Nombre de la práctica	**LENGUAJE C**			**No.**	**1**
Asignatura:	**MÉTODOS NÚMERICOS**	**Carrera:**	**ISIC**	**Duración de la práctica (Hrs)**	

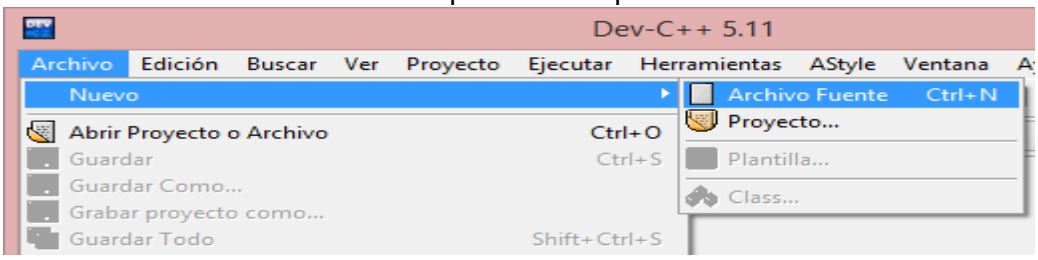
I. Competencia(s) específica(s):

II. Lugar de realización de la práctica (laboratorio, taller, aula u otro):
Casa

III. Material empleado:
Dev c++
Lenguaje_C.pdf
Variables_C.pdf
Operador_incorrecto.pdf
Precedencia_de_operaciones.pdf
Entrada y Salida de datos.pdf

IV. Desarrollo de la práctica:

Para crear un proyecto, nos dirijimos a la parte de archivo-nuevo-archivo fuente. Al dar clic nos dara una pantalla para desarrollar nuestro codigo

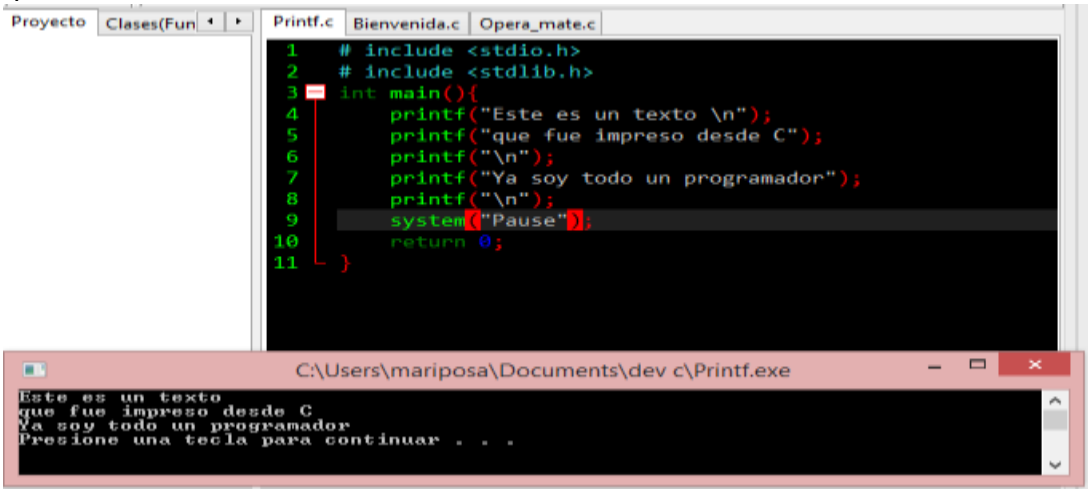


En esta parte se muestra la platilla para el codigo, donde los proyectos se deben de guardar con punto c, para poder ejecutar y compilar en caso de errores poder corregir para que realice su funcion de manera correcta.

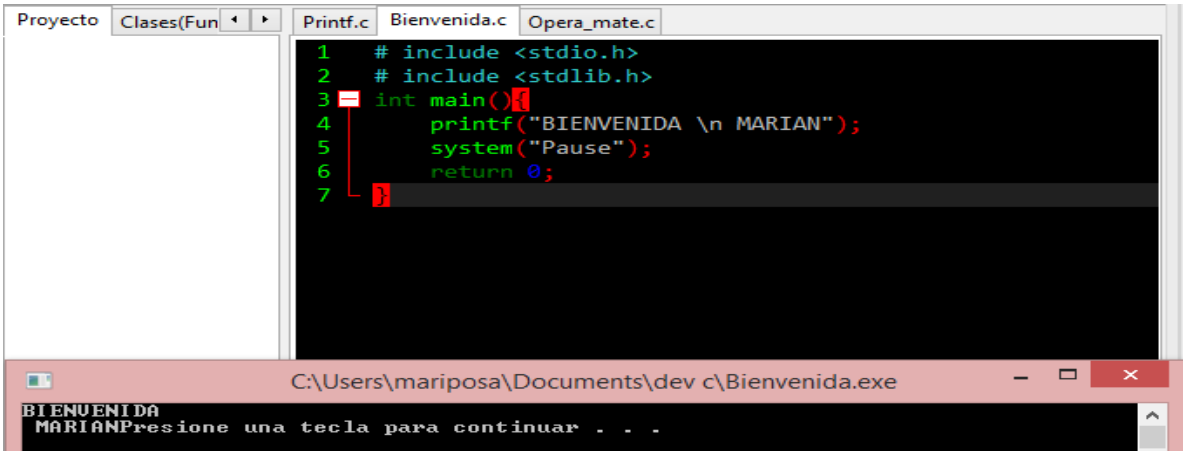
Se incluye los estándares de entrada y salida, las bibliotecas, el método principal donde estará nuestro código de forma estructurada.

Mostraremos una secuencia donde se imprimirá un mensaje en la que (\n) nos funcionara como un salto de línea esto quiere decir que el otro mensaje se pasara al siguiente renglón. Para cerrar el programa se pone un system con un pause, terminando con 0.

Entramos a símbolo de sistema, compilamos y ejecutamos, nos muestra en pantalla lo que tenemos en nuestro código.

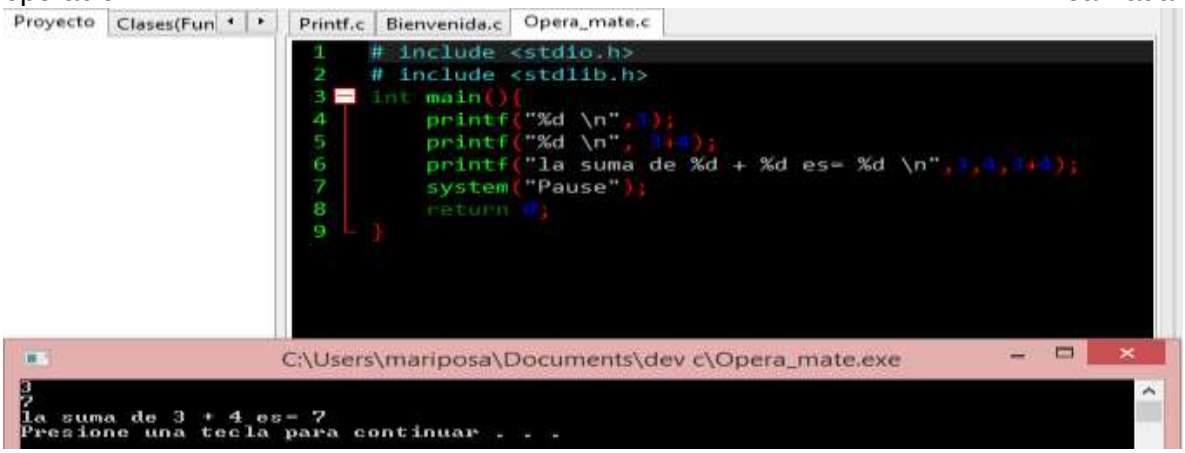


En el siguiente código nos pide que por medio de mensajes mostremos una bienvenida con nuestro nombre. En el cual de la misma manera los pasos se repiten, ponemos los estándares de entrada y salida, la clase principal, las secuencias, con los mensajes utilizando el printf en medio un salto de línea

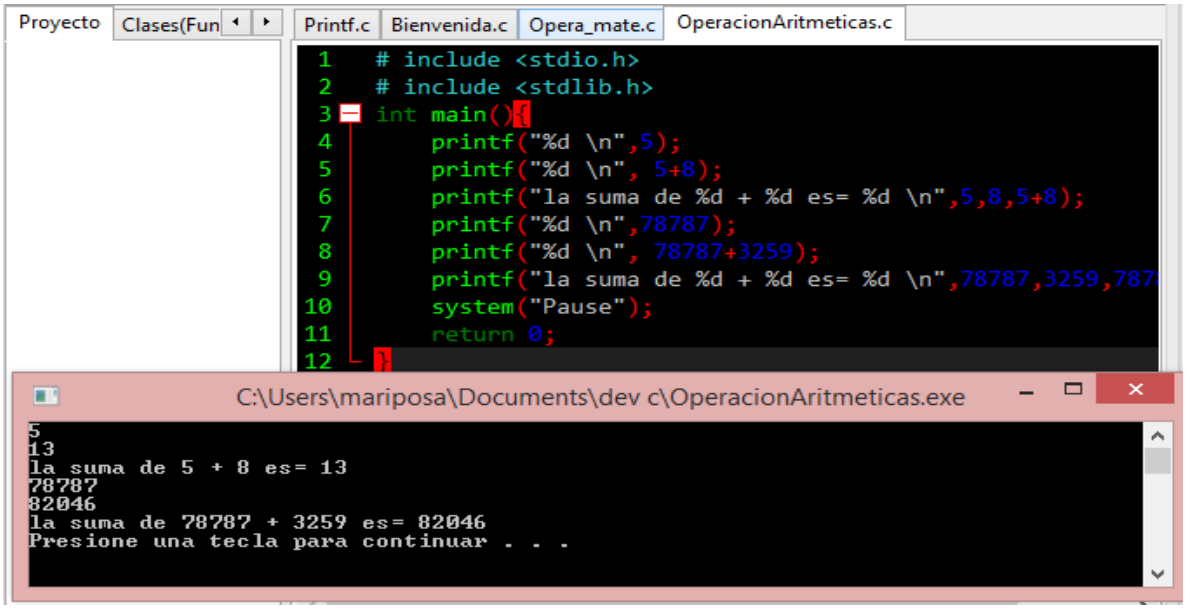


Los estándares se ingresan, la clase principal, el printf nos sirve para incluir variables de valores enteros porque “%d” es para enteros, donde se realiza una suma de dos números, en un mensaje se realiza la suma, las variables se guardan en %d así que cada valor debe tener su respectivo espacio en memoria.

Lo compilamos y ejecutamos mostrando nuestro código, así como la suma de la operación realizada.



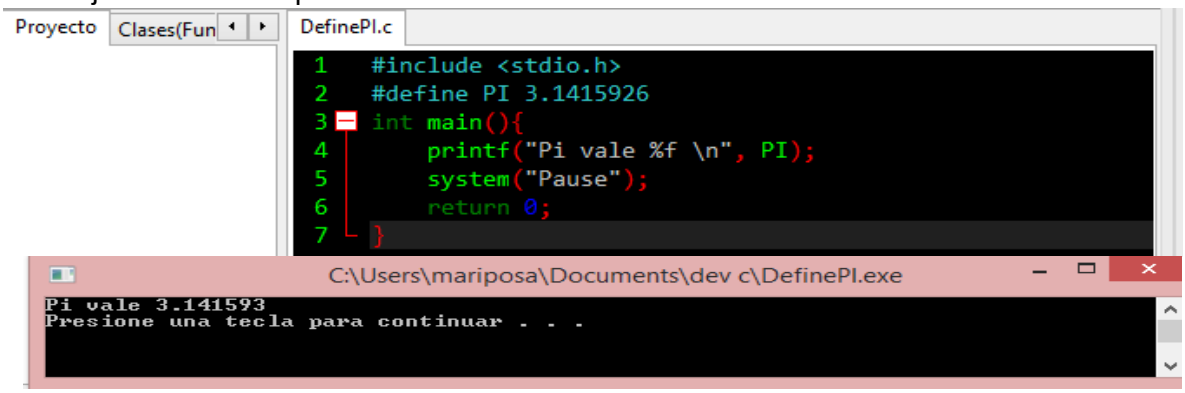
En el siguiente código de igual manera, se realiza una suma tomando en cuenta lo anterior, los estándares, secuencia principal, espacio en memoria de los números para respectivamente hacer la operación. La compilación, ejecución del programa mostrando la suma de los números.



VARIABLES C

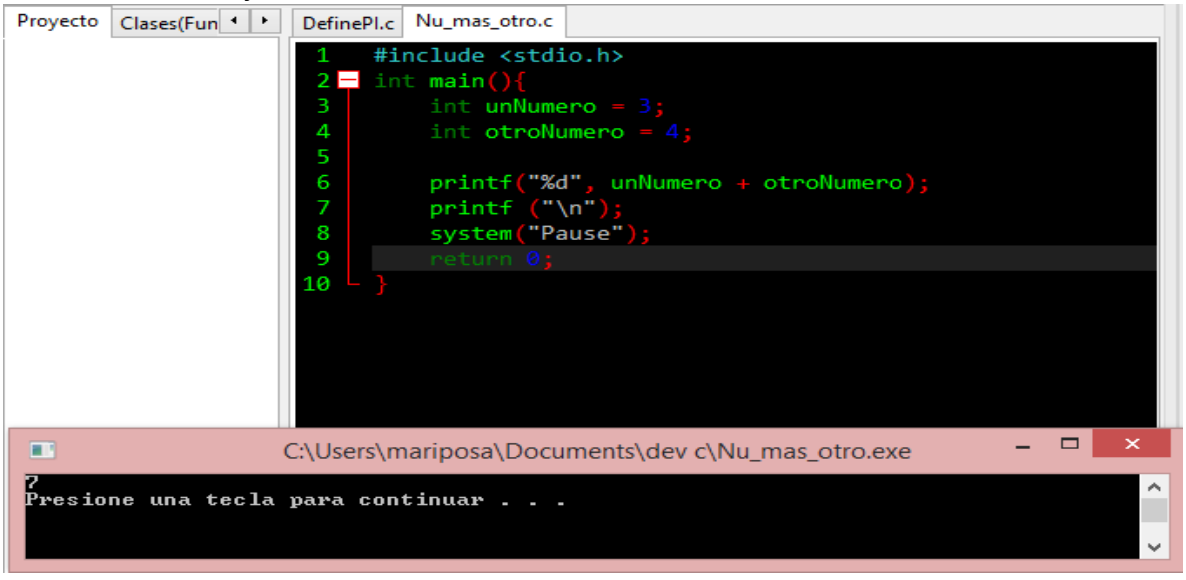
Los estándares se incluyen, se define una variable con valor estático, se imprime un mensaje con el valor ya declarado, terminando la secuencia con una pause, la salida del mensaje.

Compilación, ejecución del programa donde nos damos cuenta que se muestra el mensaje que nos da el valor



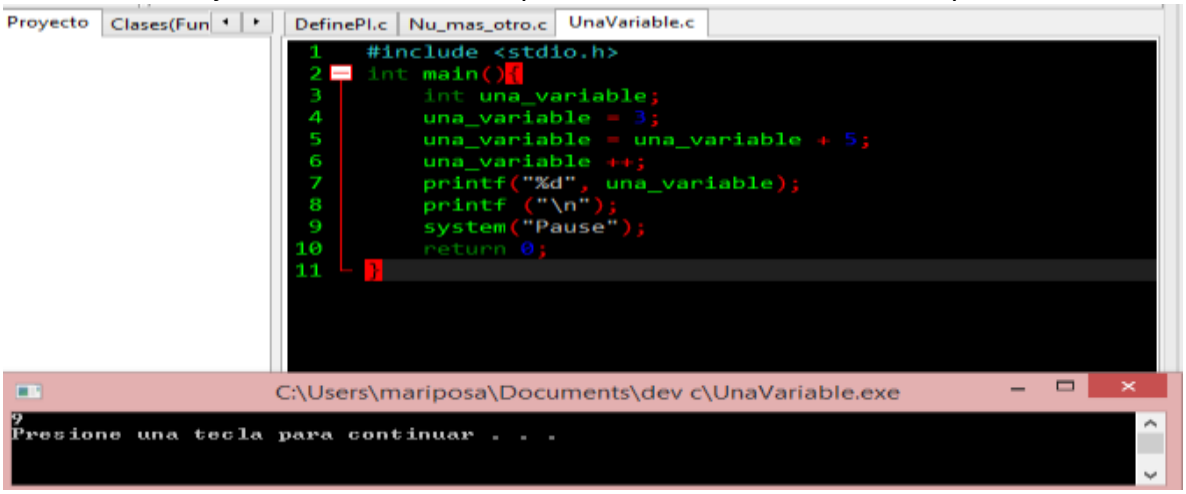
Los estándares se declaran, ahora declaramos variables con su respectivo valor, en el mensaje se realiza la suma guardando el resultado en espacio en memoria, un salto de página así mostrara la parte de la pause debajo del resultado.

En la ejecución muestra el valor de la suma.



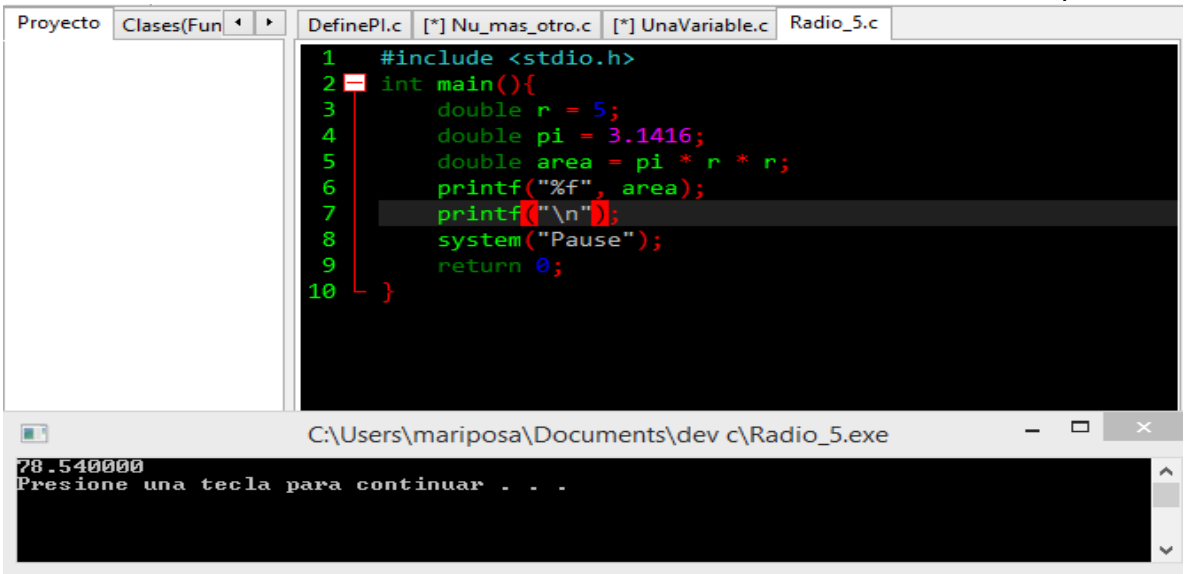
Los estándares se agregan, clase principal, se declara un valor de tipo entero, la variable se ocupa de nuevo asignado un número 3, la variable vale 3 se le suma 5, así variable vale 8 y como se incrementa eso quiere decir que se le suma. Se imprime el mensaje con el resultado de la variable.

En la ejecución se imprime el resultado que es 9.



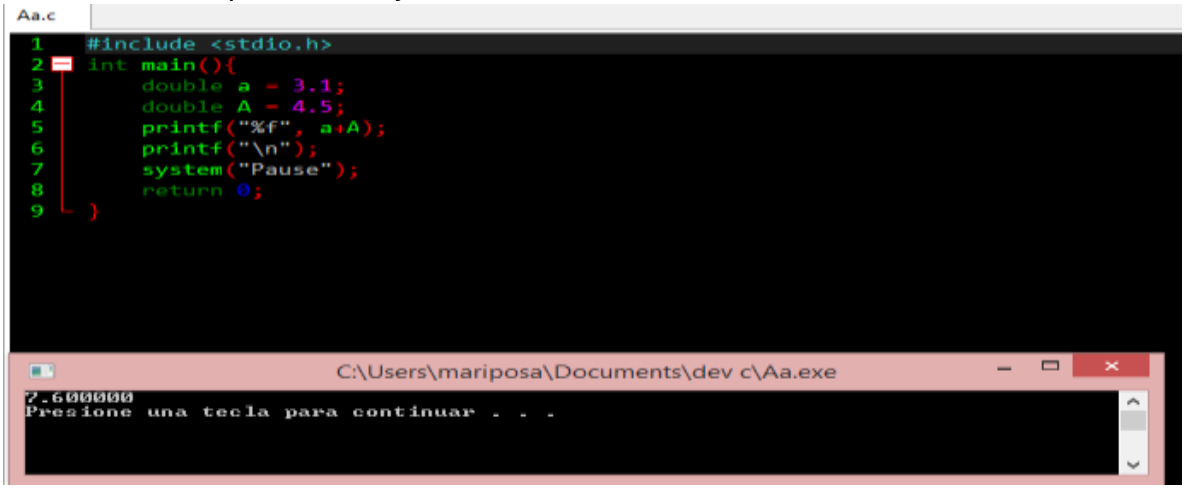
El punto de este código, es obtener el área de un circulo, determinando así el valor del radio como el valor de PI, las variables se declaran como double por los puntos decimales, existe una variable con la multiplicación para obtener el valor.

Se imprime con un espacio en memoria de la variable de resultado, en la ejecución se muestra el resultado de la operación.

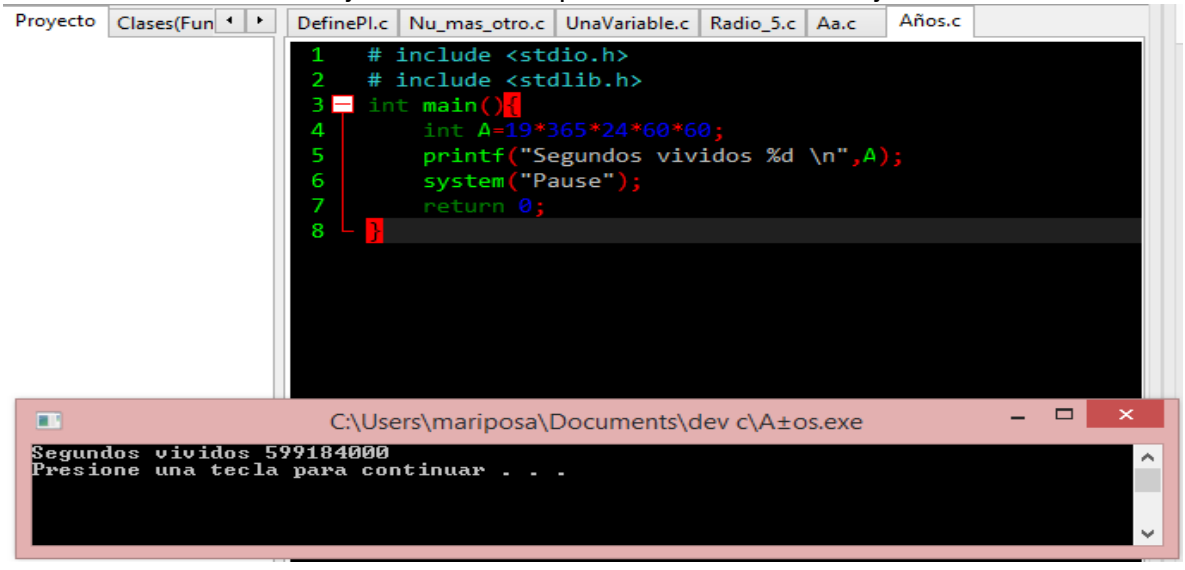


Los estándares, con la clase principal, donde se declaran variables de tipo double con su valor ya definido, lo que se realiza es una suma de estas dos variables, donde en la impresión se reserva un espacio en memoria para el resultado.

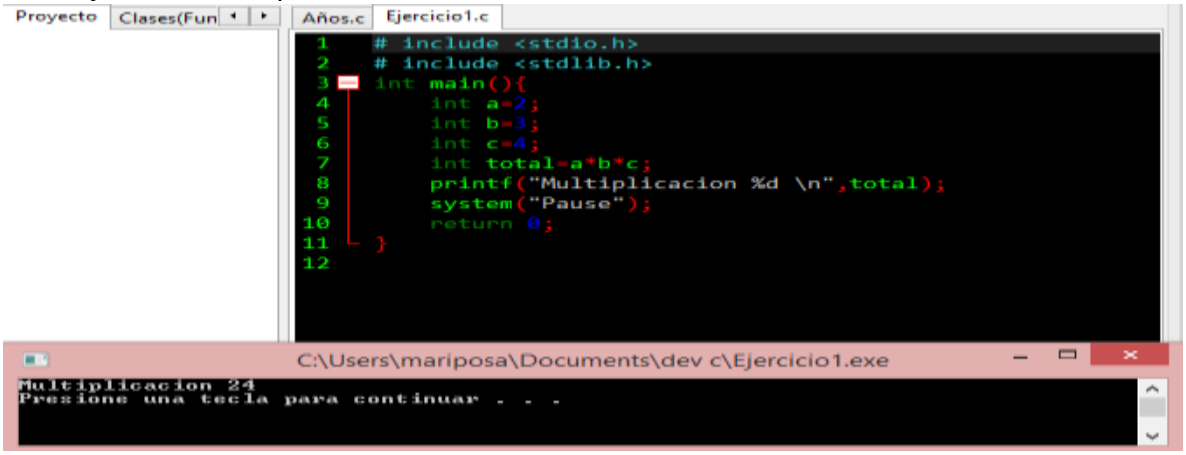
En la compilación nos dirá si existe un error de ser así no se podrá ejecutar y nos dirá donde está el error por lo que debemos de corregirlo para que este sea ejecutado. Nos damos cuenta que en la ejecución nos muestra la suma de estas dos variables.



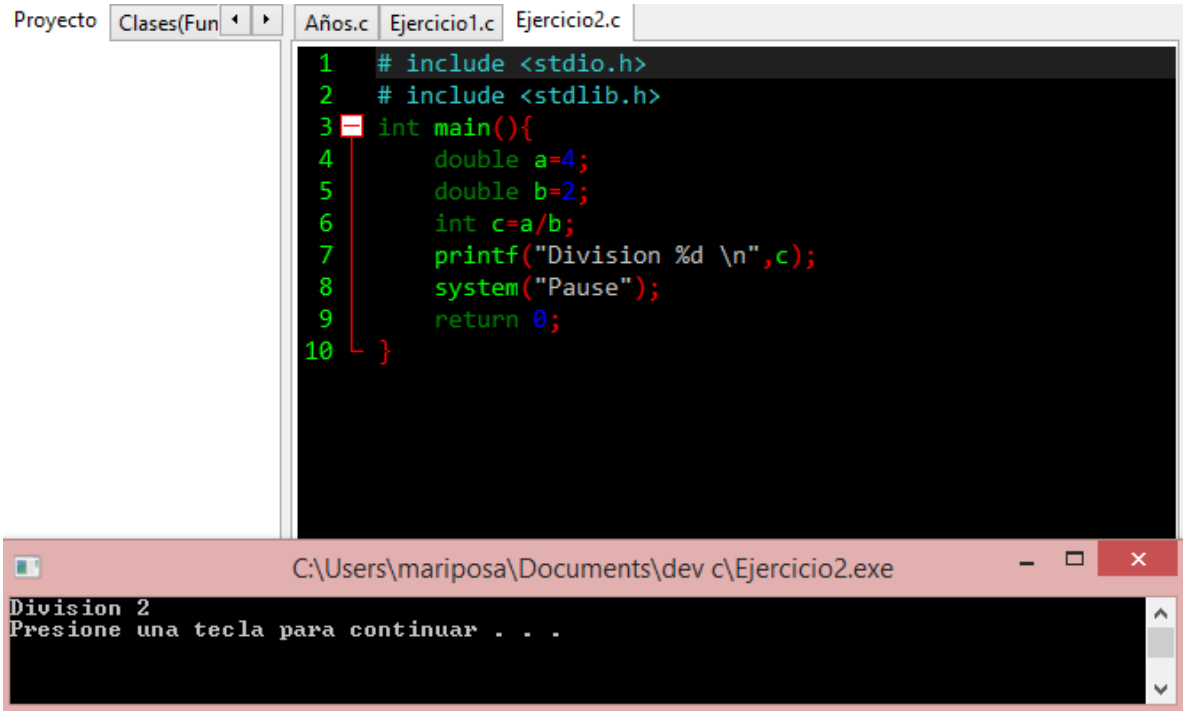
El ejercicio nos pide que calculemos en segundos los años vividos, por lo que declaramos una variable de tipo entero donde su valor es la multiplicación de los años, por los días del año, horas, minutos segundos. En el mensaje ingresamos un texto dentro de las comillas, pero después recordemos que debe de llevar el espacio en memoria, así como el nombre de la variable. En la ejecución nos aparece nuestro mensaje como el resultado.



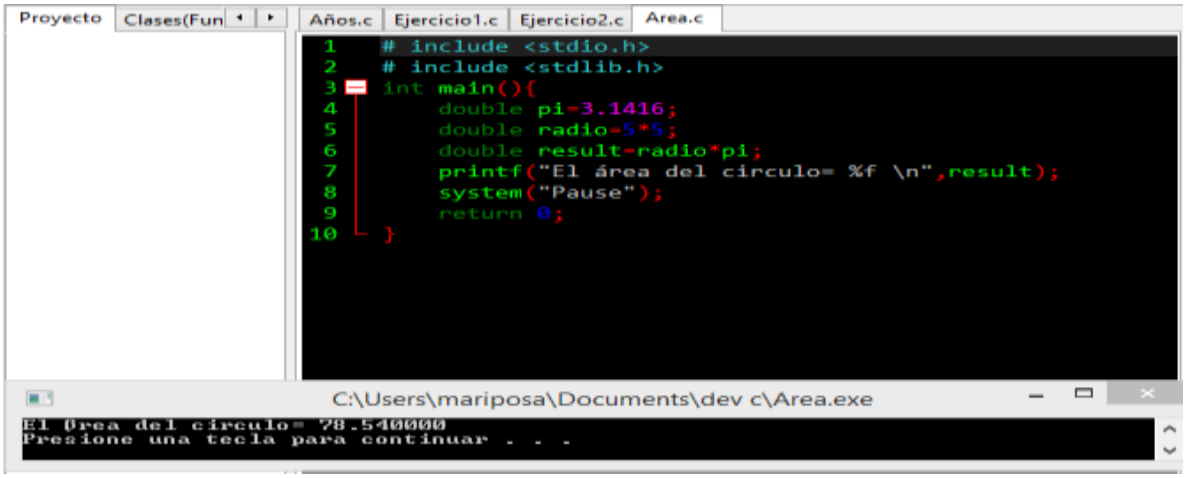
Seguimos con la misma practica de realizar operaciones, con la declaración de variables, mensaje, espacio en memoria, los estándares.



Seguimos con la misma practica de realizar operaciones, con la declaración de variables, mensaje, espacio en memoria, los estándares



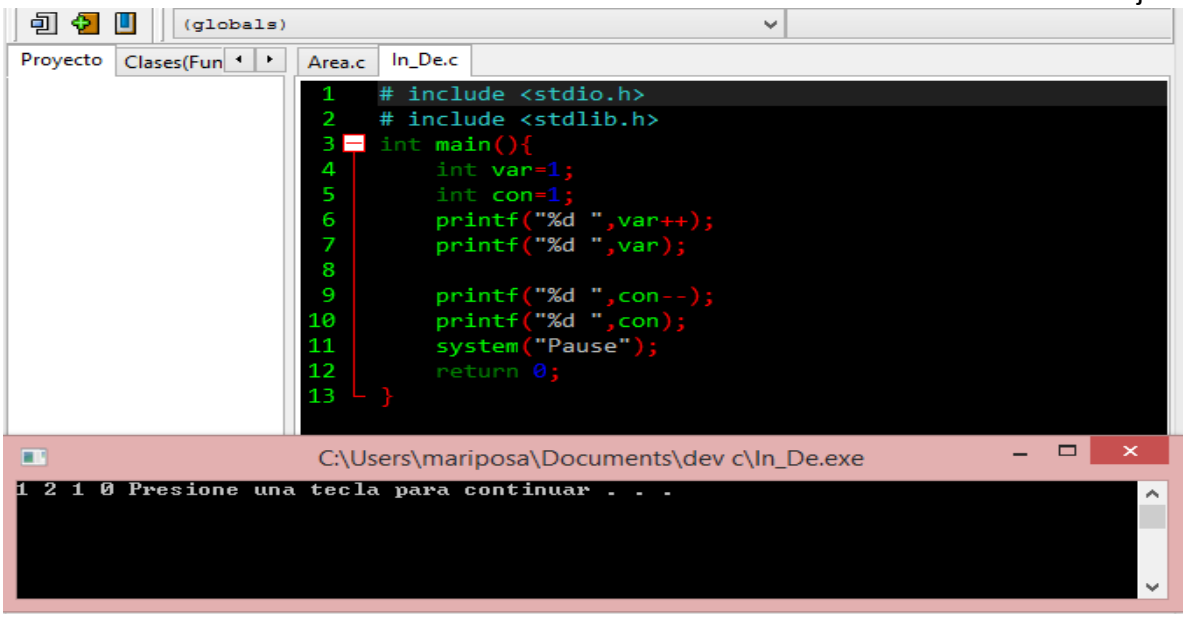
Continuamos con la misma practica de realizar operaciones, con la declaración de variables, mensaje, espacio en memoria, los estándares y teniendo en cuenta el tipo ya sea entero o decimal



OPERADOR INCREMENTO

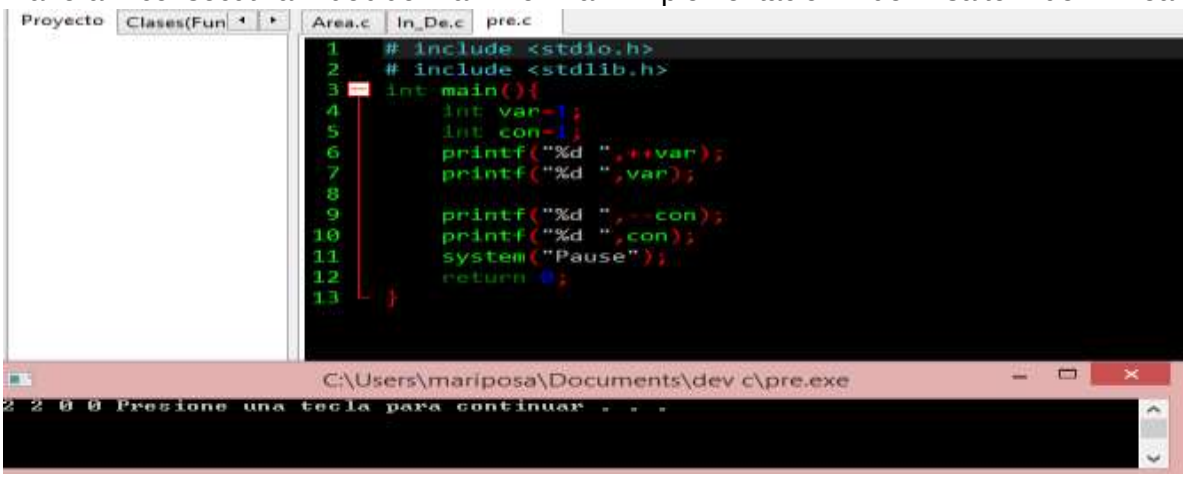
Este ejercicio nos dice que declaramos dos variables con su valor, pero en la impresión de mensaje tenemos un incremento-decremento, esto quiere decir que a la variable "var++" se suma 1, después no se le hace nada, ahora se le resta 2.

Esto nos queda de la siguiente manera en impresión, los resultados e muestra uno después de otro esto es porque no le pusimos el salto de línea dentro de las comillas que están en los mensajes.



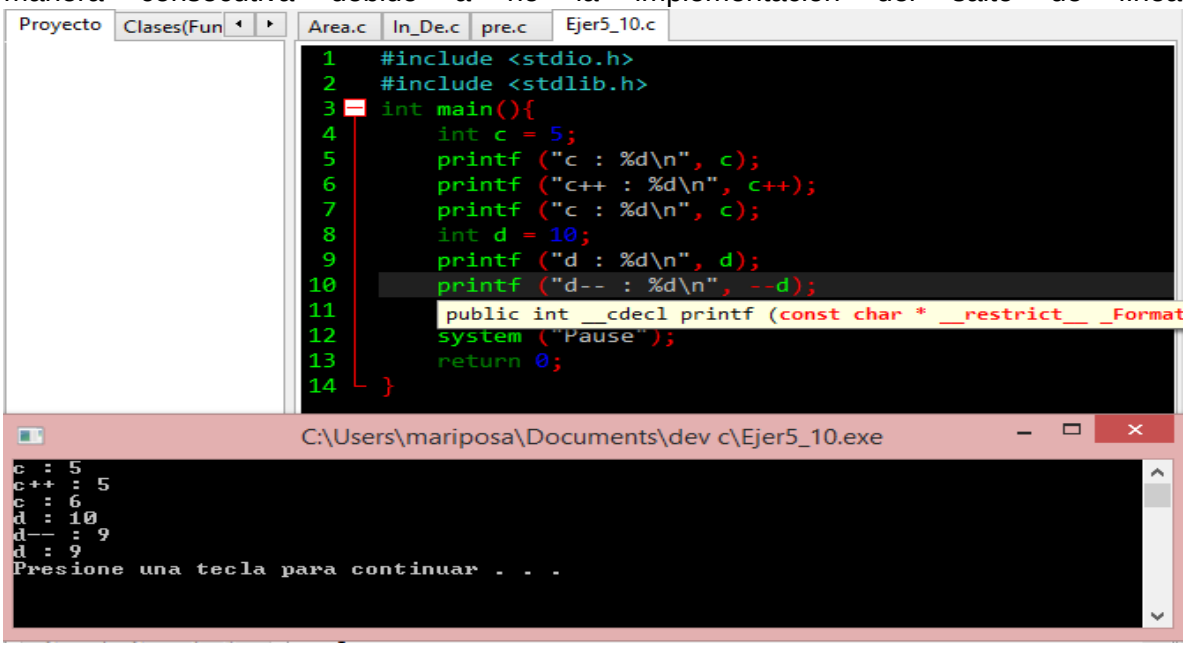
Estándares de entra y salida, la clase principal, se declaran variables con valor de 1, ahora se hace un post decremento que esto es antes de la variable donde su valor cambia como vale uno se le suma 1 o sea 2, pero con el otro caso se le resta entonces ahora el resultado es 0 en los dos casos.

Lo podemos ver en la ejecución del programa de igual manera los resultados están de manera consecutiva debido a no la implementación del salto de línea.



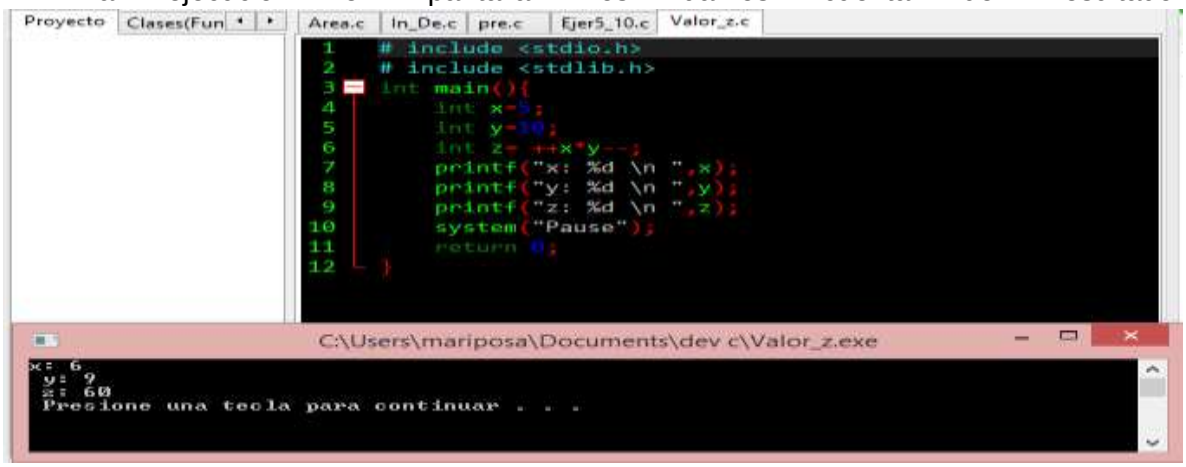
Estándares de entra y salida, la clase principal, se declara una variable de 5, ahora se hace incremento que esto es después de la variable donde su valor cambia como 5 se le suma 1 o sea 6, la otra variable vale 10, su valor en el decremento el valor es de 9.

Lo podemos ver en la ejecución del programa de igual manera los resultados están de manera consecutiva debido a no la implementación del salto de línea



En este ejercicio es encontrar el valor de z por medio de variables ya declaradas ocupando el pos-incremento y decremento por una multiplicación. Donde y vale 10 y x vale 6, mostrando los mensajes con las variables y el espacio en memoria, por el salto de línea por eso es que los números están uno debajo del otro.

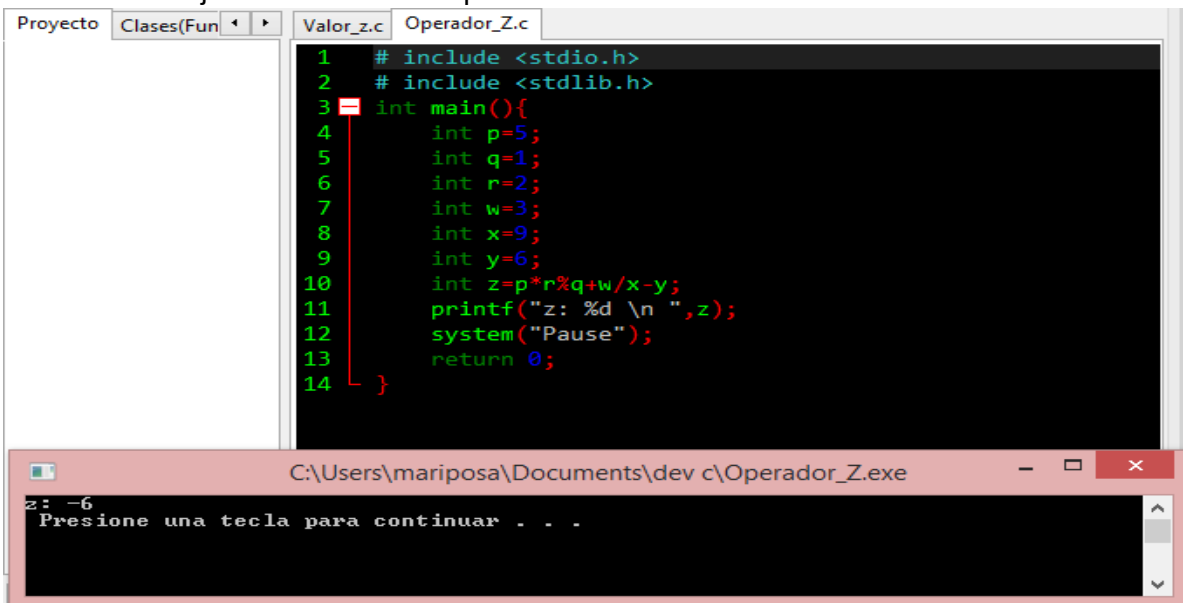
En la ejecución en pantalla nos damos cuenta del resultado.



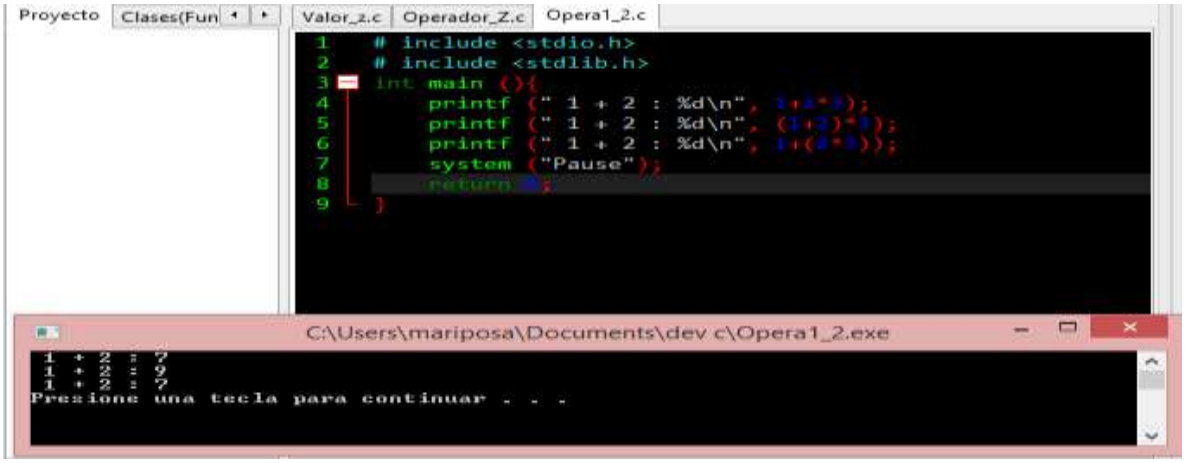
PRECEDENCIA DE OPERADORES

En este ejercicio declaramos variables de tipo entero con su valor, a z se le asigna operaciones de acuerdo a las variables donde hay multiplicaciones, resta, división. Para esto tomemos en cuenta la prioridad de los signos para obtener el resultado

En la ejecución en pantalla nos damos cuenta de ello.

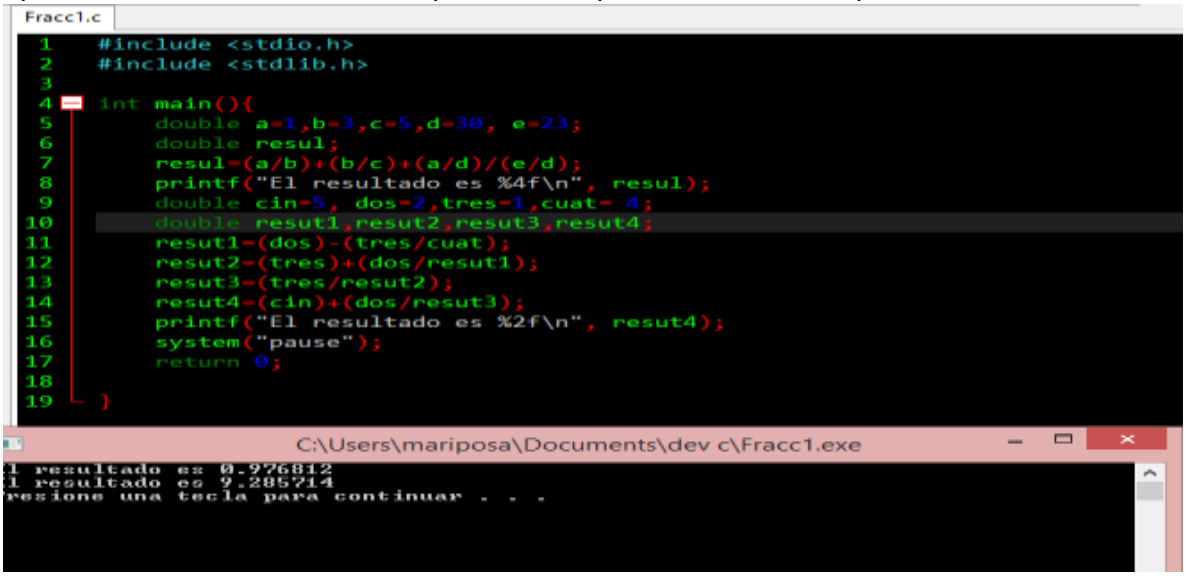


En este código hablamos más de los operadores y de la función de los paréntesis, donde modifican mucho los resultados, tenemos el salto de línea, reserva el espacio en memoria, en la ejecución nos damos cuenta.



El ejercicio consta de tener muy en cuenta los paréntesis, declaramos los valores de las variables para que los valores los tome de la variable, es suma, división de fracciones.

En el desarrollo del código nos damos cuenta de como funcionan los paréntesis ya que primero se realiza lo que está dentro de ellos. En la ejecución nos da el valor de las operaciones, en decimal por el tipo de dato que es un double.



OPERADORES LÓGICOS Y DE RELACIÓN

true	OR	true	=	True
true	OR	false	=	True
false	OR	true	=	True
false	OR	false	=	False

true	AND	true	=	True
true	AND	false	=	False
false	AND	true	=	False
False	AND	false	=	false

NOT	true	=	False
NOT	false	=	True

true	XOR	true	=	False
true	XOR	false	=	True
false	XOR	true	=	True
false	XOR	false	=	False

El operador lógico funciona de esa manera donde cada uno nos dará un resultado diferente de acuerdo a si es un or, y, xor, negación.

Tenemos un ejemplo de cómo funciona cada uno sustituyendo el true=1, false=0, para poder obtener el resultado.

En la ejecución nos podemos dar cuenta como es que funciona en el y, solo uno puede ser verdadero, en el or, al menos un verdadero nos da verdadero, en el xor solo debe de existir un verdadero.

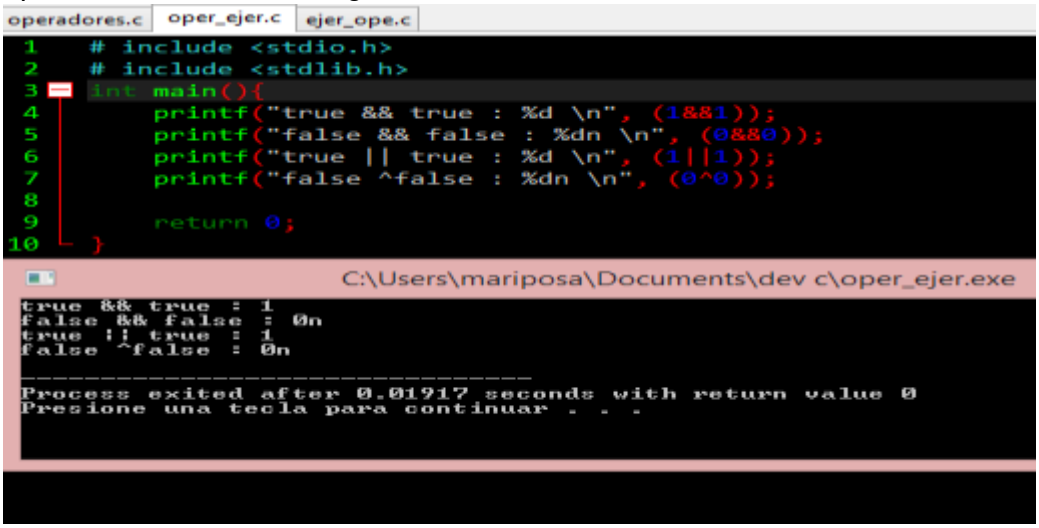
operadores.c

```c
# include <stdio.h>
# include <stdlib.h>
int main(){
    printf("***and***\n");
    printf("true && true : %d \n", (1&&1));
    printf("true && false : %d \n", (1&&0));
    printf("false && true : %d \n", (0&&1));
    printf("false && false : %dn \n", (0&&0));

    printf("***or***\n");
    printf("true || true : %d \n", (1||1));
    printf("true || false : %d \n", (1||0));
    printf("false || true : %d \n", (0||1));
    printf("false || false : %dn \n", (0||0));

    printf("***xor***\n");
    printf("true ^ true : %d \n", (1^1));
    printf("true ^ false : %d \n", (1^0));
    printf("false ^ true : %d \n", (0^1));
    printf("false ^false : %dn \n", (0^0));
    system("Pause");
    return 0;
}
```

```
C:\Users\mariposa\Do
***and***
true && true : 1
true && false : 0
false && true : 0
false && false : 0n
***or***
true || true : 1
true || false : 1
false || true : 1
false || false : 0n
***xor***
true ^ true : 0
true ^ false : 1
false ^ true : 1
false ^false : 0n
Presione una tecla para continuar . .
```

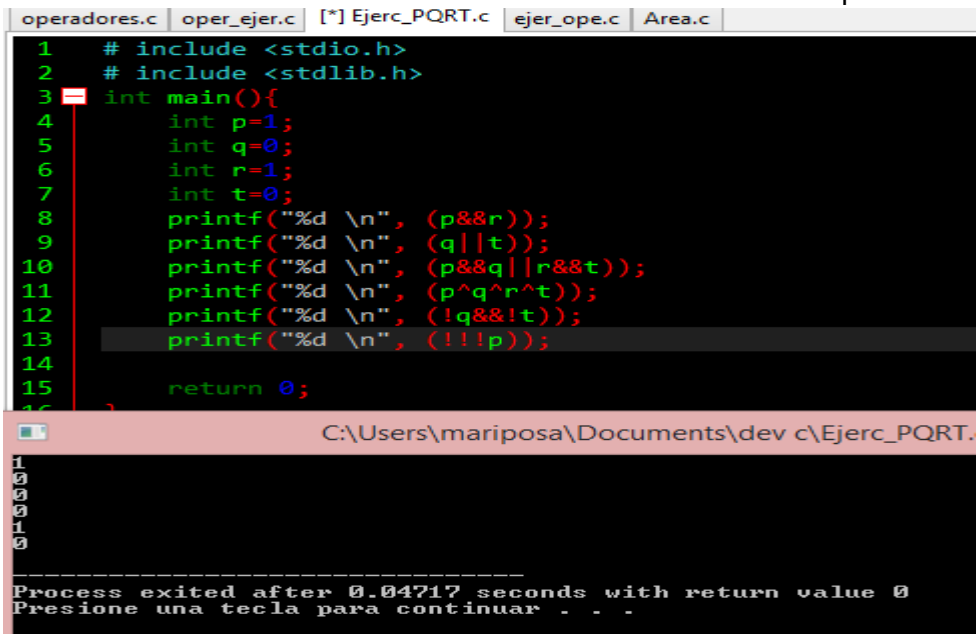
Ocupamos los estándares de entrada y salida, así como la clase principal, el espacio en memoria y el salto de línea.

Tenemos otro ejercicio de igual manera donde el true=1 y false=0, donde ocupamos operadores lógicos de manera diferente.

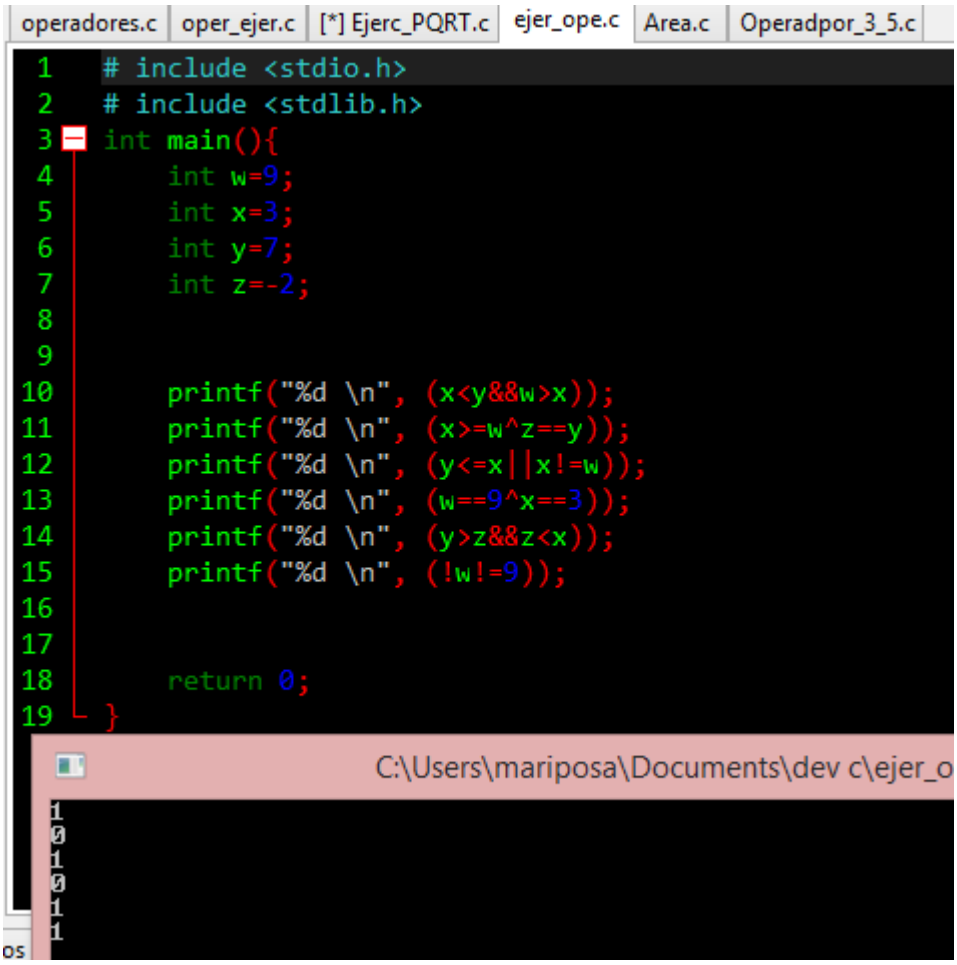


Definimos las variables su valor como el tipo de dato, de nuevo ocupamos los operadores lógicos y usamos paréntesis.

En la ejecución nos damos cuenta como el valor cambia de acuerdo a lo que se pide, bueno de acuerdo a los operadores lógicos.



En este ejercicio declaramos variables de tipo entero así como su valor, en la impresión nos damos cuenta de las operaciones que se realizan en este caso es de operaciones lógicas, donde son mas de una.



V. Conclusiones:

Es un programa por un conjunto de funciones que al menos contiene la función main, de valor que retorna y una lista de valor que retorna y una lista vde argumentos encerrados en paréntesis. El cuerpo está formado por sentencias y declaraciones comprimidas entre llaves.