	Carátula para entrega de prácticas	
	Facultad de Ingeniería	Laboratorio de docencia

Laboratorio de computación salas A y B

<i>Profesor:</i>	Alejandro Esteban Pimentel Alarcón
<i>Asignatura:</i>	Fundamentos de programación
<i>Grupo:</i>	3
<i>No de Práctica(s):</i>	7
<i>Integrante(s):</i>	Gabriela Sabrina Orea Torres
<i>No. de Equipo de cómputo empleado:</i>	
<i>No. de Lista o Brigada:</i>	#0686
<i>Semestre:</i>	2020-1
<i>Fecha de entrega:</i>	6 octubre 2019
<i>Observaciones:</i>	Tarde entrega. La práctica esta incompleta, no muestras evidencias (capturas) de haber compilado y ejecutado los programas correctamente. Además te faltan conclusiones.

CALIFICACIÓN: 6

PRÁCTICA 7. Fundamentos de Lenguaje C.

Objetivo: Elaborar programas en lenguaje C utilizando las instrucciones de control de tipo 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 variables y expresiones.

Introducción

Después de haber identificado los datos de entrada y salida de un problema, de haber diseñado un algoritmo para su solución y su diagrama de flujo o pseudocódigo correspondiente, se procede a la etapa de codificación que se puede realizar con algún lenguaje de programación, en este caso el lenguaje C.

Todo programa C consiste en un conjunto de funciones, una de ellas llamada `main()` la cual es la principal y es la primera que se ejecuta al comenzar el programa. Para la creación de un programa C se deben seguir tres etapas: edición en la que se escribe el código fuente desde un editor de textos, compilación en la que se genera el archivo en lenguaje máquina, y por último la ejecución.

Tipos de variables.

Antes de poder usar cualquier variable, ésta debe ser declarada.
Los tipos de datos básicos en C son los siguientes:

- Caracteres: codificación definida por la máquina
- Enteros: números sin punto decimal
- Flotantes: número reales
- Dobles: números reales de doble precisión

Las variables enteras que existen el lenguaje C son las siguientes:

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	$-(2^{63})$ to $(2^{63})-1$
unsigned long long int	8	0 to 18,446,744,073,709,551,615

Y las variables reales, que siempre poseen signo, son:

<i>Tipo</i>	<i>Bits</i>	<i>Valor Mínimo</i>	<i>Valor Máximo</i>
<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

Para mostrar y leer el valor de una variable se debe especificar el tipo de dato y los especificadores son los siguientes:

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

En el siguiente ejemplo se muestra la sintaxis utilizada para declarar variables, asignarles valores, mostrar y leerlos:

```

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

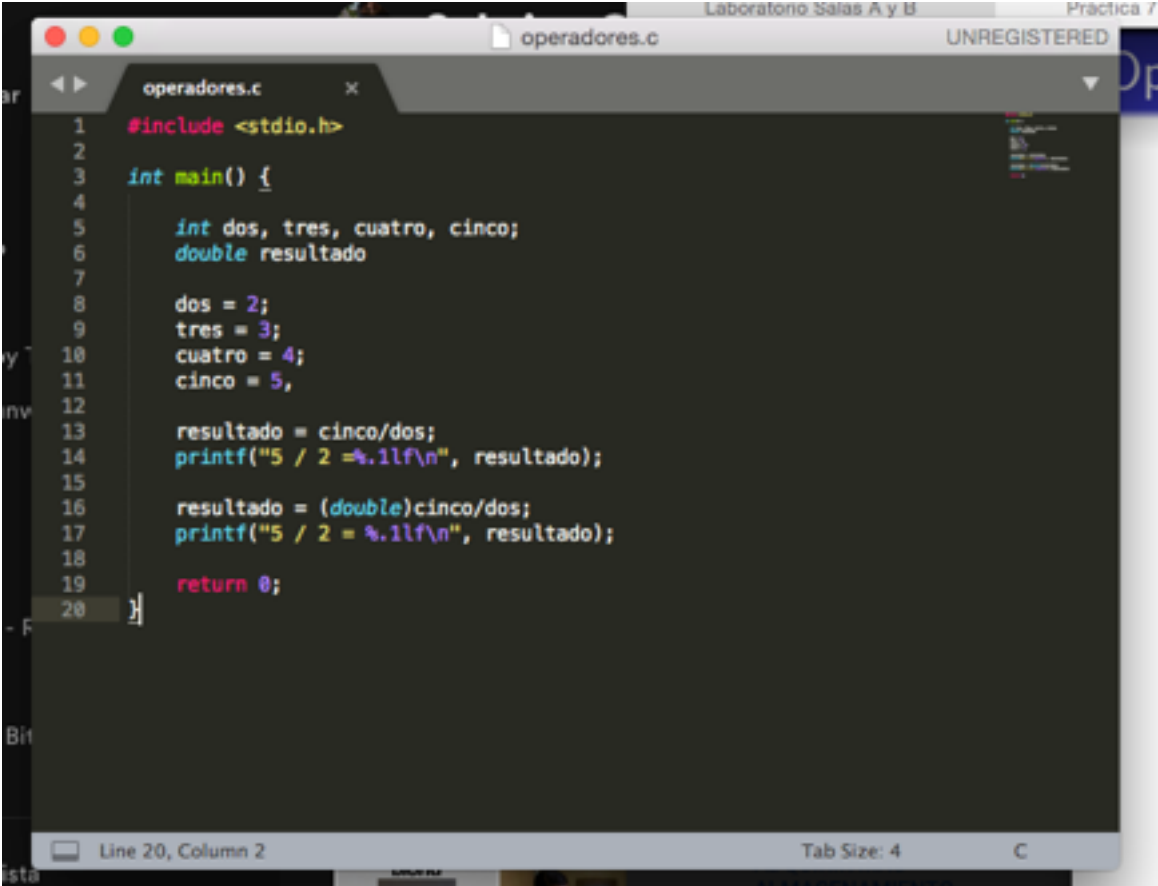
```

printf es una función para imprimir con formato.
scanf es una función para leer datos de entrada.

Los **operadores matemáticos** que se usan en lenguaje C son los siguientes:

<i>Operador</i>	<i>Operación</i>	<i>Uso</i>	<i>Resultado</i>
+	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

Y a continuación un ejemplo de su utilización en el editor de texto:



```
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 }
```

The screenshot shows a code editor window titled 'operadores.c' with a dark background. The code is written in C and demonstrates division using the `printf` function. The program declares variables `dos`, `tres`, `cuatro`, and `cinco` as integers, and `resultado` as a double. It assigns values to these variables and then performs division using both integer and floating-point division, printing the results with one decimal place. The status bar at the bottom indicates 'Line 20, Column 2' and 'Tab Size: 4'.

Los **operadores de relación** nos permiten comparar constantes y variables, números o letras:

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

Y los **operadores lógicos** permiten formular condiciones complejas.

Operador	Operación	Uso
!	No	! p
&&	Y	a > 0 && a < 11
	O	opc == 1 salir != 0

```

1 #include <stdio.h>
2
3 int main () {
4     int num1, num2, res;
5     char c1, c2;
6
7     num1 = 7;
8     num2 = 15;
9     c1 = 'h';
10    c2 = 'H';
11
12    printf("¿ num1 es menor a num2 ? -> %d\n", num1 < num2);
13    printf("¿ c1 es igual a c2 ? -> %d\n", c1 == c2);
14    printf("¿ c1 es diferente a c2 ? -> %d\n", c1 != c2);
15
16    res = num1 < num2 && c1 == 'h';
17    printf("¿ num1 < num2 Y c1 es igual a 'h' ? -> %d\n", res);
18
19    res = c1 == 's' || c2 == 'H' ;
20    printf("¿ c1 es igual a 's' O c2 a 'H' ? -> %d\n", res);
21
22    return 0;
23 }
  
```