. Crea un programa que le solicite su edad al usuario, leer el datos( emplear scanf) y mostrarlo en pantalla.

```

C prog2.c > main()
1  //EJERCICIO 7
2  #include <stdio.h>
3  int edad;
4
5  int main ()
6  {
7      printf("Ingresa tu edad: ");
8      scanf("%d",&edad);
9      printf("\n\tTu edad es %d",edad);
10     printf("\n");
11 }

```

PROBLEMAS   SALIDA   TERMINAL   CONSOLA DE DEPURACIÓN

```

● Aldair:aprendiendo en C aldair$ gcc prog2.c -o prog2.c
● Aldair:aprendiendo en C aldair$ ./prog2.c
Ingresa tu edad: 21

Tu edad es 21

```

8. Revisar y colocar cuando se emplea MOD y cuando se emplea % (pseudocódigo y codificación) agrega un ejemplo de su uso.

```

C prog2.c > main()
1  //EJERCICIO 8
2  #include <stdio.h>
3  int num, resultado;
4
5  int main ()
6  {
7      printf("Ingresa el valor de un número y te diré si es divisible entre 2: ");
8      scanf("%d",&num);
9      if (num%2==0)
10         printf("\nEl valor es divisible entre 2");
11         printf("\n");
12 }

```

PROBLEMAS   SALIDA   TERMINAL   CONSOLA DE DEPURACIÓN

```

● Aldair:aprendiendo en C aldair$ gcc prog2.c -o prog2.c
● Aldair:aprendiendo en C aldair$ ./prog2.c
Ingresa el valor de un número y te diré si es divisible entre 2: 20

El valor es divisible entre 2

```

La función MOD, en lenguaje C se usa el signo % para conocer el residuo de una división.

9. Comparación entre Editor de Texto y Procesador de Texto(Realizar una tabla comparativa)

EDITOR DE TEXTO	PROCESADOR DE TEXTO
-----------------	---------------------

Solamente permite texto Lee el archivo e interpreta los caracteres	Permite cambiar su aspecto, modificar su aspecto y no solo el texto. Contiene formato
---	--

### 10. Indica los comandos utilizados para compilar y para ejecutar un programa en iOS o Linux .ls

En este caso como se está usando la TERMINAL al finalizar el programa se debe salir con ayuda de **:wq**. Posteriormente se usan los comandos **gcc nombre del programa.c -o nombre del programa. out**. Finalmente se pone **./nombre del programa.c**

### 11. Compilación y prueba del programa antes mostrado en DEV C++ u otro IDE en sistema operativo Windows.

```

C prog2.c > main()
1 //EJERCICIO 8
2 #include <stdio.h>
3 int num, resultado;
4
5 int main ()
6 {
7     printf("Ingresa el valor de un número y te diré si es divisible entre 2: ");
8     scanf("%d",&num);
9     if (num%2==0)
10        printf("\nEl valor es divisible entre 2");
11        printf("\n");
12 }

```

PROBLEMAS SALIDA TERMINAL CONSOLA DE DEPURACIÓN

```

● Aldair:aprendiendo en C aldair$ gcc prog2.c -o prog2.c
● Aldair:aprendiendo en C aldair$ ./prog2.c
Ingresa el valor de un número y te diré si es divisible entre 2: 20
El valor es divisible entre 2

```

La compilación está en la parte de abajo donde dice terminal.

### 12. Genera un programa que solicite dos variables enteras al usuario y realice las 4 operaciones básicas, compila y ejecuta el programa utilizando terminal y los comandos indicados para cada instrucción.

```

C prog12.c > main()
1 //PROGRAMA QUE AL RECIBIR DOS NÚMEROS HAGA LAS 4 OPERACIONES BÁSICAS
2 #include <stdio.h>
3 int var1, var2;
4 int suma, resta, multiplicacion, division;
5 int main ()
6 {
7     printf("Dame el primer valor");
8     scanf("%d",&var1);
9     printf("\nDame el segundo valor");
10    scanf("%d",&var2);
11    suma=(var1+var2);
12    resta=(var1-var2);
13    multiplicacion=(var1)*(var2);
14    division=(var1/var2);
15    printf("\n\tLa suma de los numeros es %d",suma);
16    printf("\n\tLa resta de los numeros es %d",resta);
17    printf("\n\tLa multiplicacion de los numeros es %d",multiplicacion);
18    printf("\n\tLa division de los numeros es %d",division);
19    printf("\n");
20 }

```

PROBLEMAS   SALIDA   TERMINAL   CONSOLA DE DEPURACIÓN

- Aldair:aprendiendo en C aldair\$ gcc prog12.c -o prog12.c
- Aldair:aprendiendo en C aldair\$ ./prog12.c

```

Dame el primer valor20
Dame el segundo valor30

La suma de los numeros es 50
La resta de los numeros es -10
La multiplicacion de los numeros es 600
La division de los numeros es 0

```

## CONCLUSIÓN

Es importante mencionar que la ventaja del lenguaje C es que es fuertemente tipado lo que ocasiona que los estudiantes sepamos modificar gran parte de las cosas.

Es importante mencionar que el lenguaje C puede servir como base para el entendimiento de otros lenguajes en el futuro.