	Carátula para entrega de prácticas	
Facultad de Ingeniería	Laboratorio de docencia	

Laboratorios de computación

Salas A y B

<i>Profesor:</i>	Alejandro Esteban Pimentel Alarcon
<i>Asignatura:</i>	Fundamentos de Programación
<i>Grupo:</i>	135
<i>No de Práctica(s):</i>	7
<i>Integrante(s):</i>	Godínez Juárez Alondra Itati
<i>No. de Equipo de cómputo empleado:</i>	50
<i>No. de Lista o Brigada:</i>	316146153
<i>Semestre:</i>	2020-1
<i>Fecha de entrega:</i>	
<i>Observaciones:</i>	Tarde entrega

CALIFICACIÓN: 8

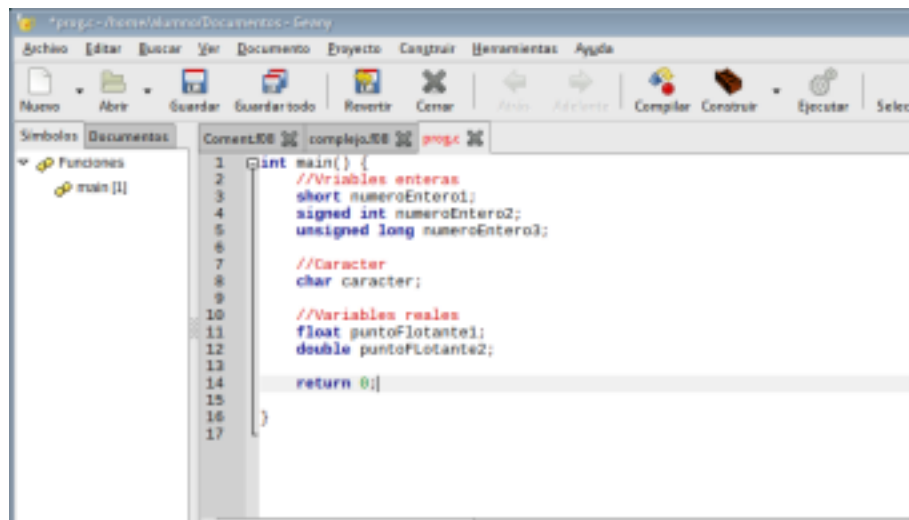
Fundamentos de Lenguaje C

Introducción. En la presente práctica nos adentraremos y definiremos aún más en la estructura de lenguaje c con tipos de variables, mostrar un mensaje, operadores lógicos y comparaciones.

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.

Tipos de variables.

Tipo	Tamaño en bits	Rango
char	8	-127 a 127
unsigned char	8	0 a 255
signed char	8	-127 a 127
int	16	-32767 a 32767
unsigned int	16	0 a 65535
signed int	16	-32767 a 32767
short int	16	-32767 a 32767
unsigned short int	16	0 a 65535
signed short int	16	-32767 a 32767
long int	32	-2147483647 a 2147483647
signed long int	32	-2147483647 a 2147483647
unsigned long int	32	0 a 4294967295
float	32	seis dígitos de precisión
double	64	diez dígitos de precisión
long double	64	diez dígitos de precisión



```
1 int main() {
2     //Variables enteras
3     short numeroEntero1;
4     signed int numeroEntero2;
5     unsigned long numeroEntero3;
6
7     //Caracter
8     char caracter;
9
10    //Variables reales
11    float puntoFlotante1;
12    double puntoFlotante2;
13
14    return 0;
15 }
16
17
```

Mostrar y Leer.

Lectura y Escritura de Datos. Para poder leer y escribir datos en el lenguaje de programación C existen una serie de funciones agrupadas en un conjunto de librerías de código objeto, que constituyen la llamada biblioteca estándar del lenguaje. En el caso concreto de las funciones de entrada y salida (lectura y escritura), su archivo de cabecera es *stdio.h*.

Existen varias funciones que realizan la entrada y salida de datos en el lenguaje de programación C, pero nos vamos a centrar solamente en dos de ellas: *printf()* y *scanf()*.

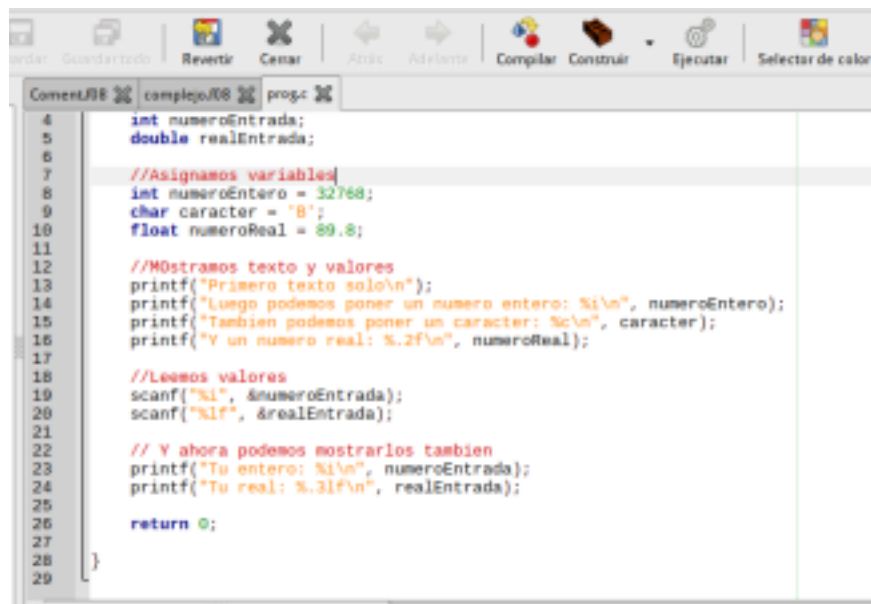
Función *printf()*. La función *printf()* sirve para escribir datos en la pantalla con un formato determinado. El prototipo de esta función es la siguiente:

```
int printf(const char *formato, arg1, arg2, ..., argn)
```

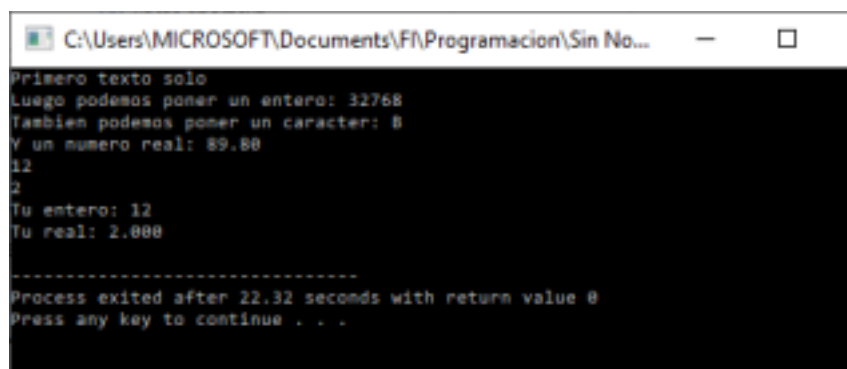
Función *scanf()*. La función *scanf()* se puede considerar de alguna manera como la inversa de la función *printf()*, pues sirve para introducir datos desde el teclado con un formato determinado. El prototipo de esta función es el siguiente:

```
int scanf(const char *formato, arg1, arg2, ..., argn);
```

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  //Comentario
2
3  #include <stdio.h>
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 numero entero: %i\n", numeroEntero);
15 printf("Tambien podemos poner un caracter: %c\n", caracter);
16 printf("Y un numero real: %.2f\n", numeroReal);
17
18 //Leemos valores
19 scanf("%i", &numeroEntrada);
20 scanf("%lf", &realEntrada);
21
22 // Y ahora podemos mostrarlos tambien
23 printf("Tu entero: %i\n", numeroEntrada);
24 printf("Tu real: %.3lf\n", realEntrada);
25
26 return 0;
27
28 }
29
```



```
C:\Users\MICROSOFT\Documents\F\Programacion\Sin No...
Primero texto solo
Luego podemos poner un entero: 32768
Tambien podemos poner un caracter: B
Y un numero real: 89.80
12
2
Tu entero: 12
Tu real: 2.000

-----
Process exited after 22.32 seconds with return value 0
Press any key to continue . . .
```

Operadores.

Nivel	Operadores	Descripción	Asoci.
1	() [] -> .	Acceso a un elemento de un vector y paréntesis	Izquierdas
2	+ - ! ~ * & ++ -- (cast) sizeof	Signo (unario), negación lógica, negación bit a bit Acceso a un elemento (unarios): puntero y dirección Incremento y decremento (pre y post) Conversión de tipo (casting) y tamaño de un elemento	Derechas
3	* / %	Producto, división, módulo (resto)	Izquierdas
4	+ -	Suma y resta	Izquierdas
5	>> <<	Desplazamientos	Izquierdas
6	< <= >= >	Comparaciones de superioridad e inferioridad	Izquierdas
7	== !=	Comparaciones de igualdad	Izquierdas
8	&	Y (And) bit a bit (binario)	Izquierdas
9	^	O-exclusivo (Exclusive-Or) (binario)	Izquierdas
10		O (Or) bit a bit (binario)	Izquierdas
11	&&	Y (And) lógico	Izquierdas
12		O (Or) lógico	Izquierdas
13	?:	Condicional	Derechas
14	= *= /= %+= += -= >>= <<= &= ^= =	Asignaciones	Derechas
15	,	Coma	Izquierdas

```

1 #include <stdio.h>
2 int main() {
3     //Declaramos variables a leer
4     int dos, tres, cuatro, cinco;
5     double resultado;
6
7     dos = 2;
8     tres = 3;
9     cuatro = 4;
10    cinco = 5;
11
12    resultado = cinco/dos;
13    printf("5 / 2 = %.11f\n", resultado);
14
15    resultado = (double)cinco/dos;
16    printf("5 / 2 = %.11f\n", resultado);
17
18    return 0;
19 }
20
21

```

```

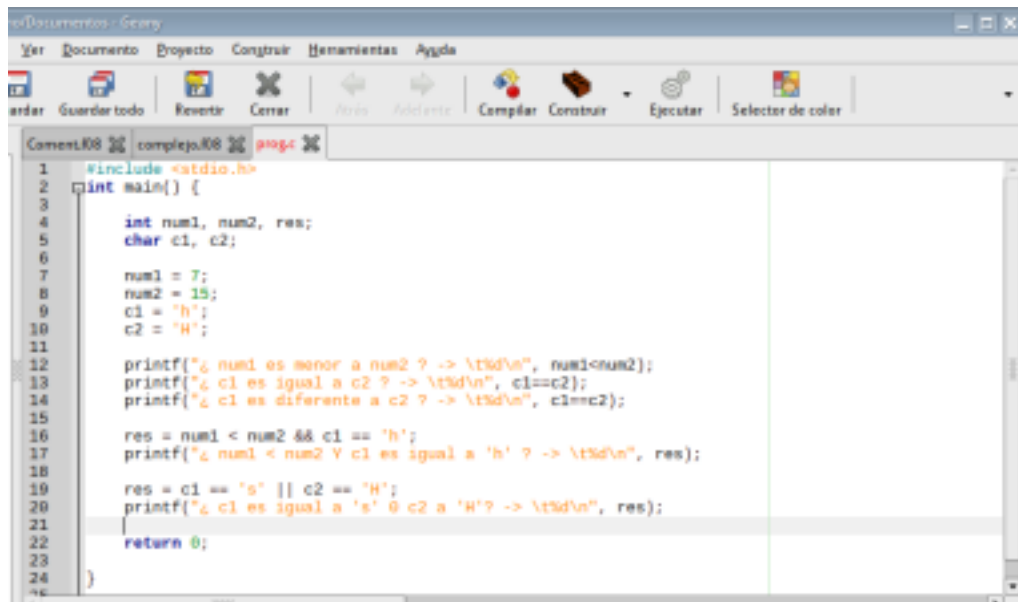
C:\Users\MICROSOFT\Documents\FI\Programacion\Sin No...
5 / 2 = 2.0
5 / 2 = 2.5

-----
Process exited after 2.922 seconds with return value 0
Press any key to continue . . .

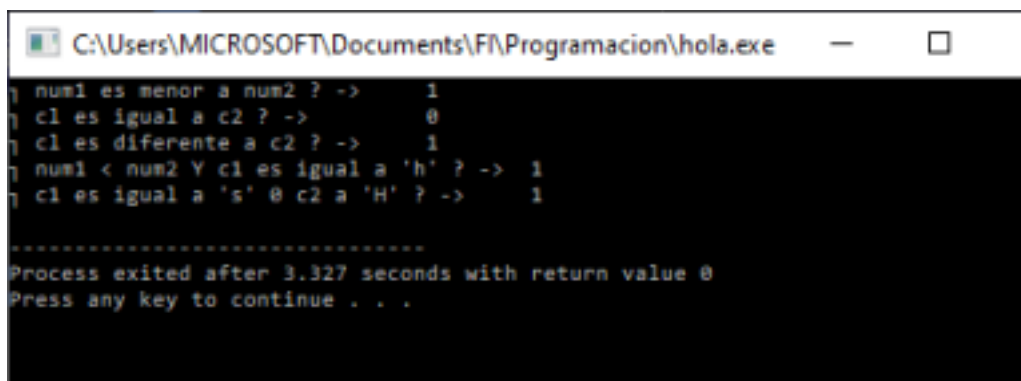
```

Operadores lógicos.

Expresión	Pregunta equivalente	Resultado	Ejemplo código
<code>(A == B) && (A < B)</code>	¿Es A igual a B y A menor que C?	0 (falso)	<code>printf ("Pregunta (A == B) && (A < B) vale %d\n", (A == B) && (A < B));</code>
<code>(A == 5) (A > 7)</code>	¿Es A igual a 5 ó es A mayor que 7?	1 (verdadero)	<code>printf ("Pregunta (A == 5) (A > 7) vale %d\n", (A == 5) (A > 7));</code>
<code>!(A == 5)</code>	¿A es NO igual a 5?	0 (falso)	<code>printf ("Pregunta !(A == 5) vale %d\n", !(A == 5));</code>



```
1 #include <stdio.h>
2 int main() {
3
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 }
24
```



```
C:\Users\MICROSOFT\Documents\FI\Programacion\hola.exe
num1 es menor a num2 ? -> 1
c1 es igual a c2 ? -> 0
c1 es diferente a c2 ? -> 1
num1 < num2 Y c1 es igual a 'h' ? -> 1
c1 es igual a 's' O c2 a 'H' ? -> 1

.....
Process exited after 3.327 seconds with return value 0
Press any key to continue . . .
```

Conclusión. Para terminar, mostramos y usamos todas las herramientas de lenguaje c que se nos propusieron, y esto es una buena guía para hacer en un futuro nuestros propios programas con estas funciones