|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nombre de la práctica** | **LENGUAJE C Y VARIABLES C** | | | **No.** | **1** |
| **Asignatura:** | **METODOS NUMERICOS** | **Carrera:** | **ING. SISTEMAS COMPUTACIONALES** | **Duración de la práctica (Hrs)** | **10** |

**Nombre del alumno: Velasco Alcántara Fernando Emmanuel**

**I. Competencia(s) específica(s):**

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

Otro

**III. Material empleado:**

Dev C++

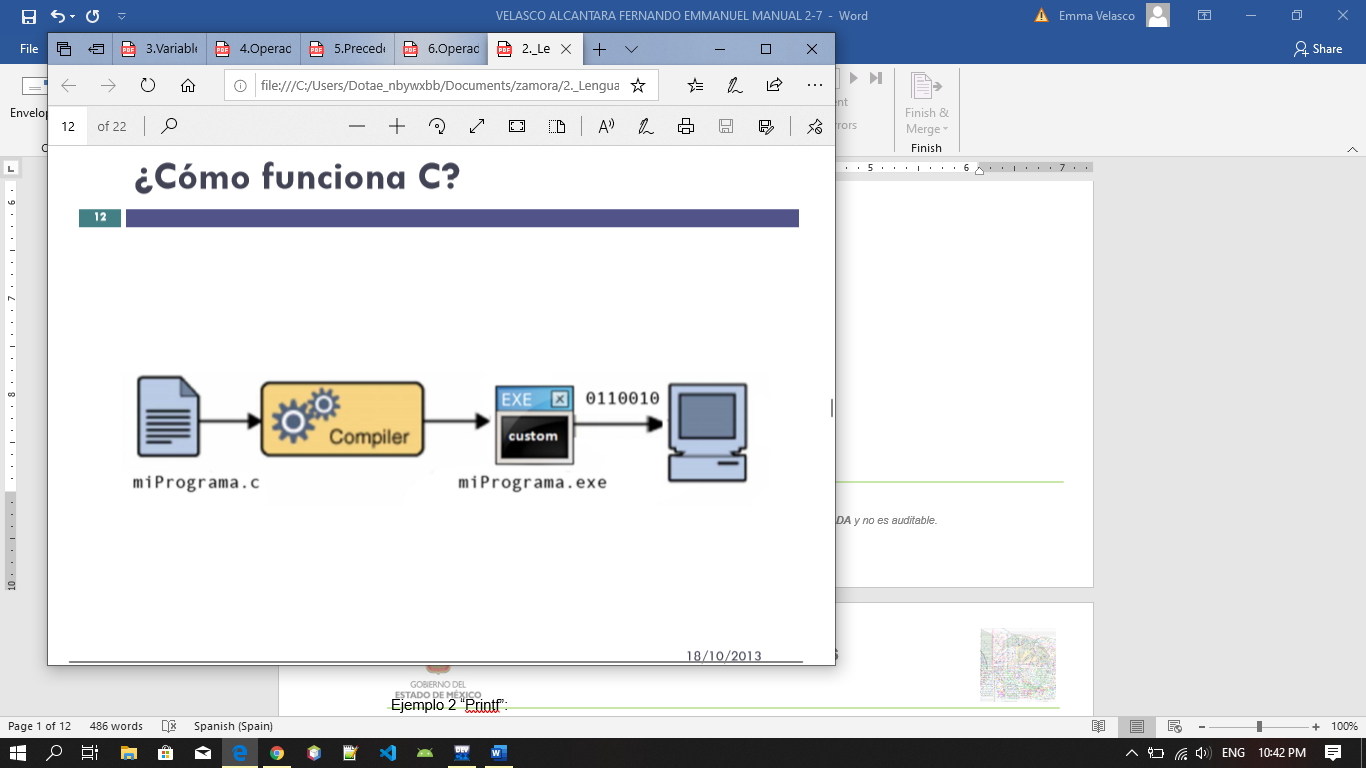
**IV. Desarrollo de la práctica:**

¿Qué es C?

* C es un lenguaje de programación estructurada desarrollado a principios de los años 70.
* Es un lenguaje de nivel medio, pero con muchas características de bajo nivel.
* Está disponible en un amplio abanico de plataformas

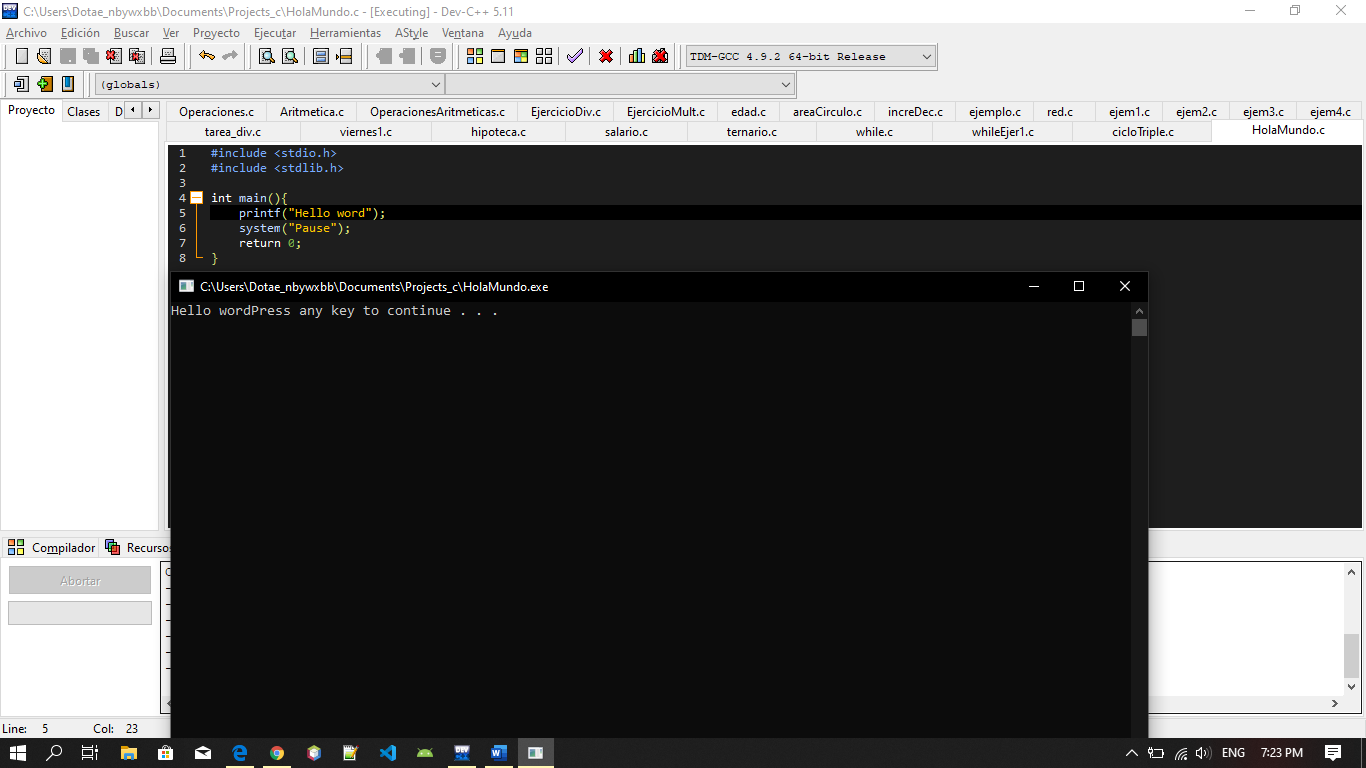
¿Como funciona?

Al igual que en otros lenguajes, se crea un documento con la extensión del leguaje, después este se compila, crea un archivo ejecutable que envía las instrucciones en código binario a nuestro ordenador.



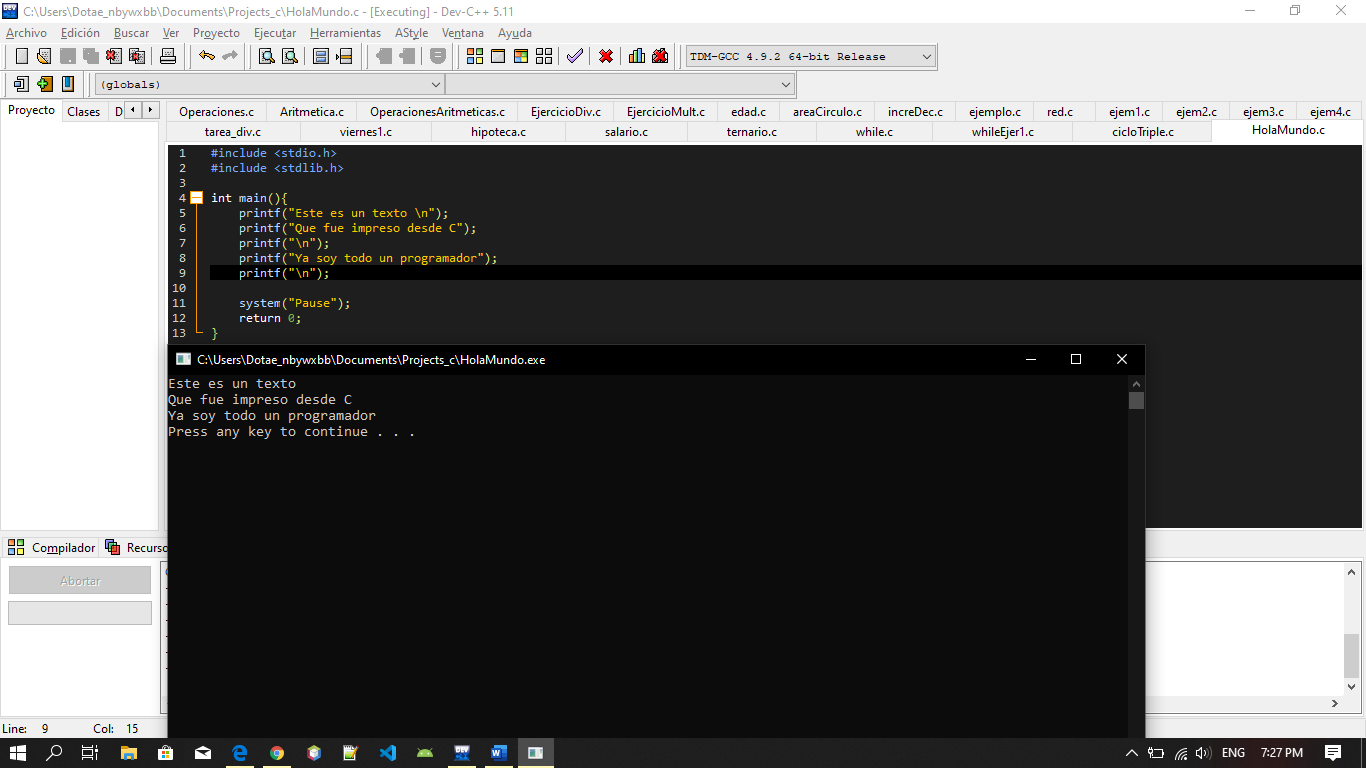
Ejemplo 1 “Hola mundo”:

Como en cada lenguaje de programación se tiene que realizar el “Hola mundo” para evitar tragedias según cuenta la leyenda 😊

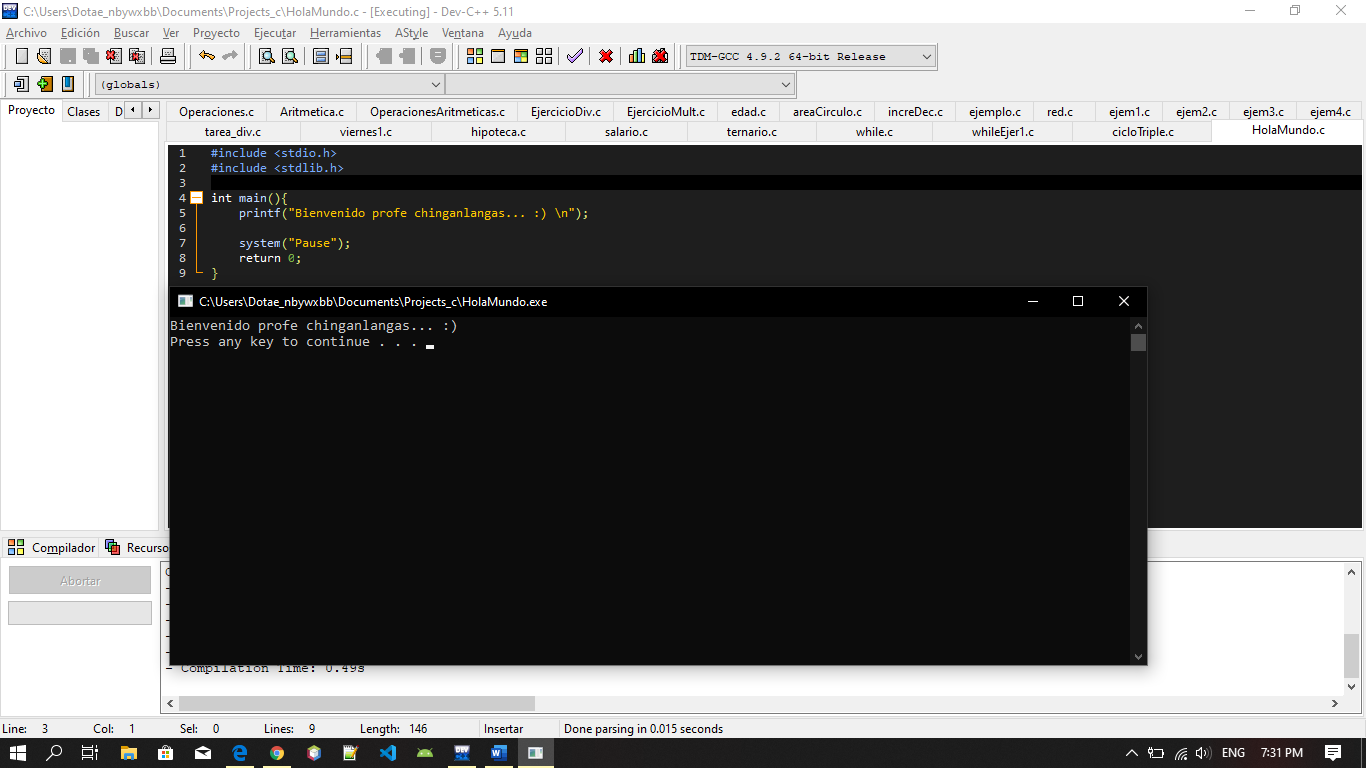


Ejemplo 2 “Printf”:

Mediante el uso de “printf” se esta enviando impresiones de texto.

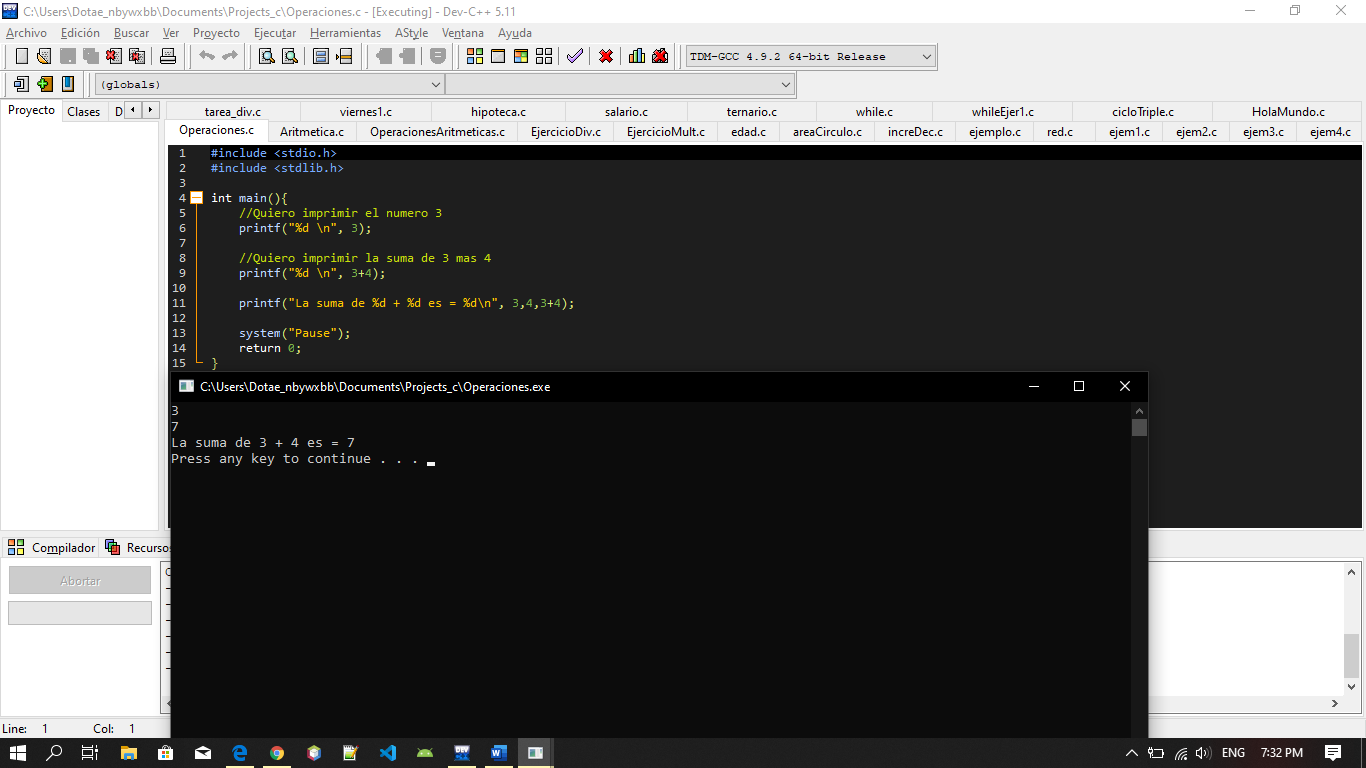


Ejercicio 1 “Crea un programa que imprima una bienvenida con tu nombre”:



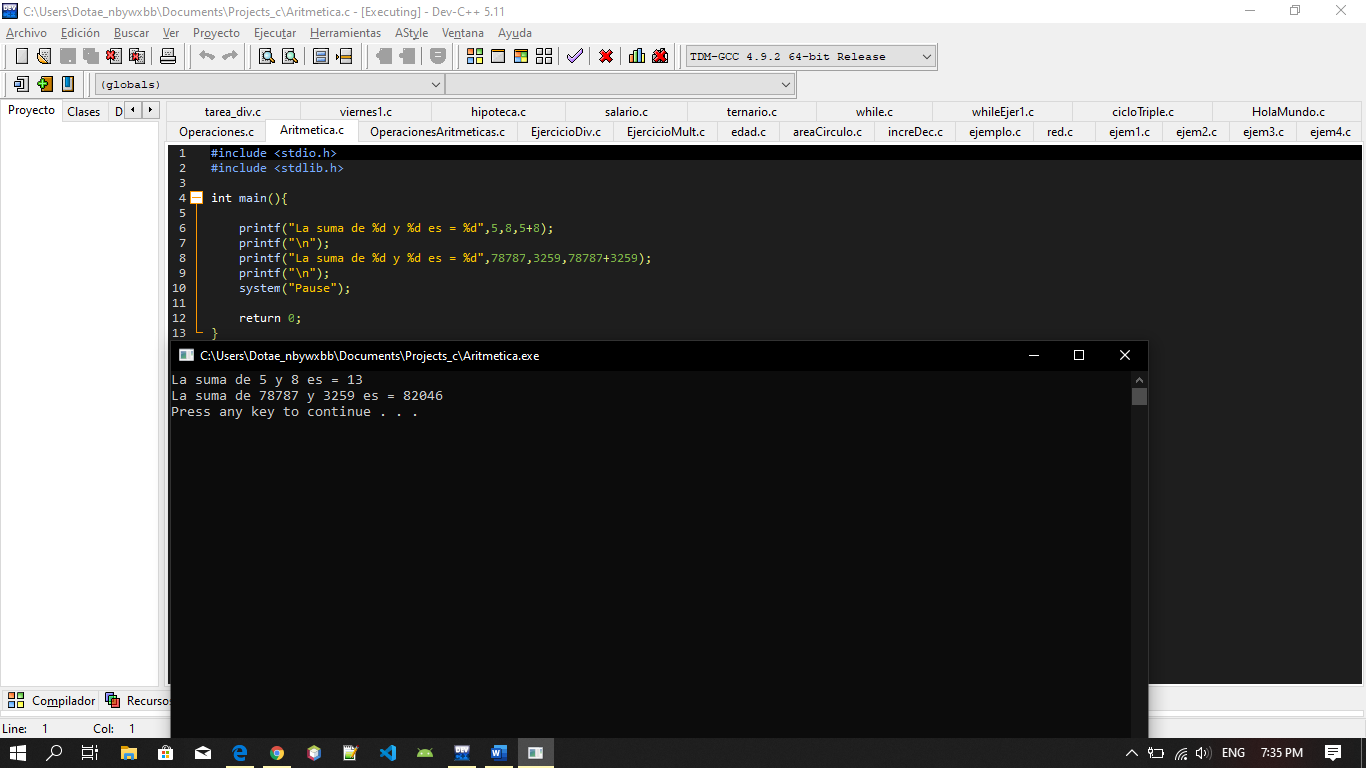
Ejemplo 3 “Operaciones matemáticas con comentarios”:

En este ejemplo se realizaran operaciones, esto es algo de poca complejidad, pues lo único que requerimos son los especificadores de conversión para poder agregar números a nuestro mensaje, a estos se les agrega valor colocando una coma después de las comillas con el valor o la variable que queremos ingresar, ahí mismo se pueden realizar las operaciones.



Ejercicio 2 “Escribir un código que realice la suma de dos números: 5+8 y 78787+3259”:\

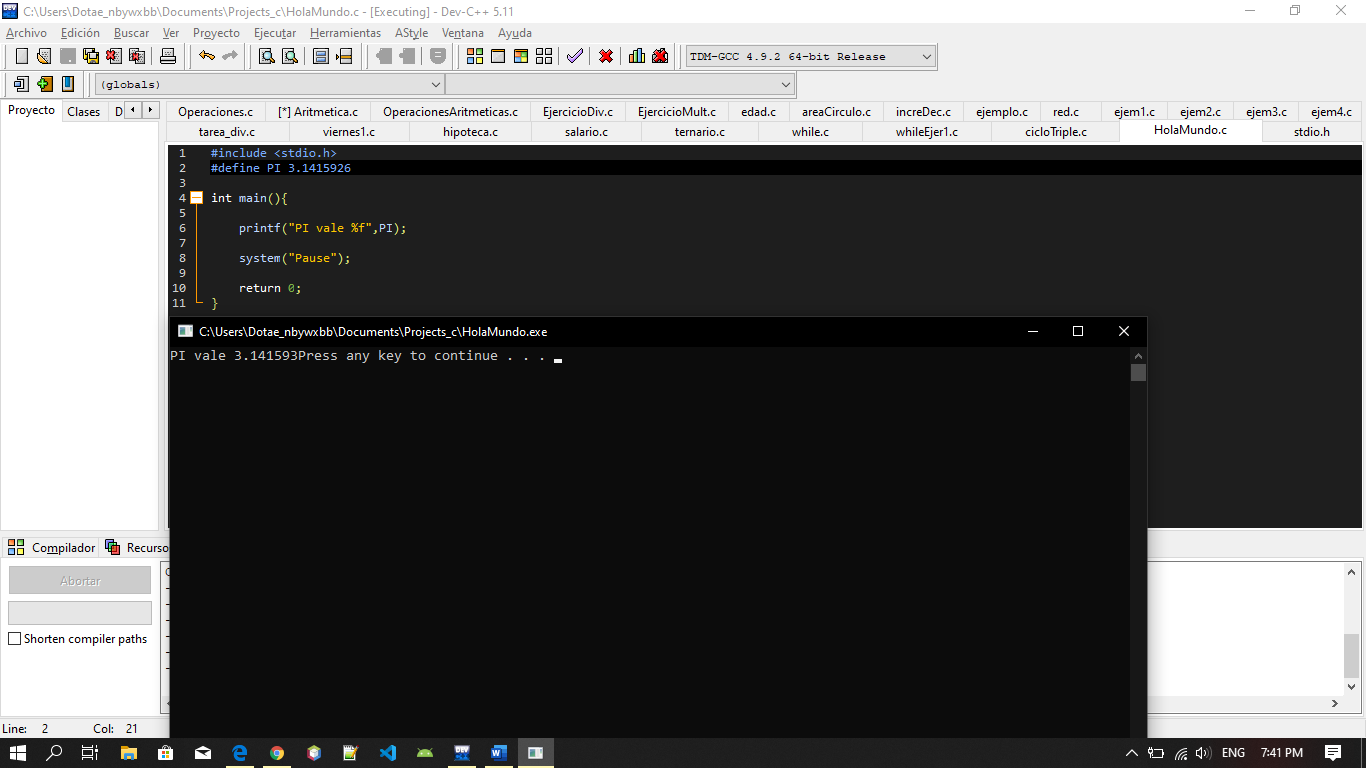
Aquí podemos mostrar que se pueden agregar mas de un solo numero por mensaje, se agregaran los números que indiquemos con los especificadores de conversión, recordar que cada uno de los valores debe de ir separado con una coma.



VARIABLES

Ejemplo 1 “constantes”:

Una constante es un valor que no va a variar, para esto lo debemos definir al principio del código con el valor que queramos que tenga con la instrucción “#define”.



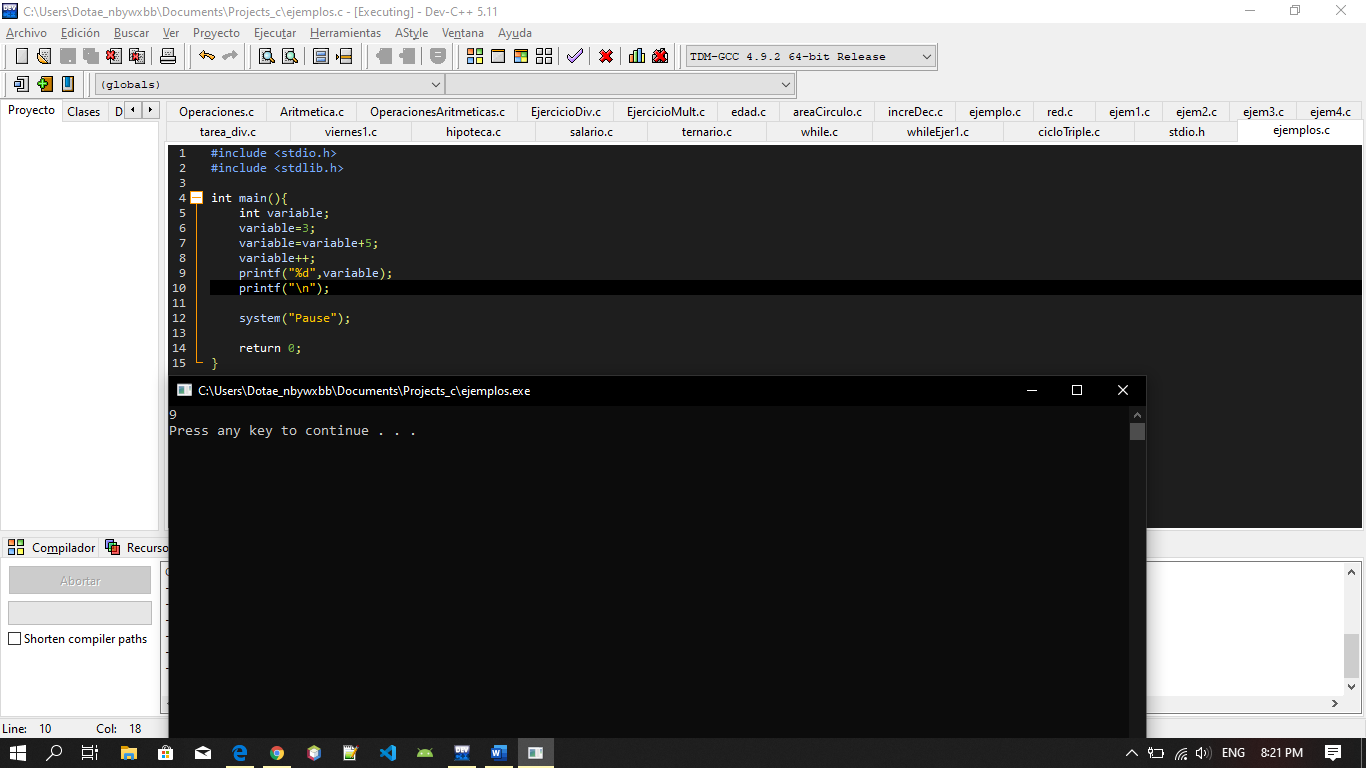
Ejemplo 2 “Variables 1”:

Una variable puede ser de diferentes tipos según lo que necesitemos, con ellas se pueden realizar diferentes operaciones y funciones.



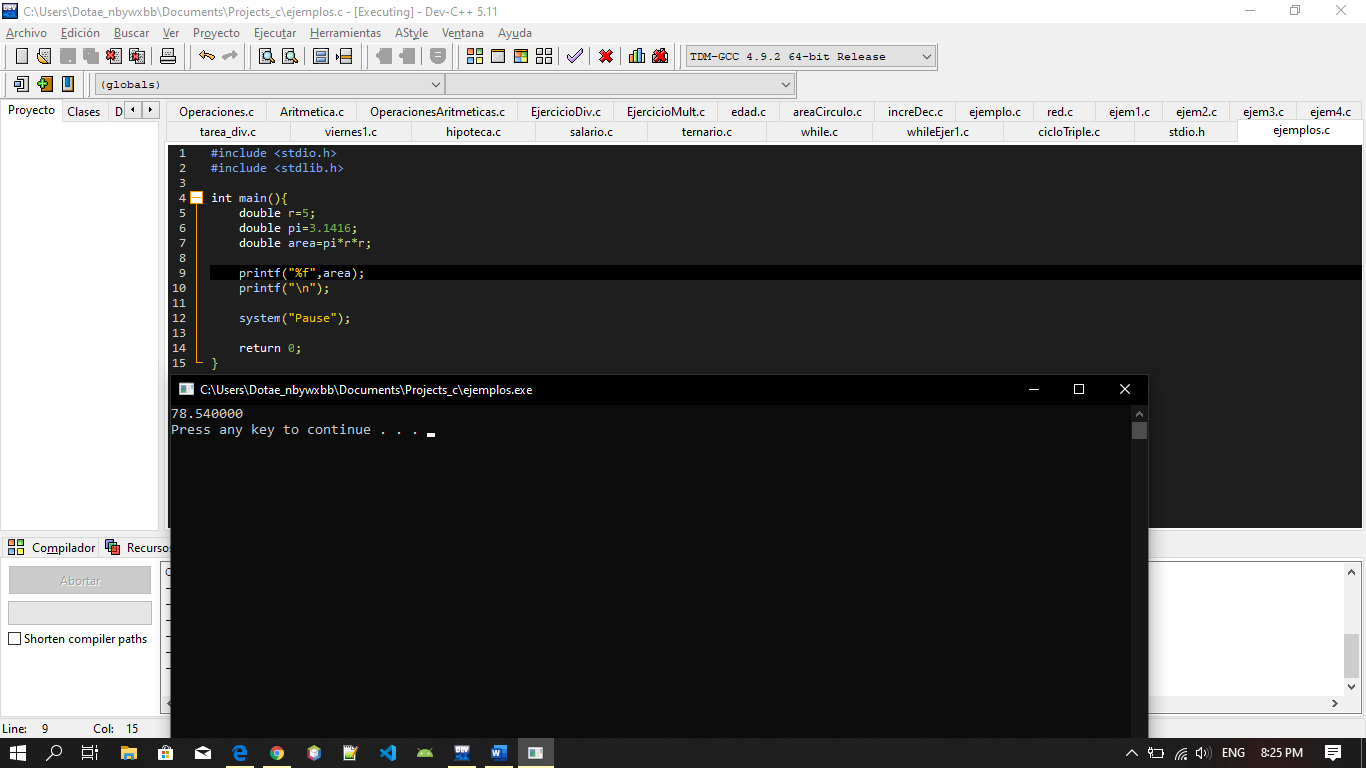
Ejemplo 3 “Variables 2”:

En este ejemplo se le esta aplicando un post decremento a nuestra variable, es decir que le estamos agrando una unidad al valor que tenga.



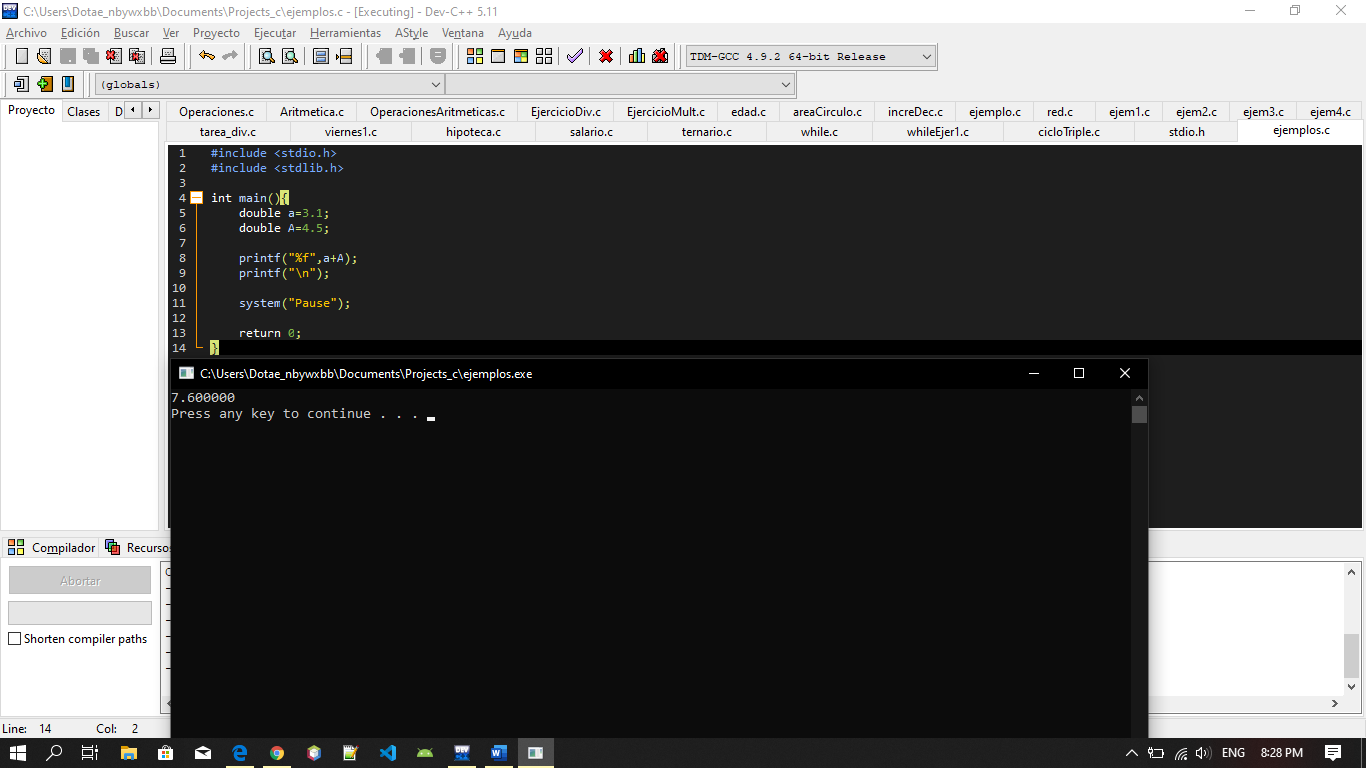
Ejemplo 4 “Variables 3”:

Las operaciones se pueden realizar de diferentes maneras, aquí se puede ver que estamos definiendo una variable con el uso de operaciones con otras variables ya definidas.

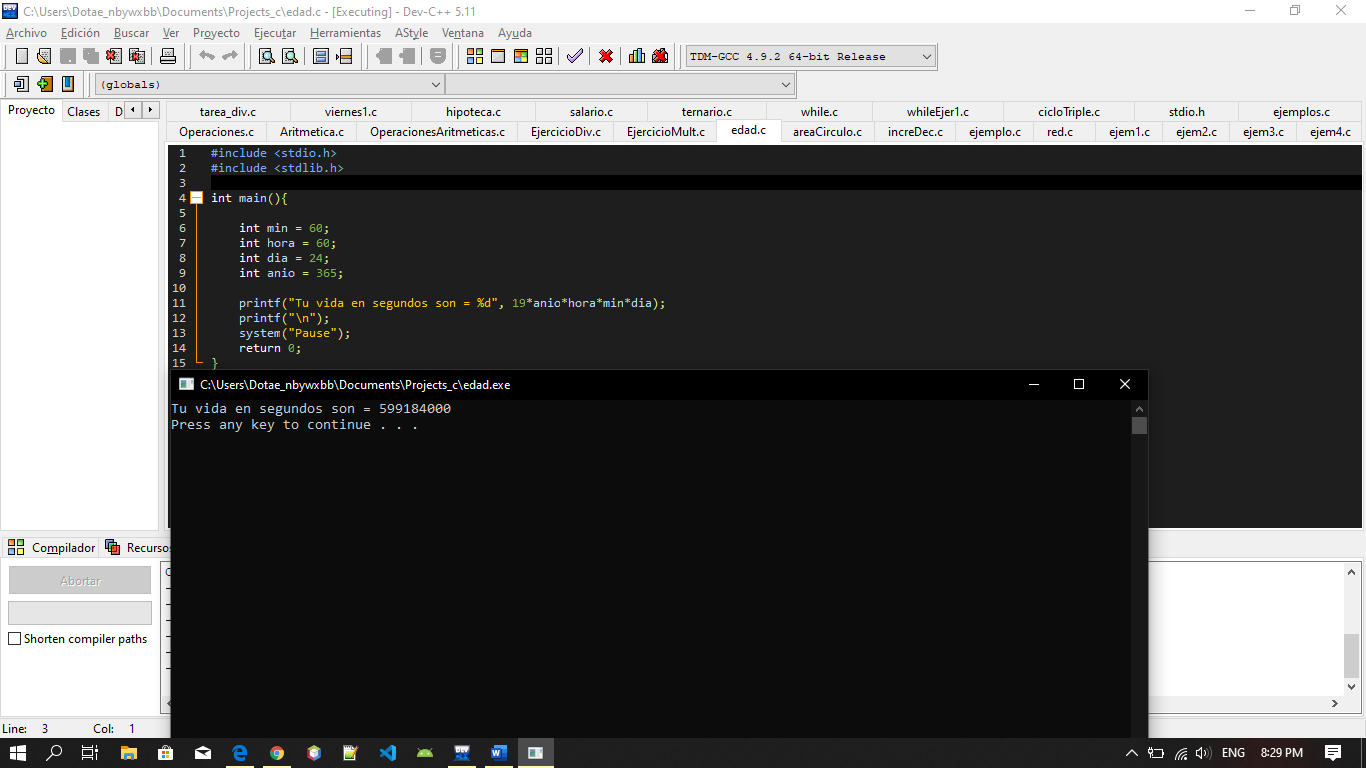


Ejemplo 5 “Variables 4”:

La otra forma es realizar la operación en la impresión de pantalla de esta manera:

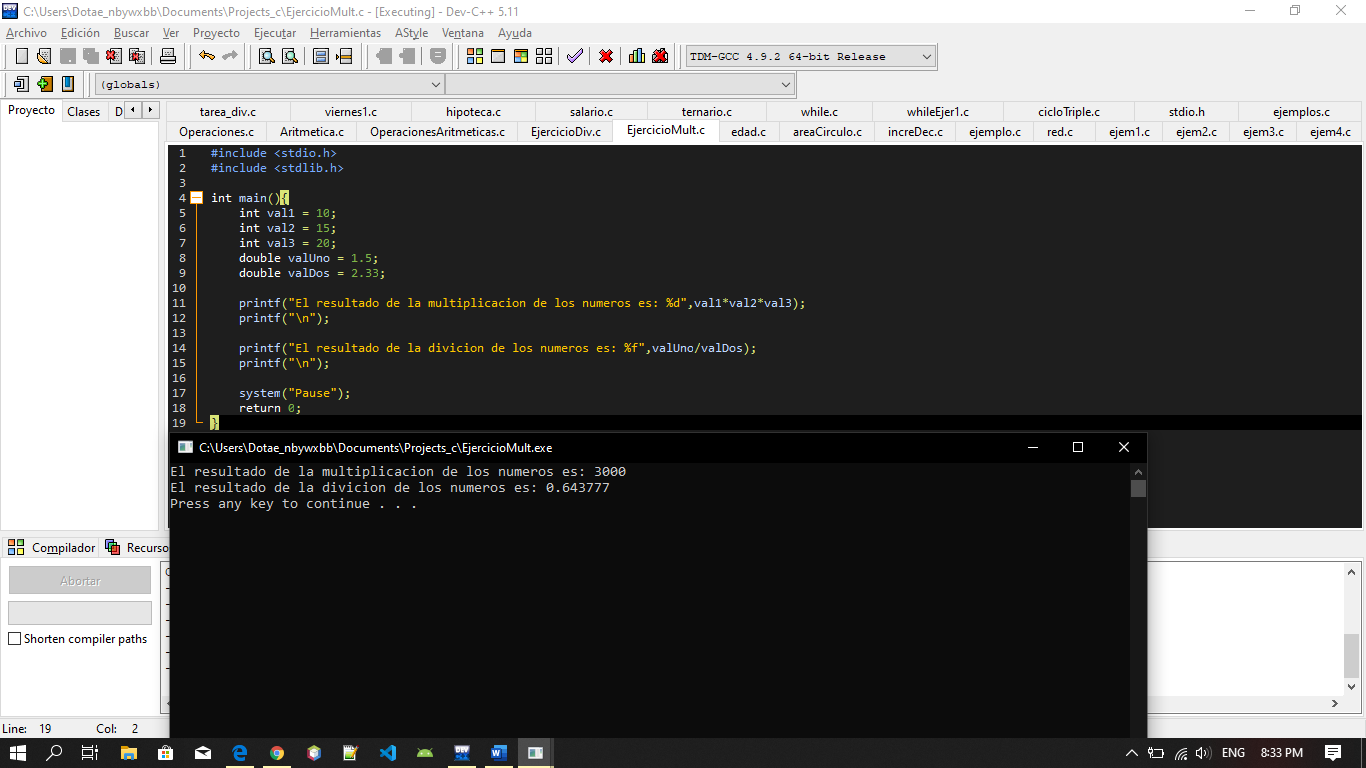


Ejercicio 1 “Calcula la cantidad de segundos que has vivido”:

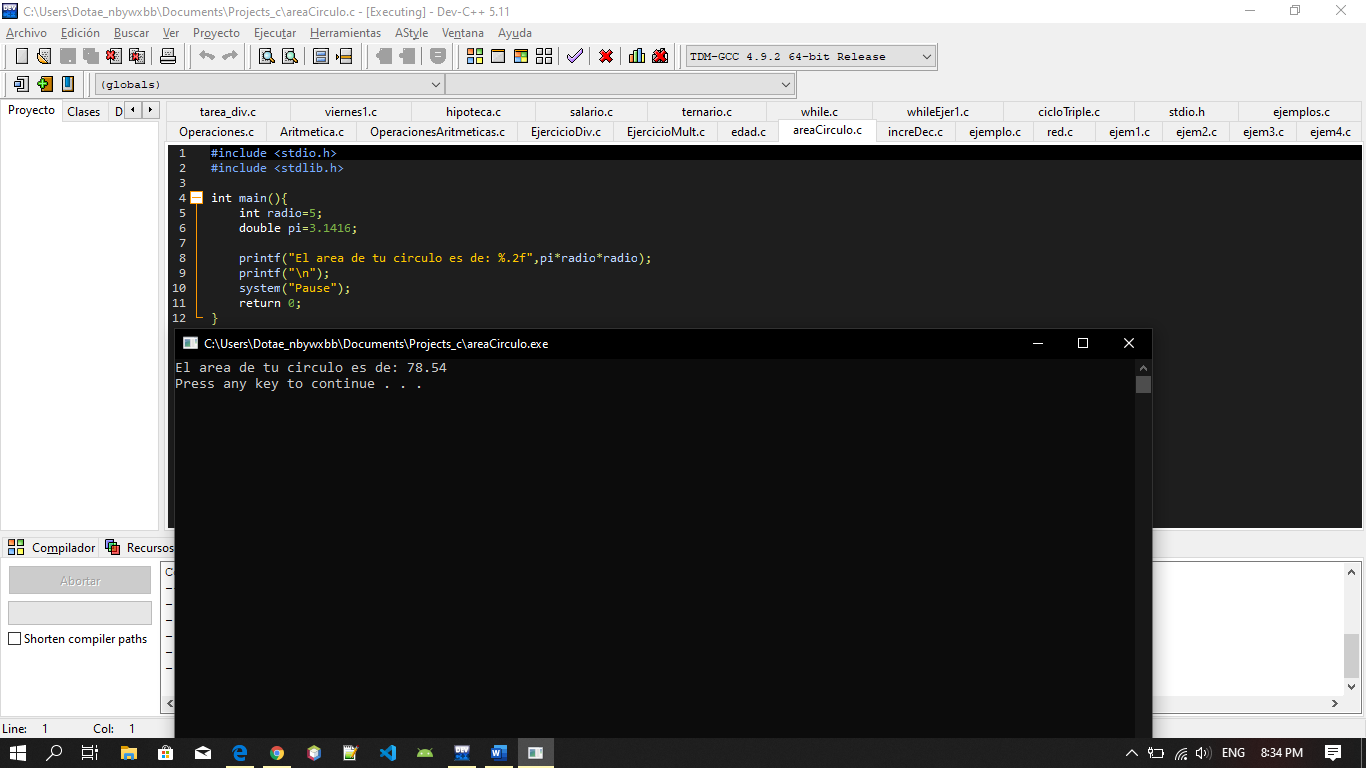


Ejercicio 2 “Declara 3 variables enteras y muestra su multiplicación. Declara dos variables dobles diferentes de 0 y divide la primera entre la segunda”:

Aquí se nota el uso de diferentes tipos de variables y como es que este tipo influye en la manera en que usaremos los especificadores de conversión.



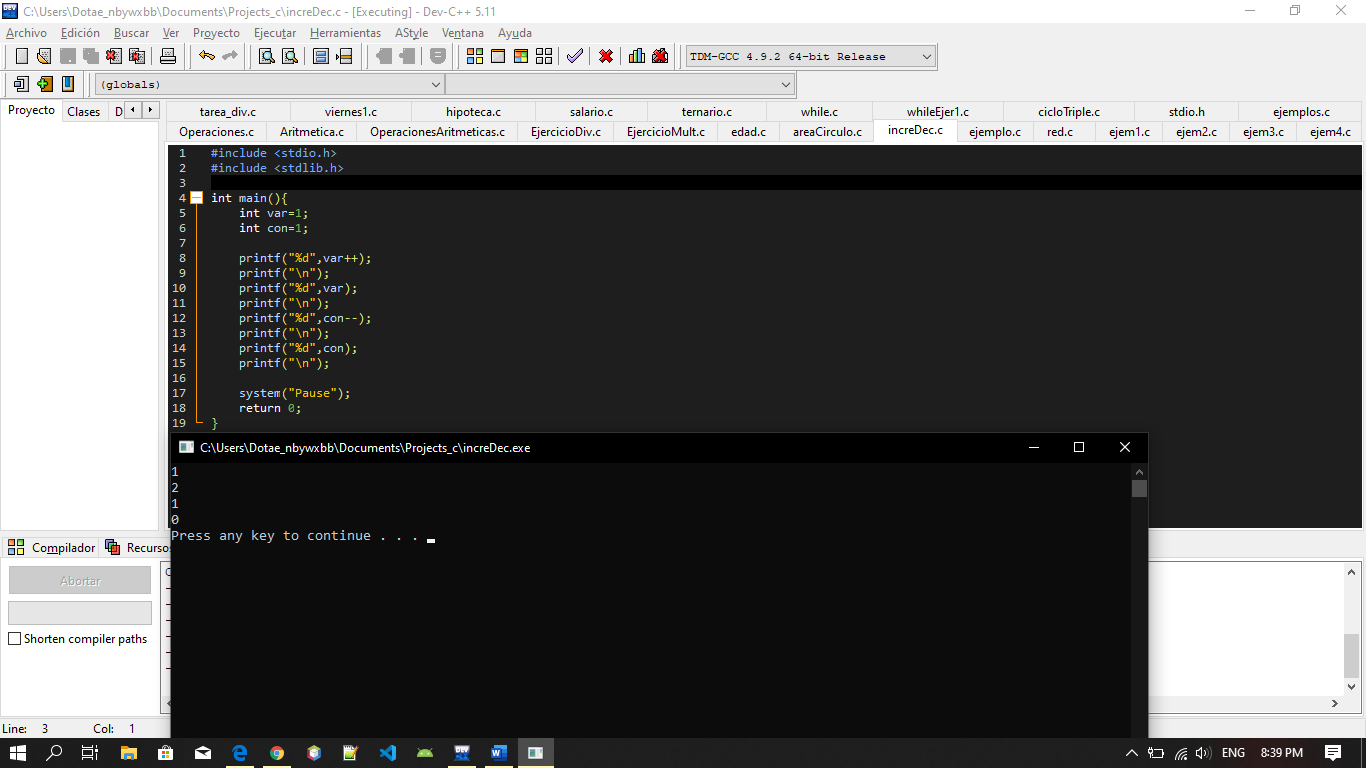
Ejercicio 3 “Crea un programa que calcule el área de un circulo de 10 metros de diámetro”:



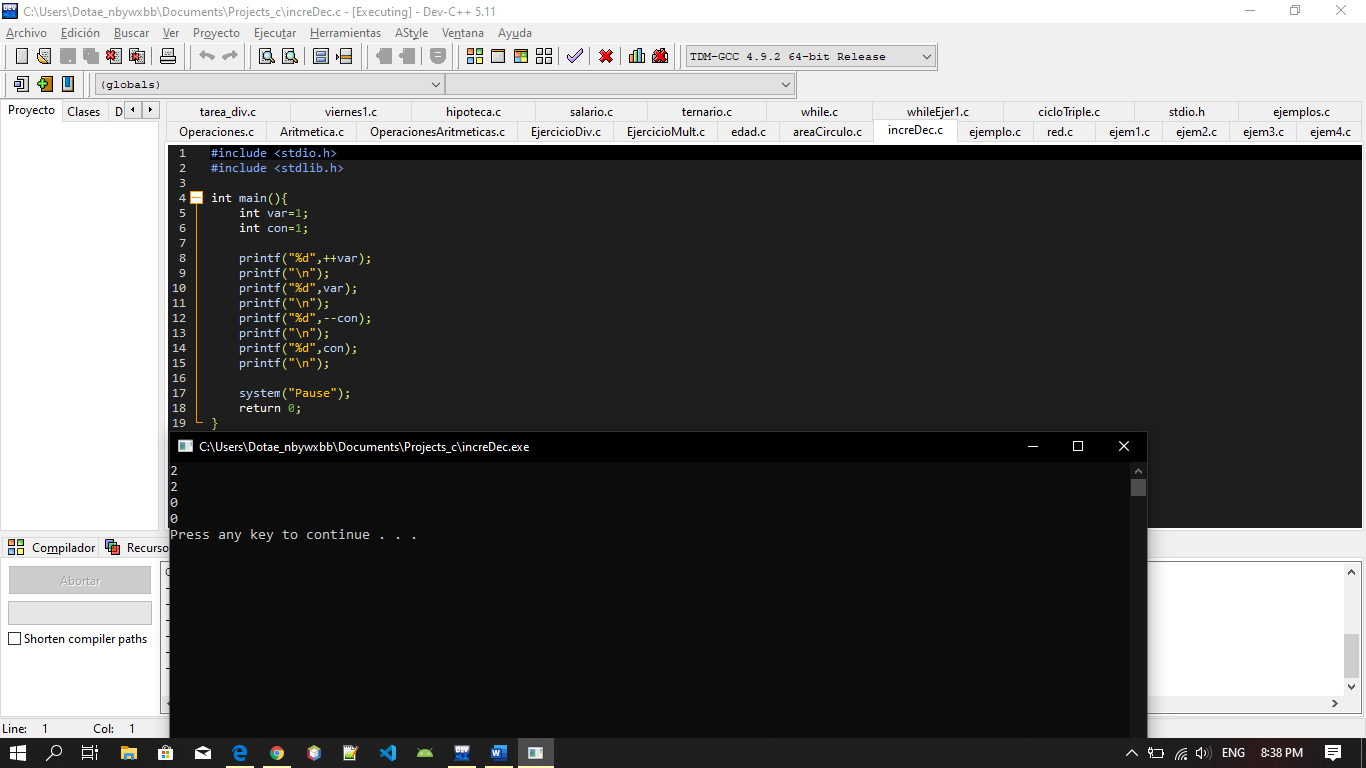
OPERADOR INCREMENTO/DECREMENTO

Estos operadores pueden ser utilizados para hacer incrementos a una variable o resultado, pero lo realiza al final, de esa misma manera se hacen decrementos, estos son llamados “post” también tenemos los “pre” estos hacen lo mismo pero al inicio.

Ejemplo 1 “Post incremento/decremento”:

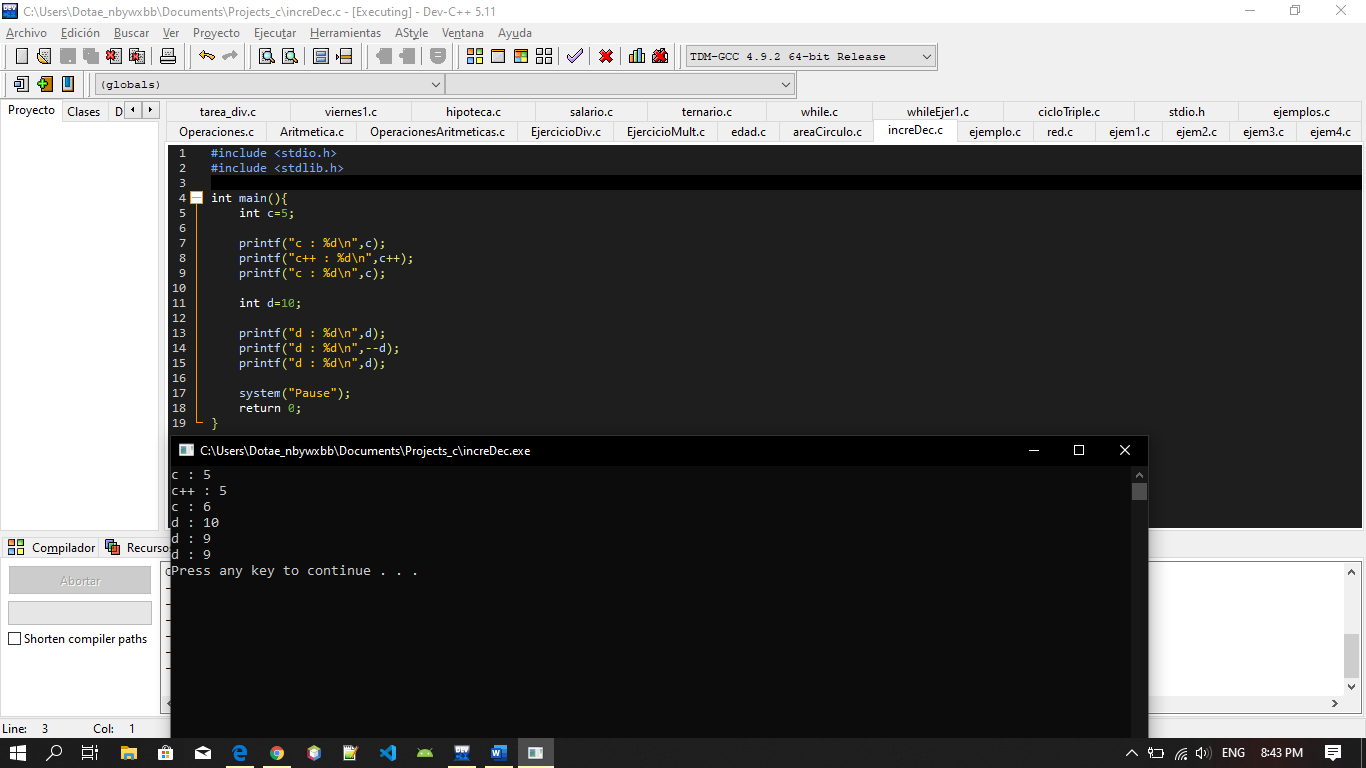


Ejemplo 2 ”Pre incremento/decremento”:

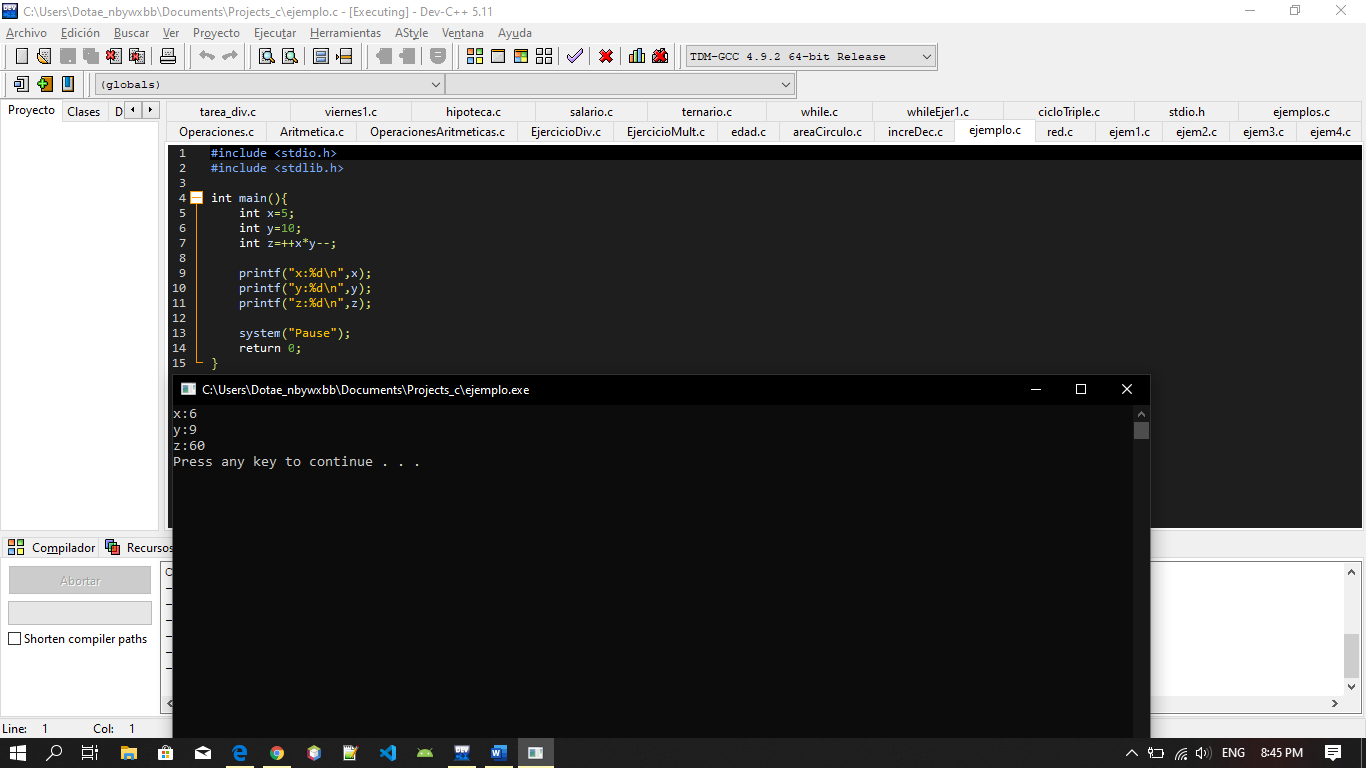


Ejemplo 3 “Operadores incremento/decremento”:

Aquí se muestran ejemplos de un post incremento y un pre decremento, imprimiendo la variable antes y después de realizar dicha función.

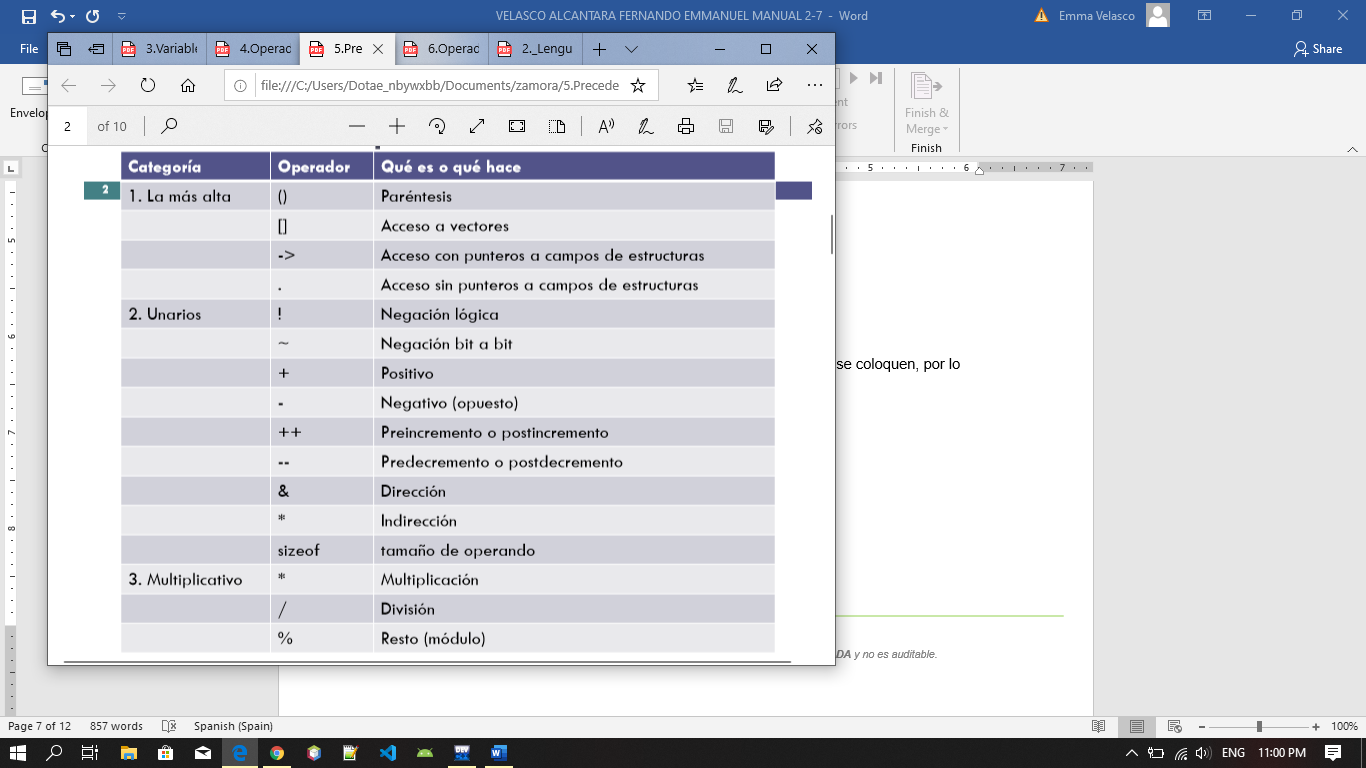


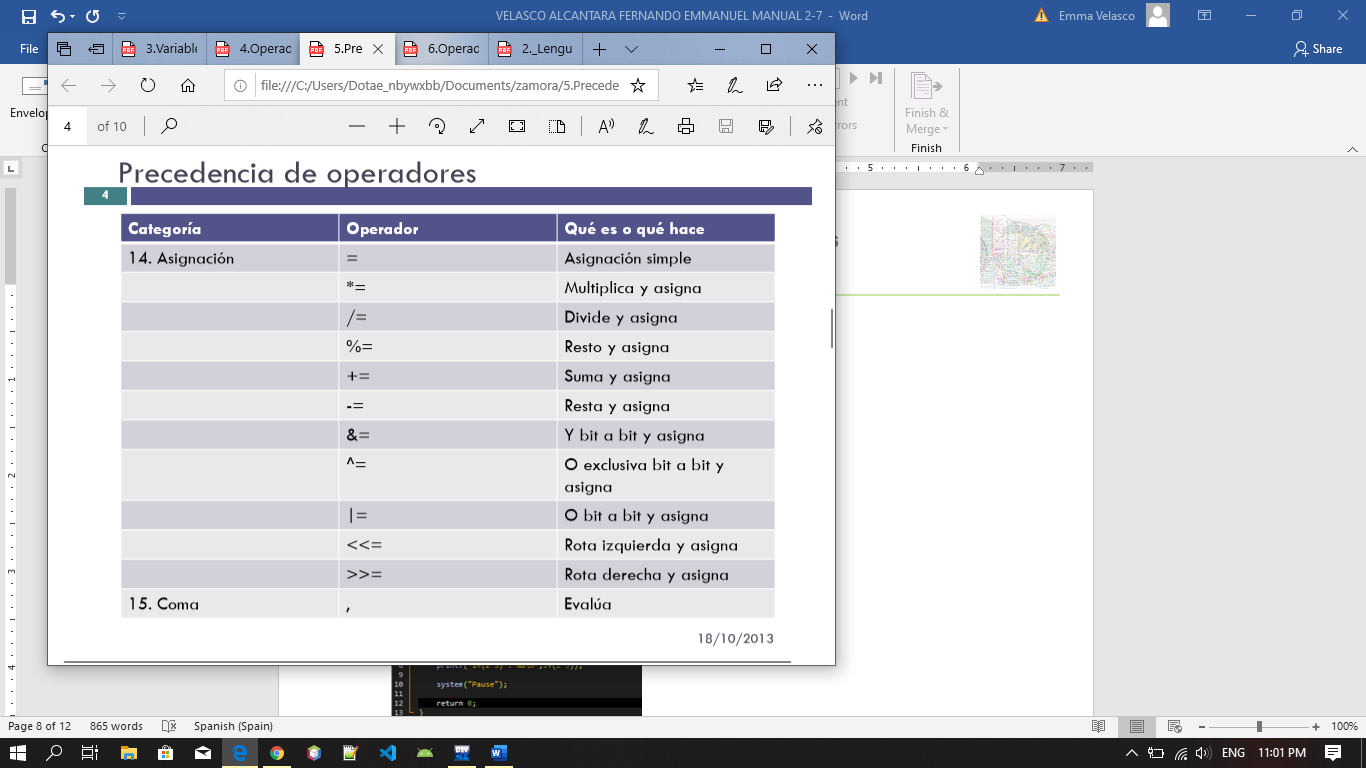
Ejemplo 4 “Operadores incremento/decremento”:



PRECEDENCIA DE OPERADORES

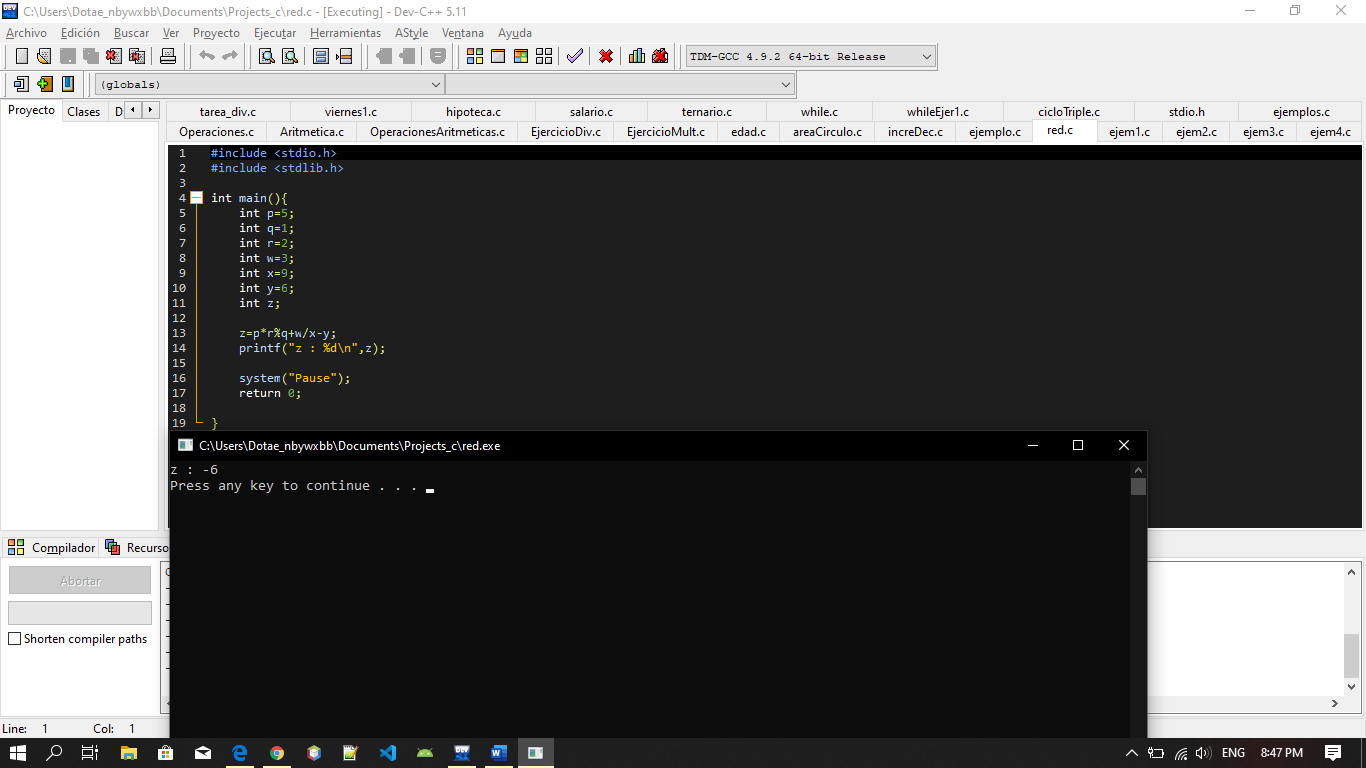
Los operadores pueden cambiar el resultado según el orden en que se coloquen, por lo tanto es importante que lleven un orden.

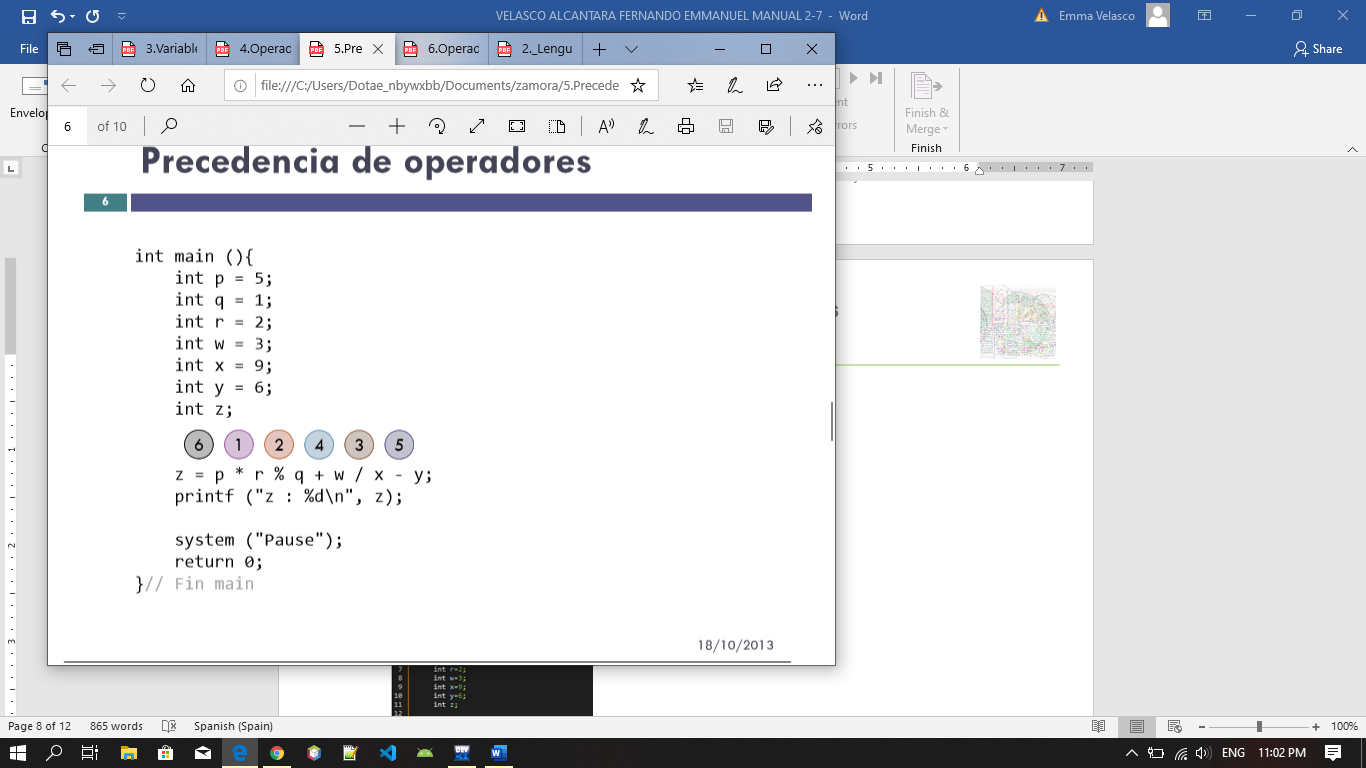


Ejemplo 1 “Precedencia de operadores”:

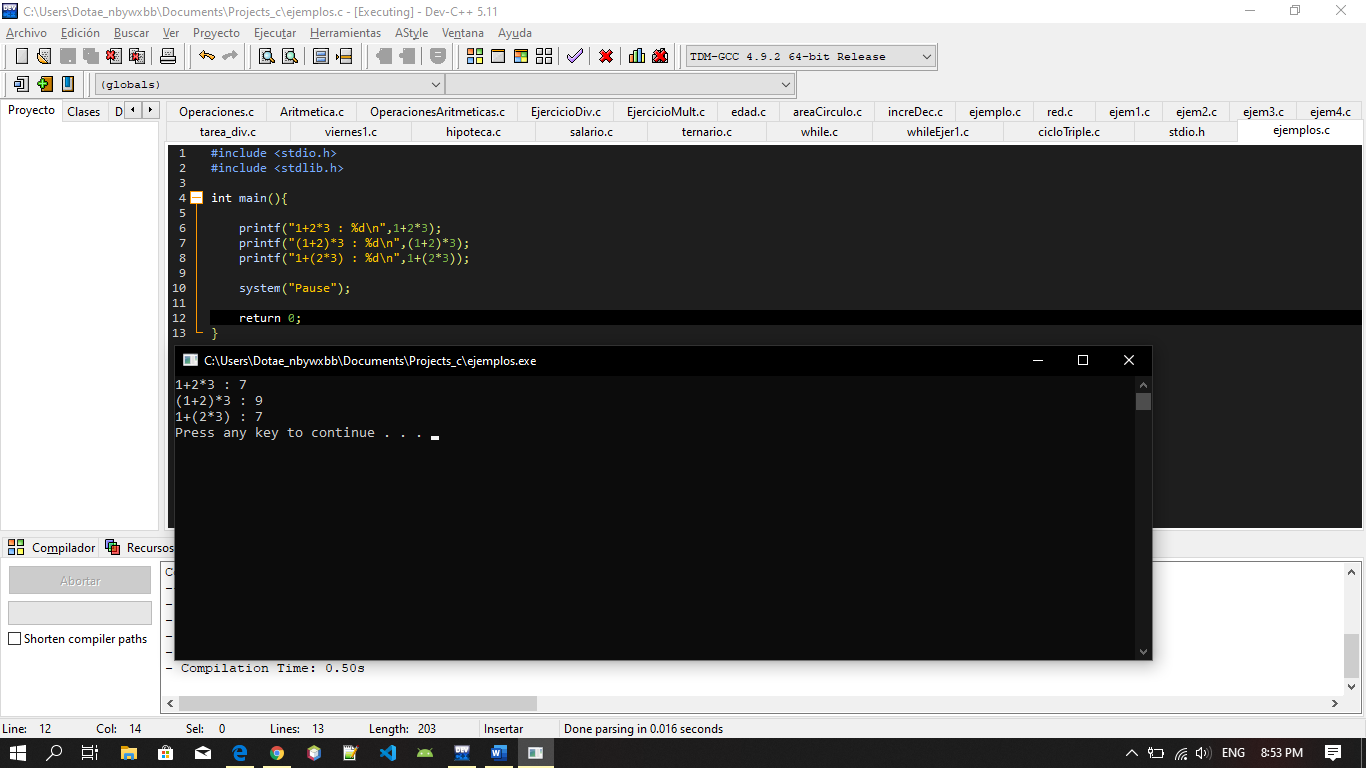
Aquí mostramos un ejemplo de uso de operadores.



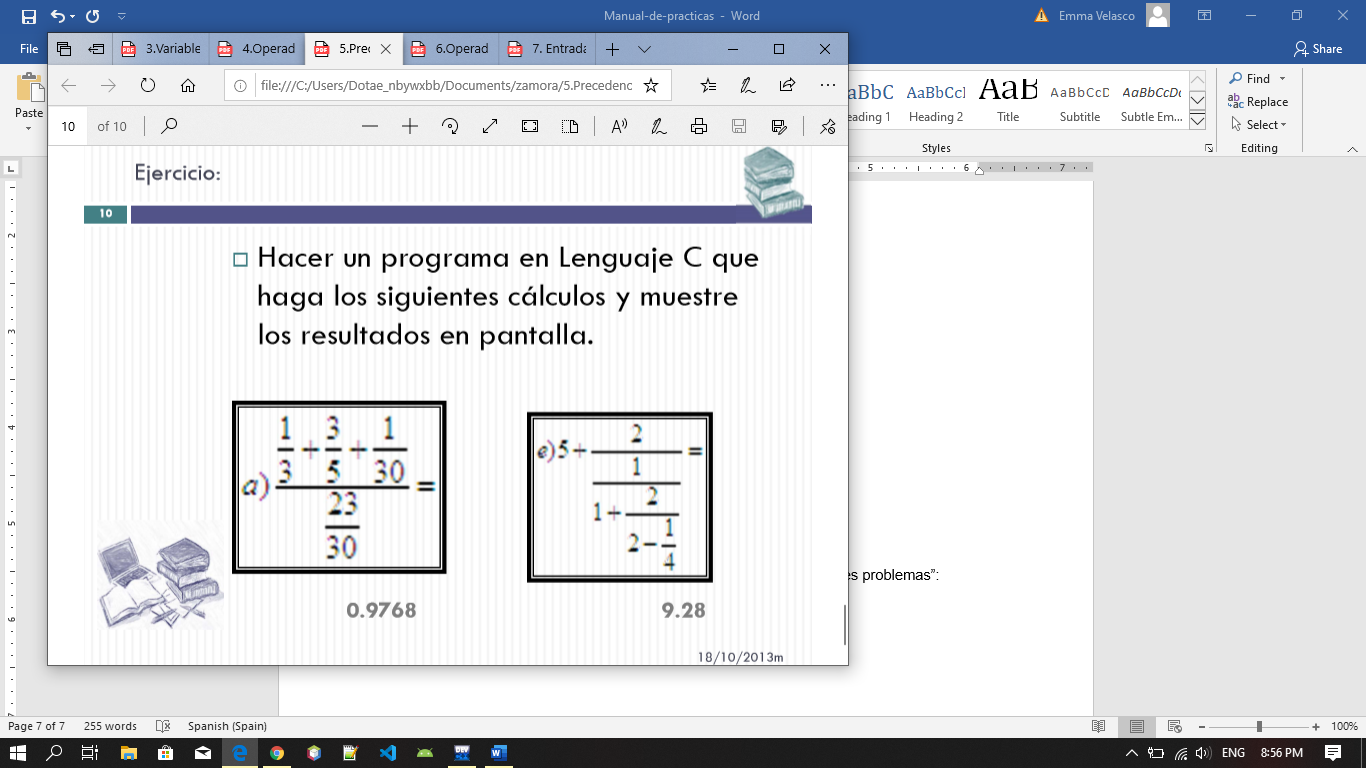
Aquí se muestra en que orden se están ejecutando, que es diferente a como los hemos escrito:



Ejemplo 2 “Precedencia de operadores”:



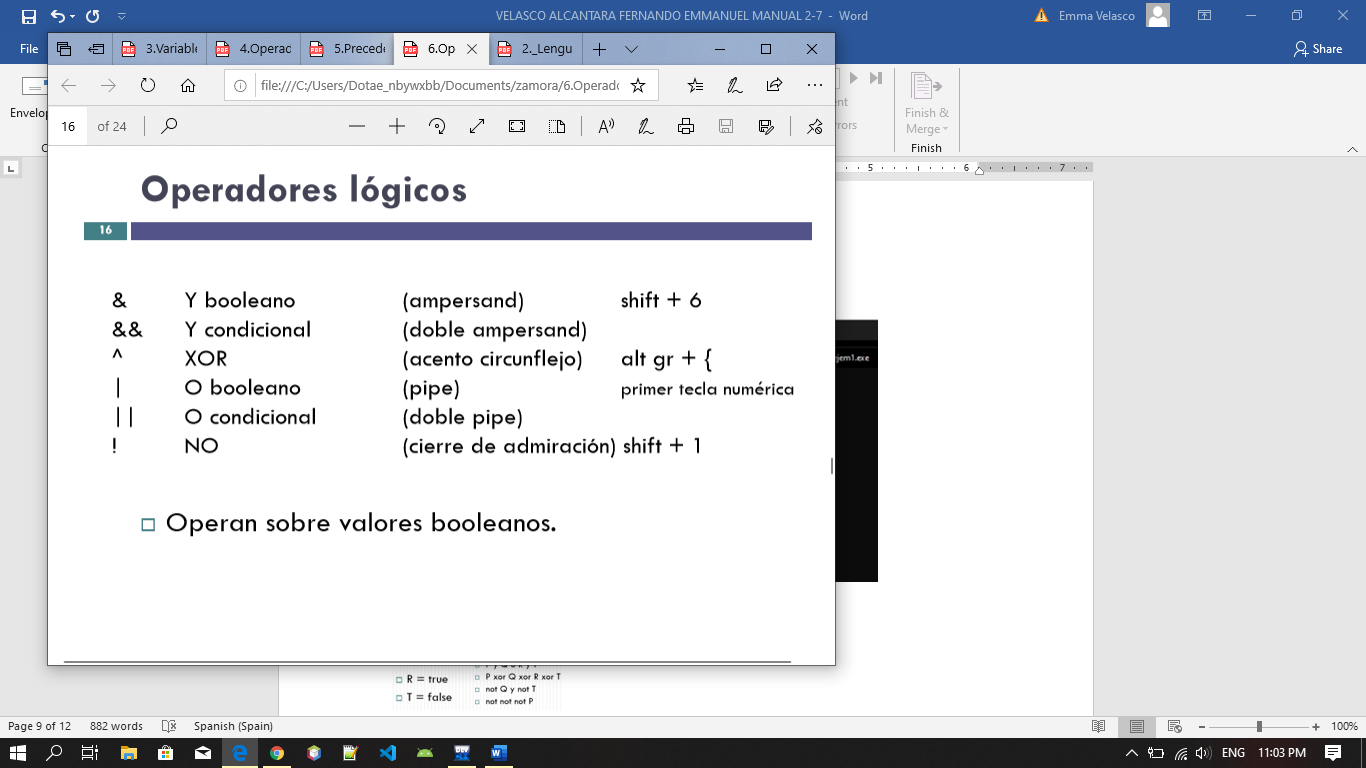
Ejercicio 1 “Hacer un programa que muestre la solución a los siguientes problemas”:





OPERADORES LOGICOS Y DE RELACION:

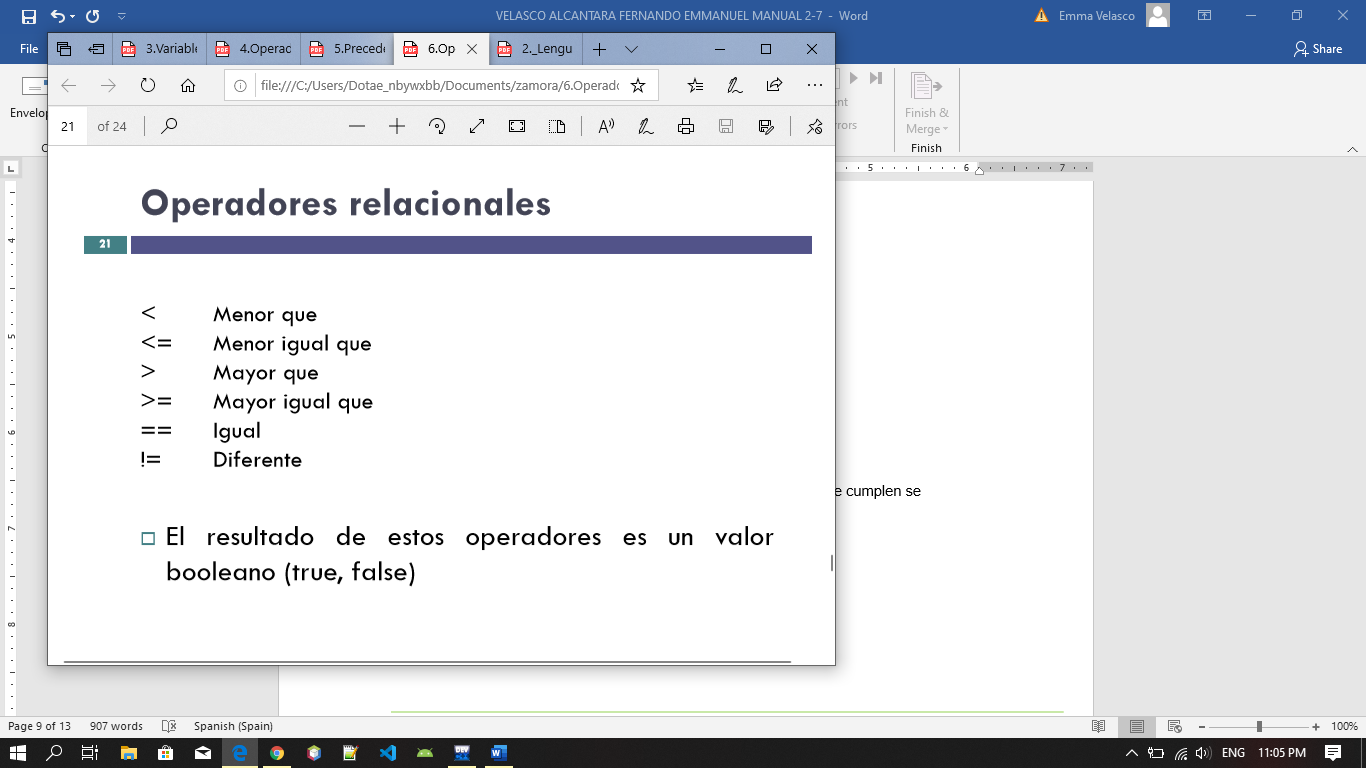
Operadores lógicos:





Estos operadores funcionan mediante condiciones, es decir que si se cumplen se obtendrá un resultado de tipo booleano (true o false).

Operadores racionales:



Estos operadores realizan comparaciones y de igual manera devuelven valores booleanos.

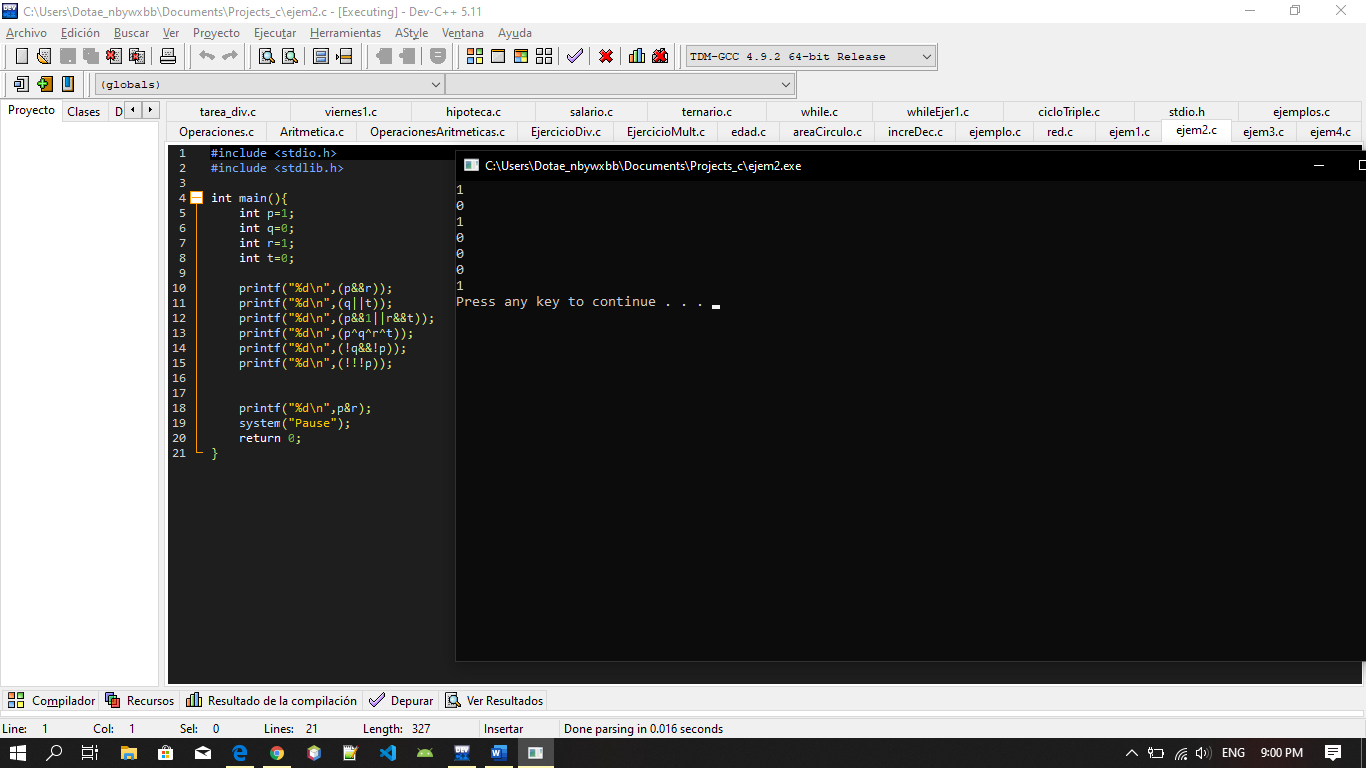
Ejemplo 1 “Operadores lógicos”:



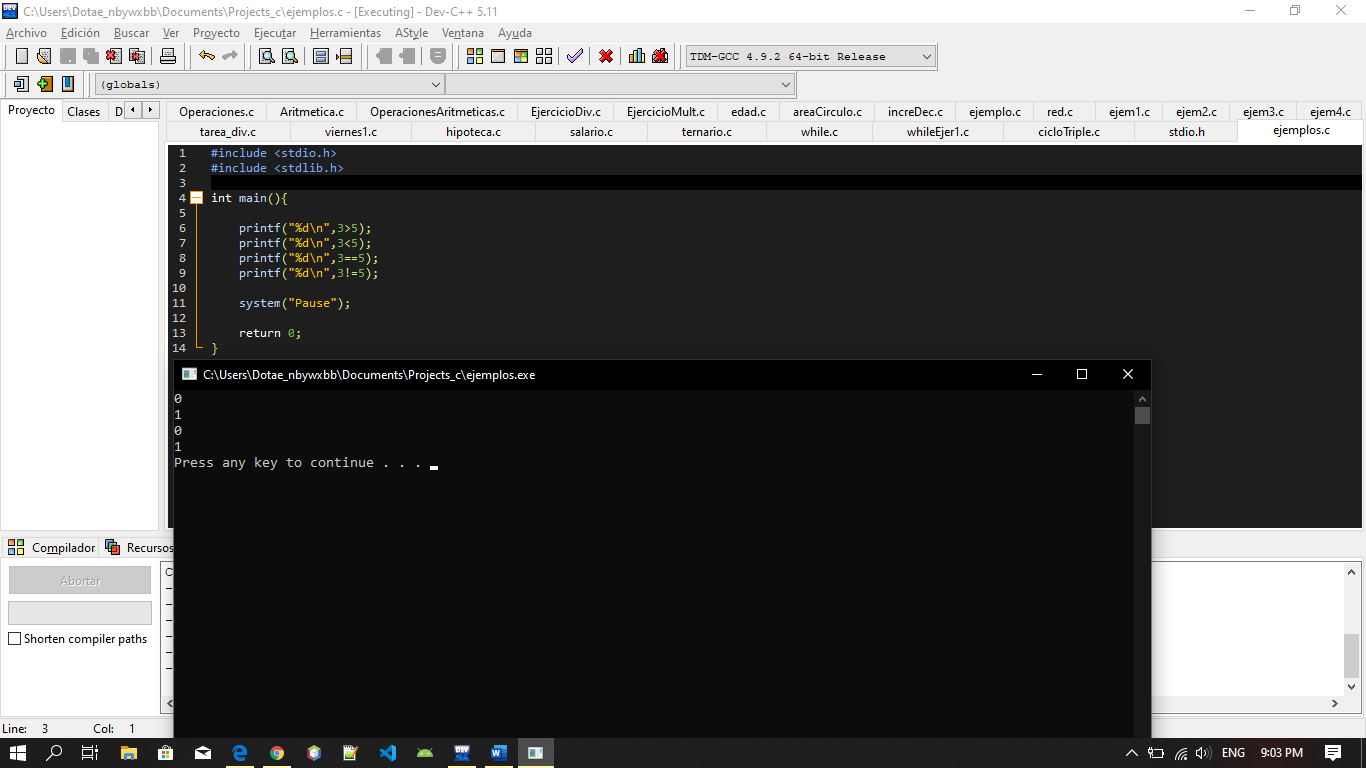
En las operaciones se están haciendo uso de estos operadores, podemos ver que devuelve los valores en ceros y unos (0 -1) esto es de acuerdo a el resultado de la operación o condición.

Ejercicio 1 “Con uso de operadores lógicos imprimir lo siguiente”:

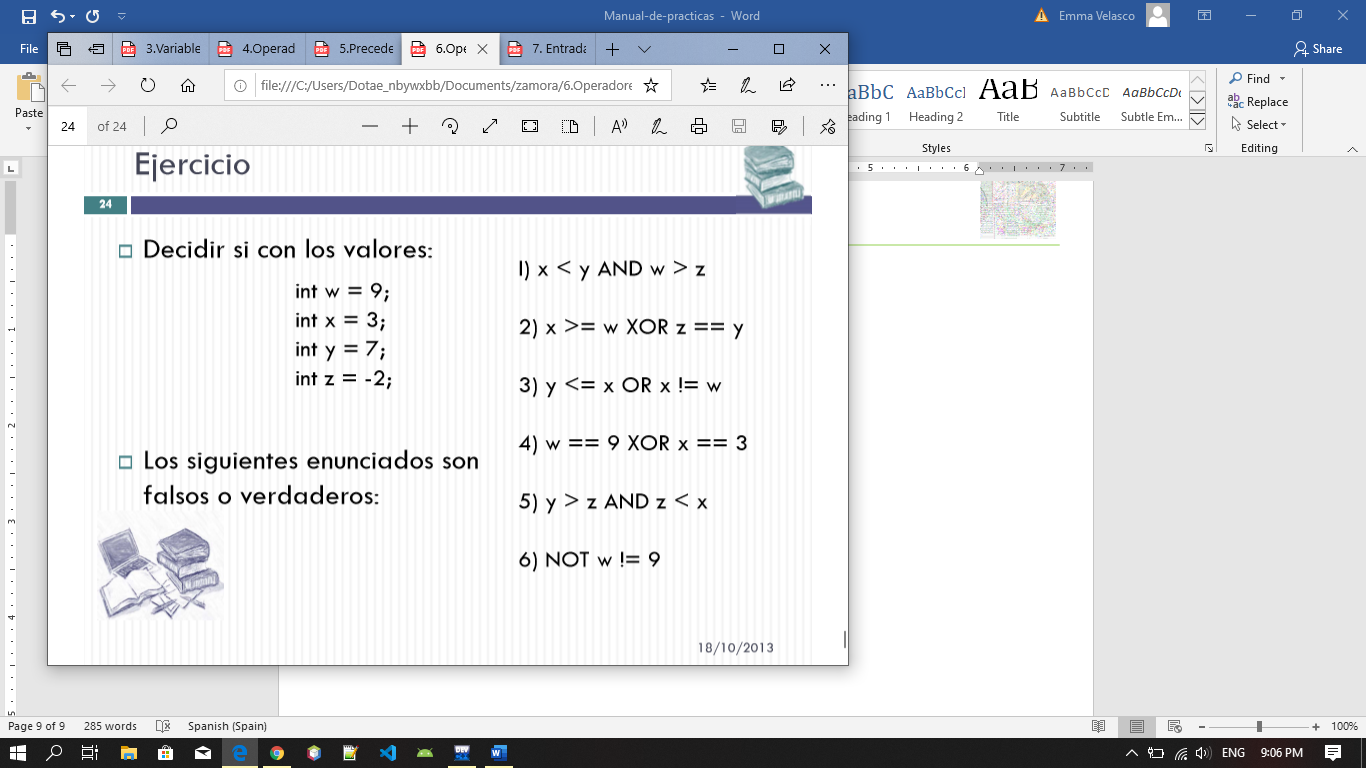
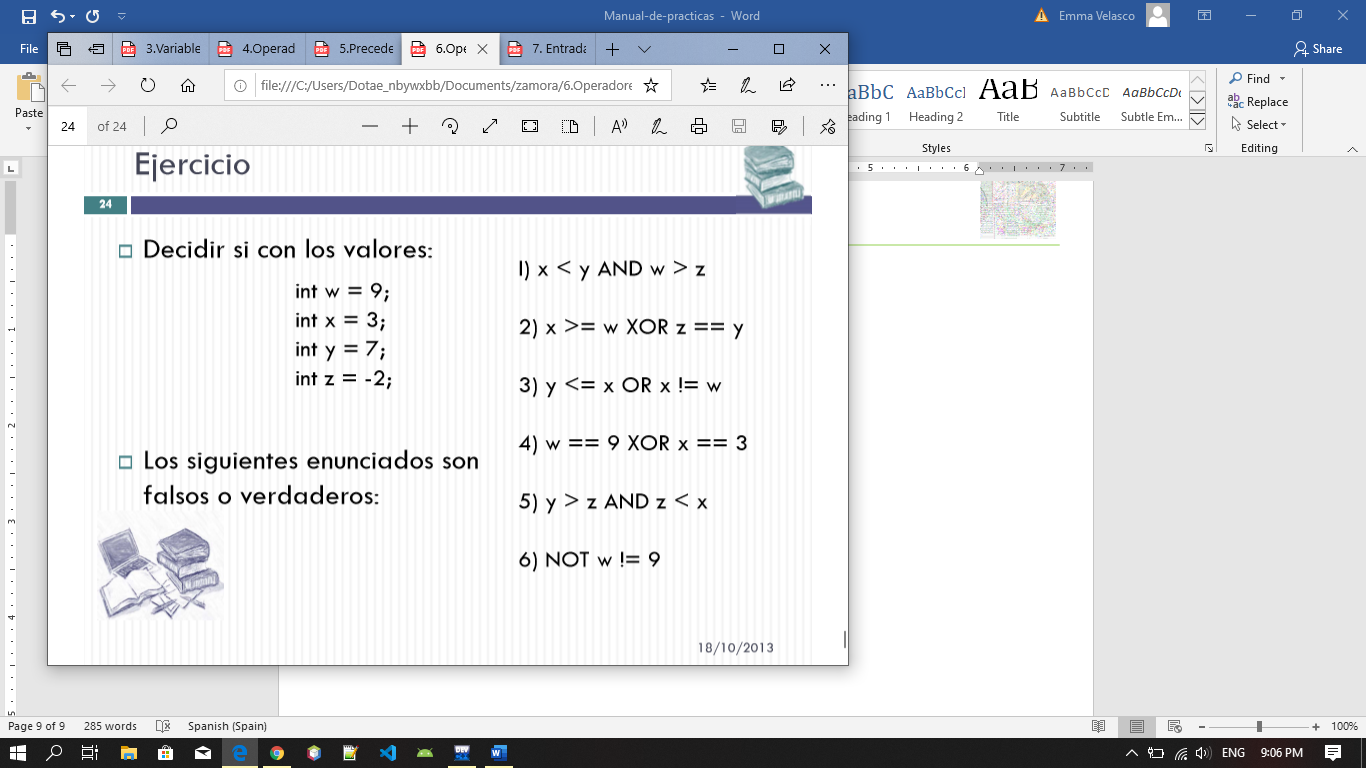


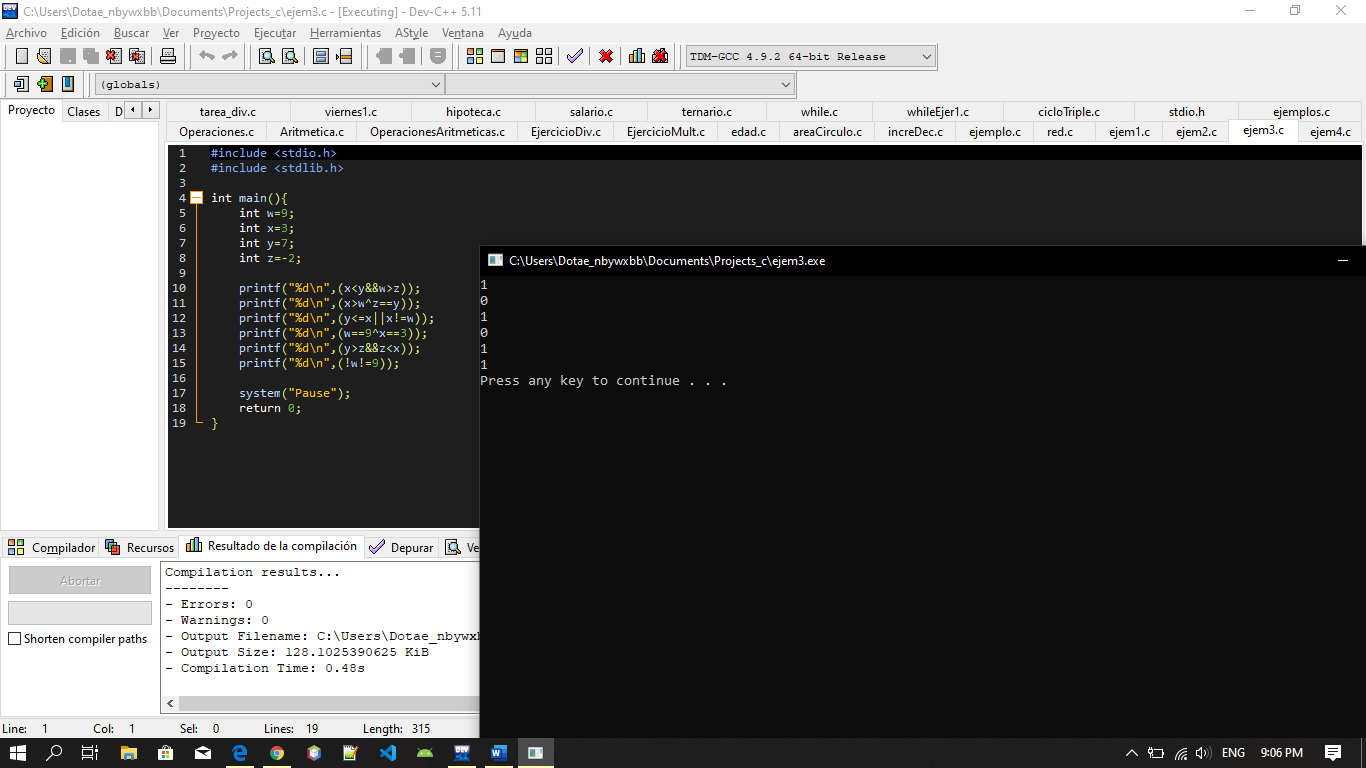


Ejemplo 2 “Operadores de comparación”:



Ejercicio 2 “Decidir con los valores”:





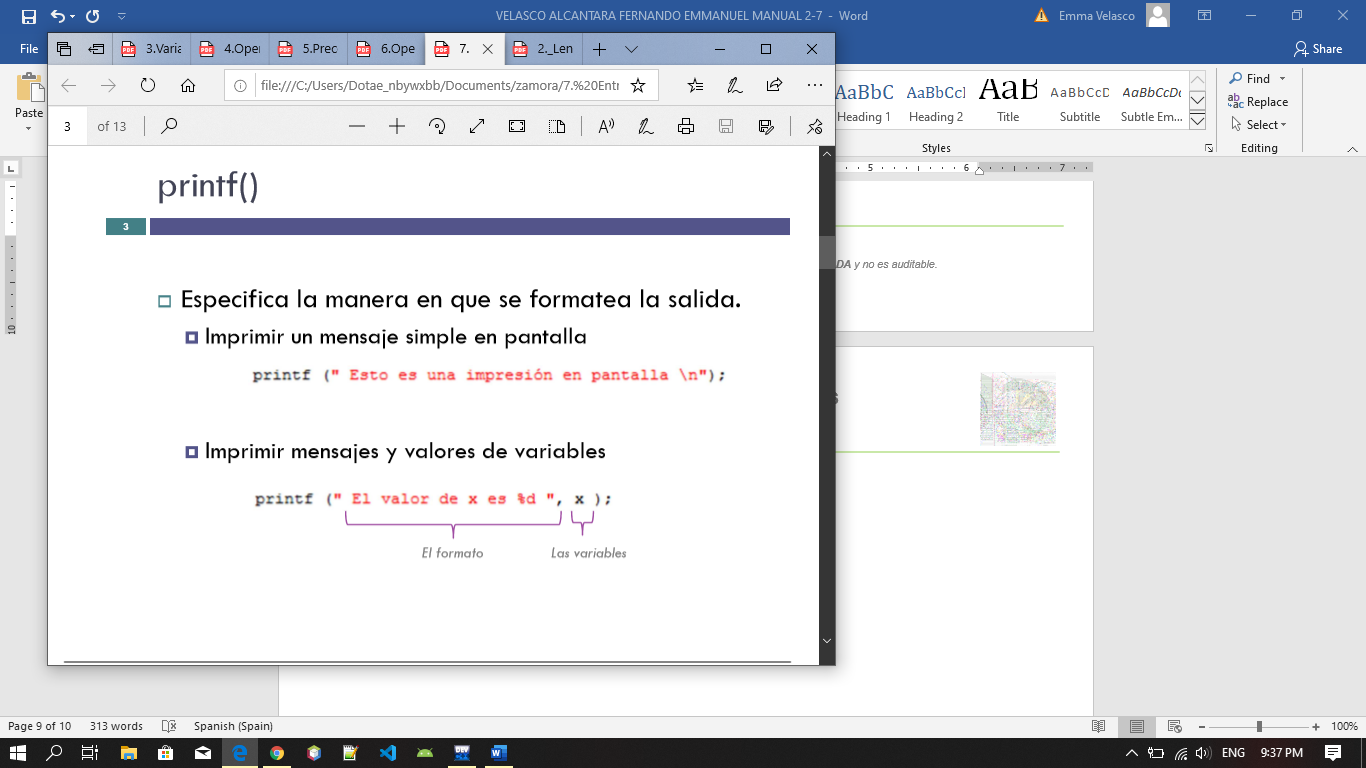
ENTRADA Y SALIDA DE DATOS



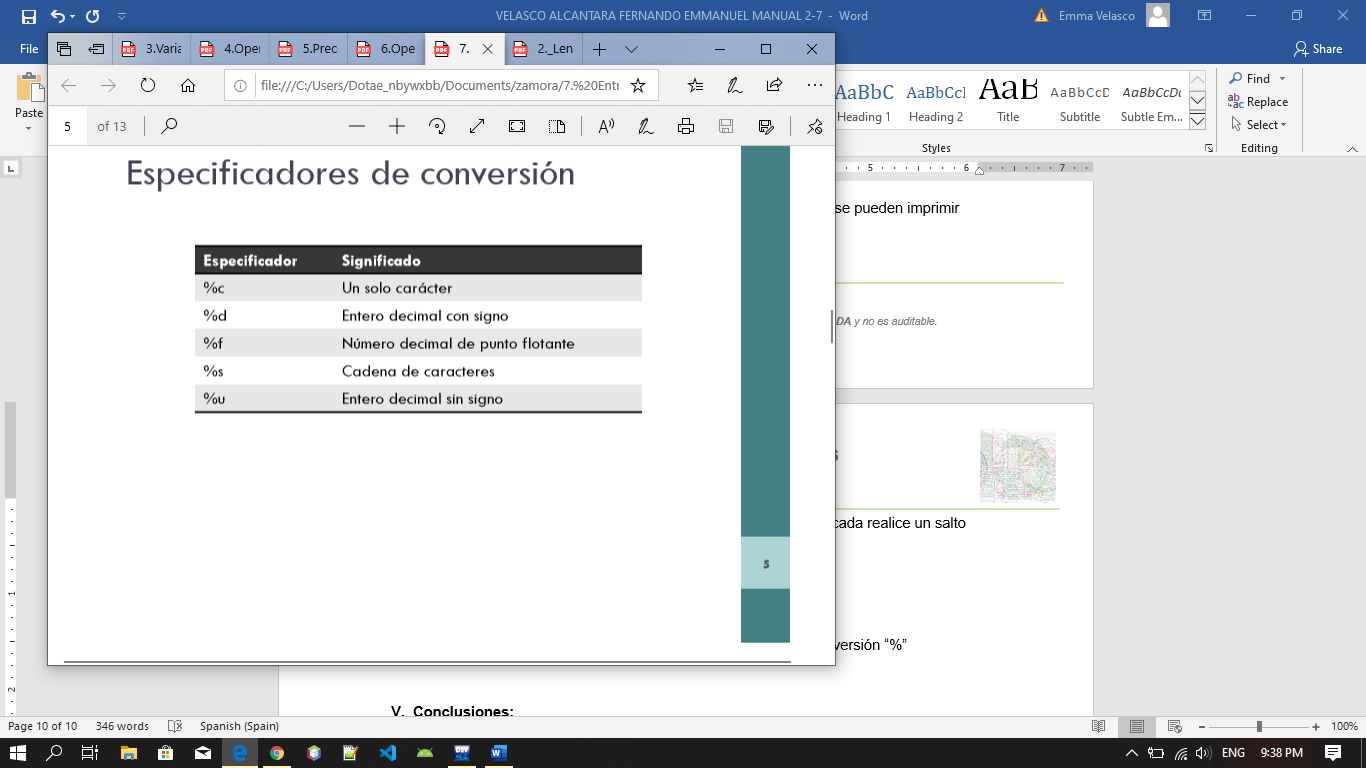
Este valor especifica la manera en que se formatea la salida, con el se pueden imprimir tanto mensaje de texto o valores numéricos.



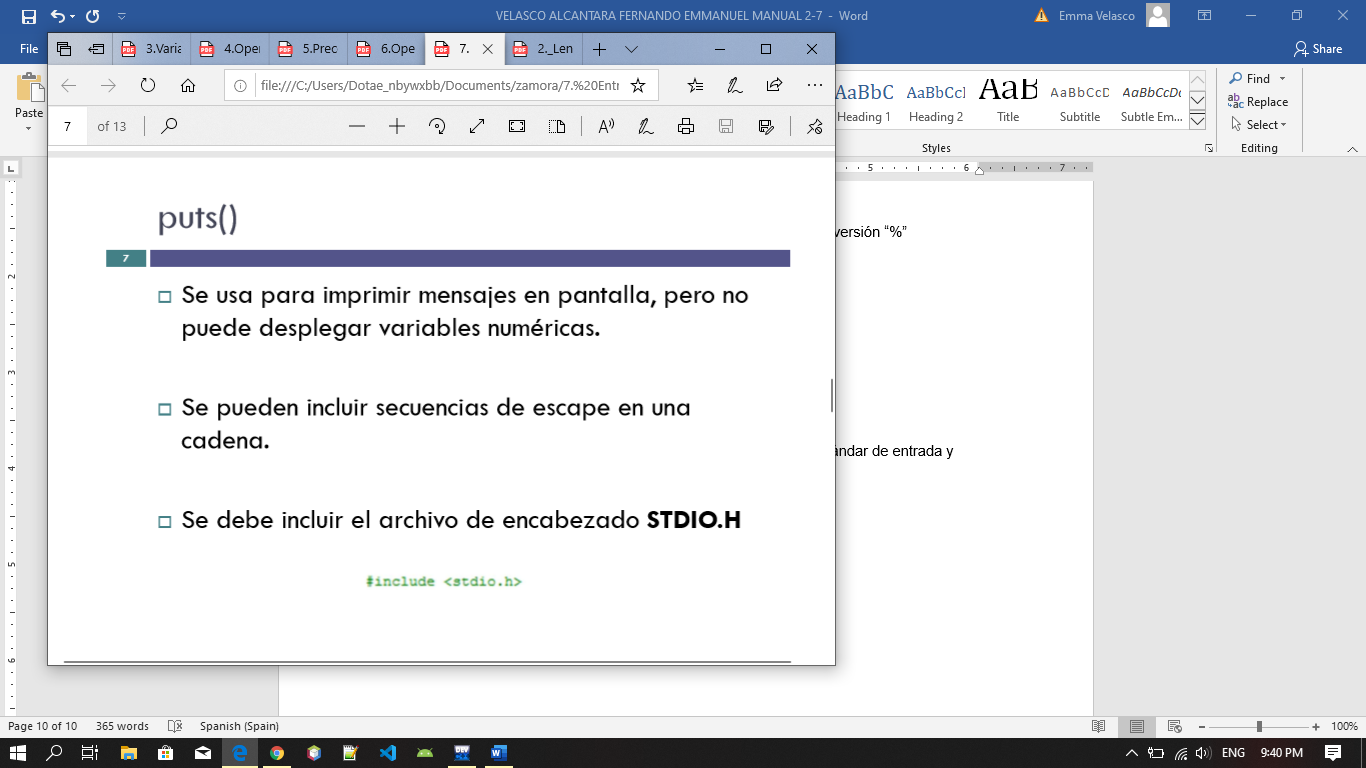
Se le agrega “\n” para que al final del mensaje o en la parte especificada realice un salto de línea



Para imprimir el valor numérico se le agrega el especificador de conversión “%”

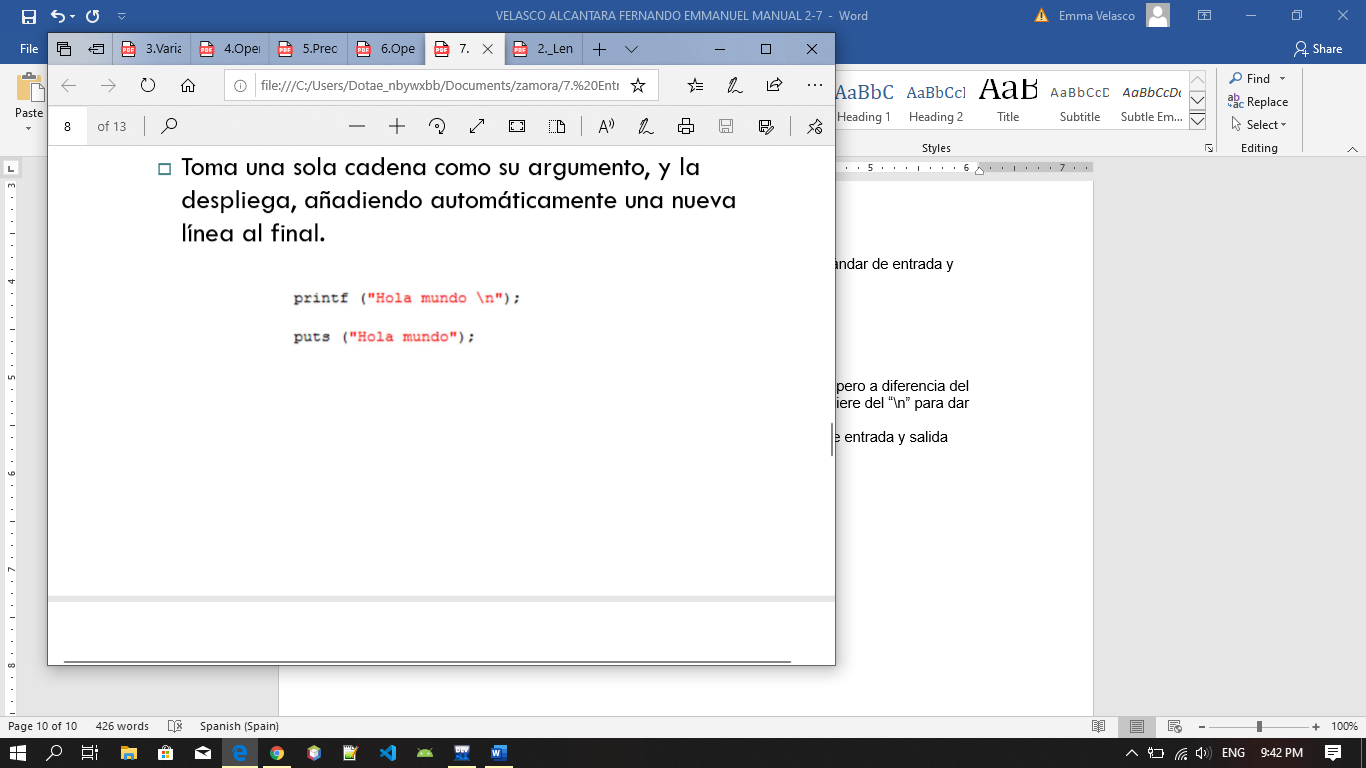


Para poder usar la función “printf” se debe incluir en el código el estándar de entrada y salida “stdio.h”.

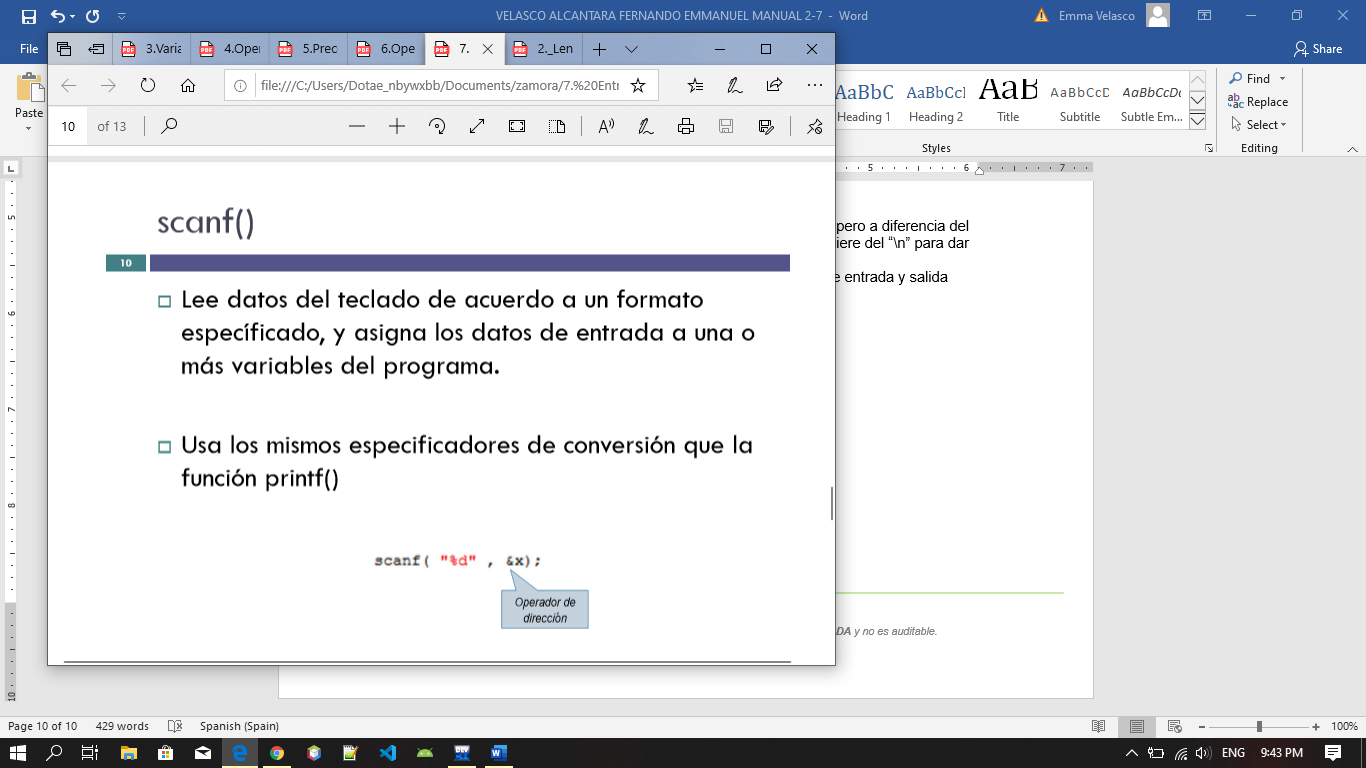


Se utiliza para mostrar mensajes de la misma manera que el “printf” pero a diferencia del mismo, este no puede imprimir valores numéricos y además no requiere del “\n” para dar un salto de línea, ya que este lo hace de manera automática

Para poder hacer uso de el también se debe de incluir el estándar de entrada y salida “stdio.h”.

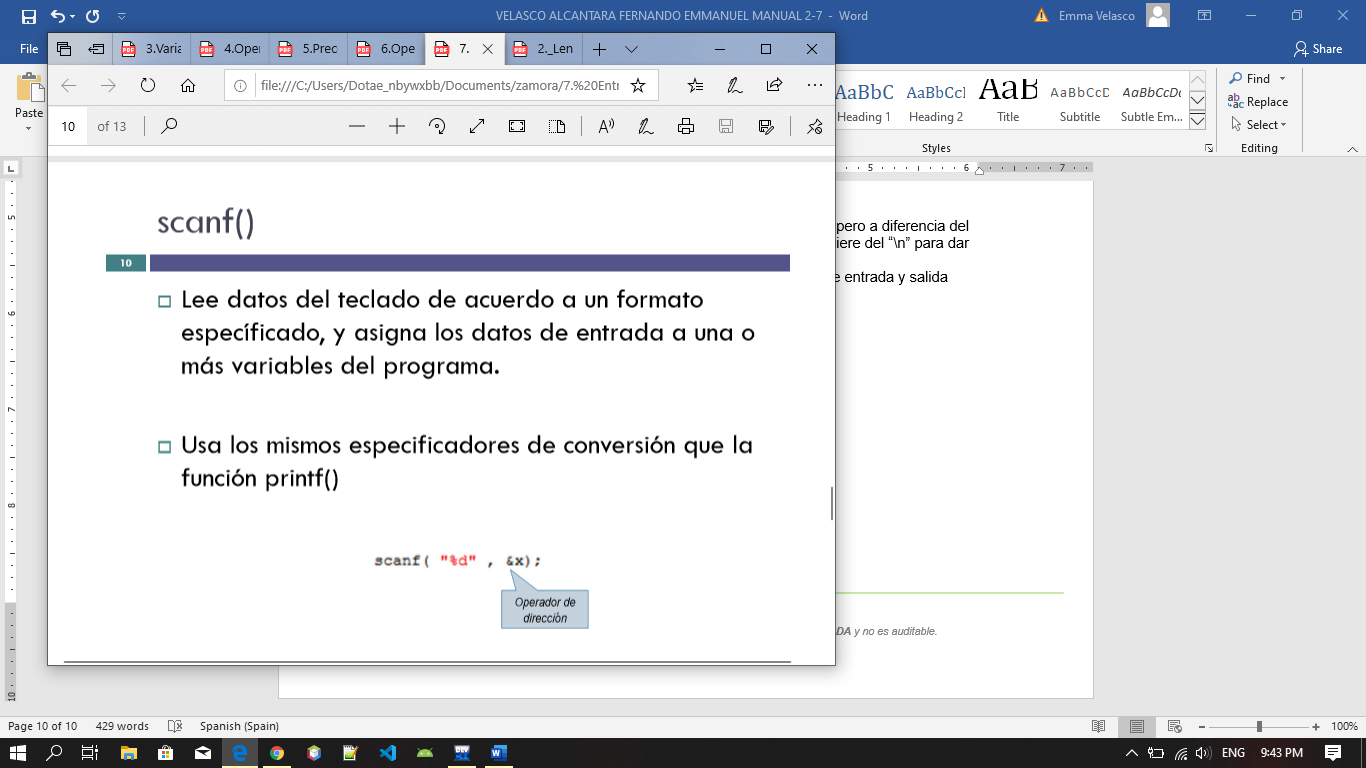


Entrada de datos:



Lee datos ingresados desde teclado de acuerdo al especificador de conversión que se le indique y los guarda en una variable.

Debe de tener un operador de dirección para que este se guarde en la memoria:



**V. Conclusiones:**