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

**I. Competencia(s) específica(s):**

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

Aula

**III. Material empleado:**

Laptop

Dev C++

**IV. Desarrollo de la práctica:**

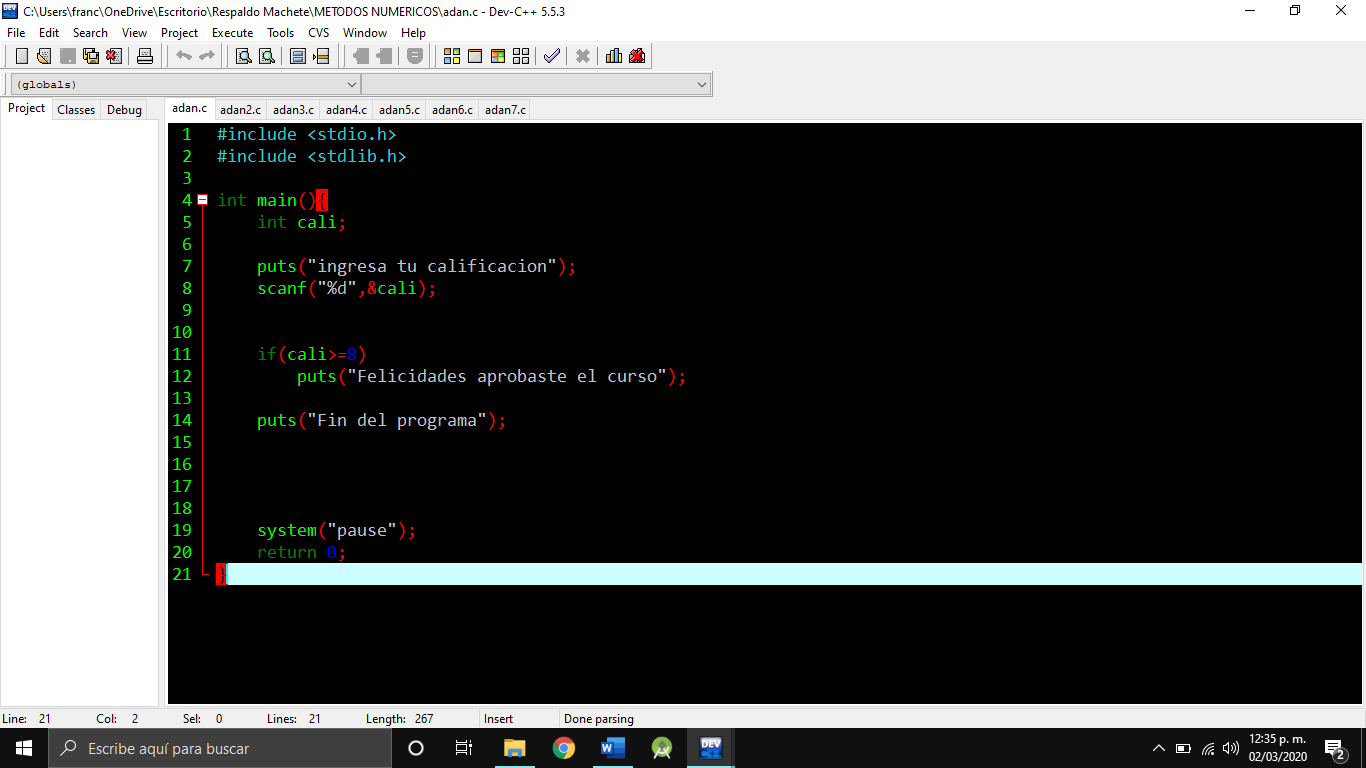
C

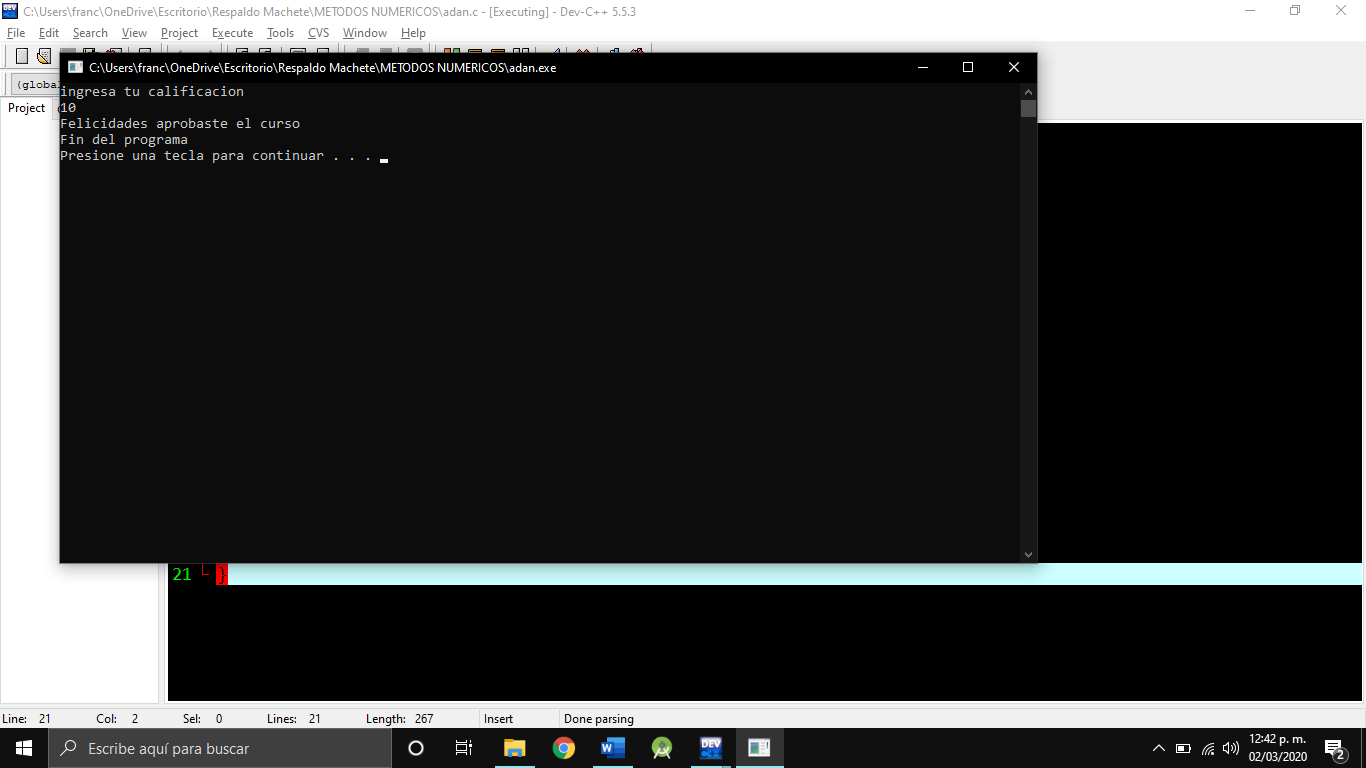
C es un lenguaje de programación estructurado desarrollado a principios de los 70´s. Reemplazando a su predecesor B, es un lenguaje de nivel medio, porque tiene características de un lenguaje de bajo nivel.

Es el padre de c++, java y entre otros.

**If´s**

**Ejercicio 1.** Ingresar la calificación de un alumno y mostrar un mensaje si aprobó el curso



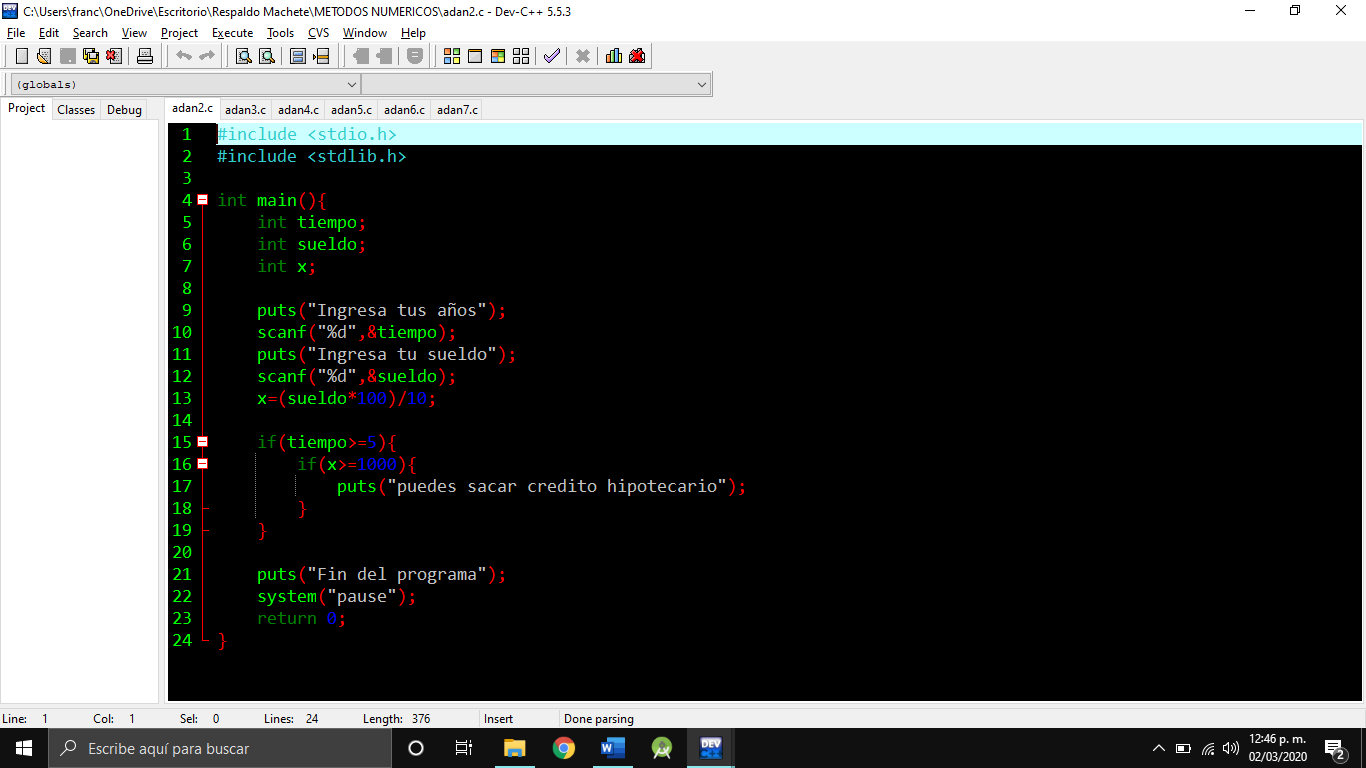


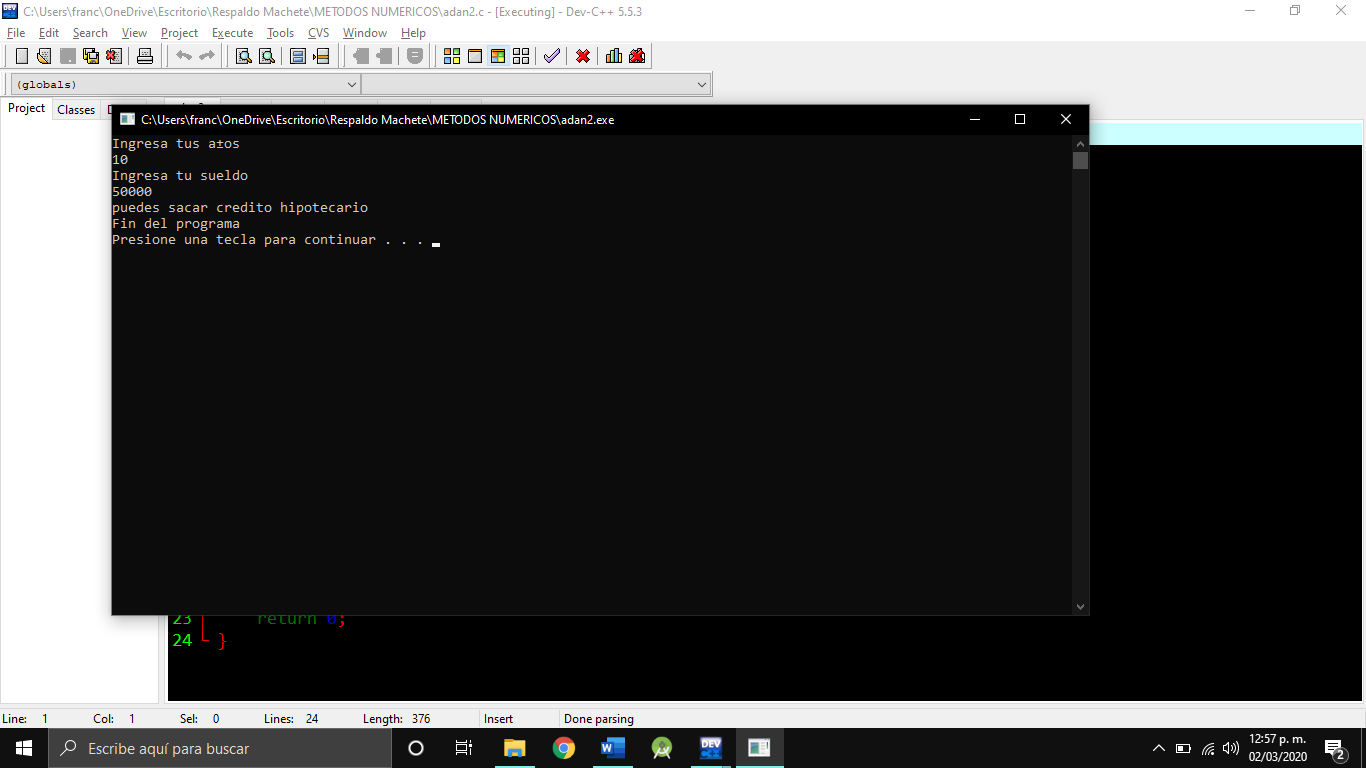
**puts ()** línea de comando que imprime un mensaje en símbolo de sistema junto con un salto de línea, no se puede introducir valores de tipo entero.

**scanf ()** línea de comando que captura el valor introducido en símbolo y que debe de llevar el Amper son para poder guardar en memoria las variables.

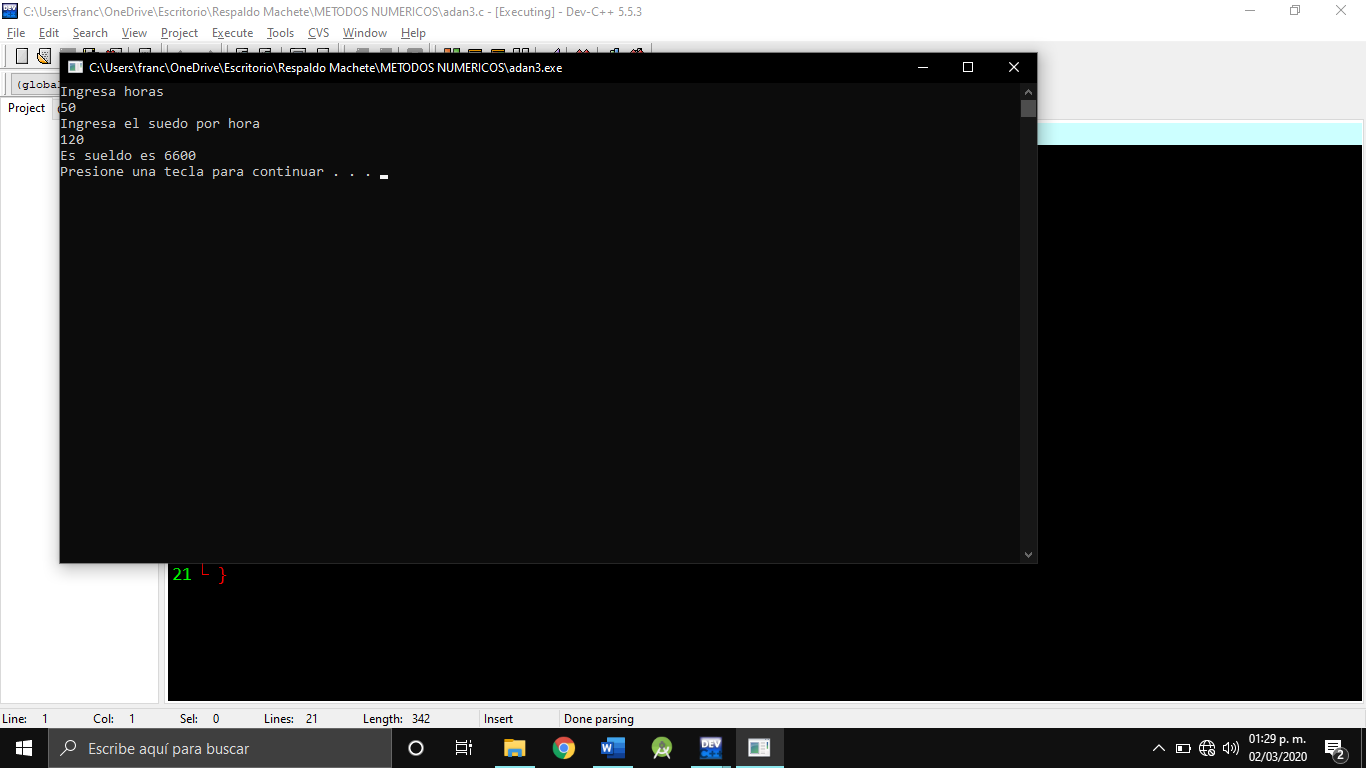
**If ()** Estructura de condición

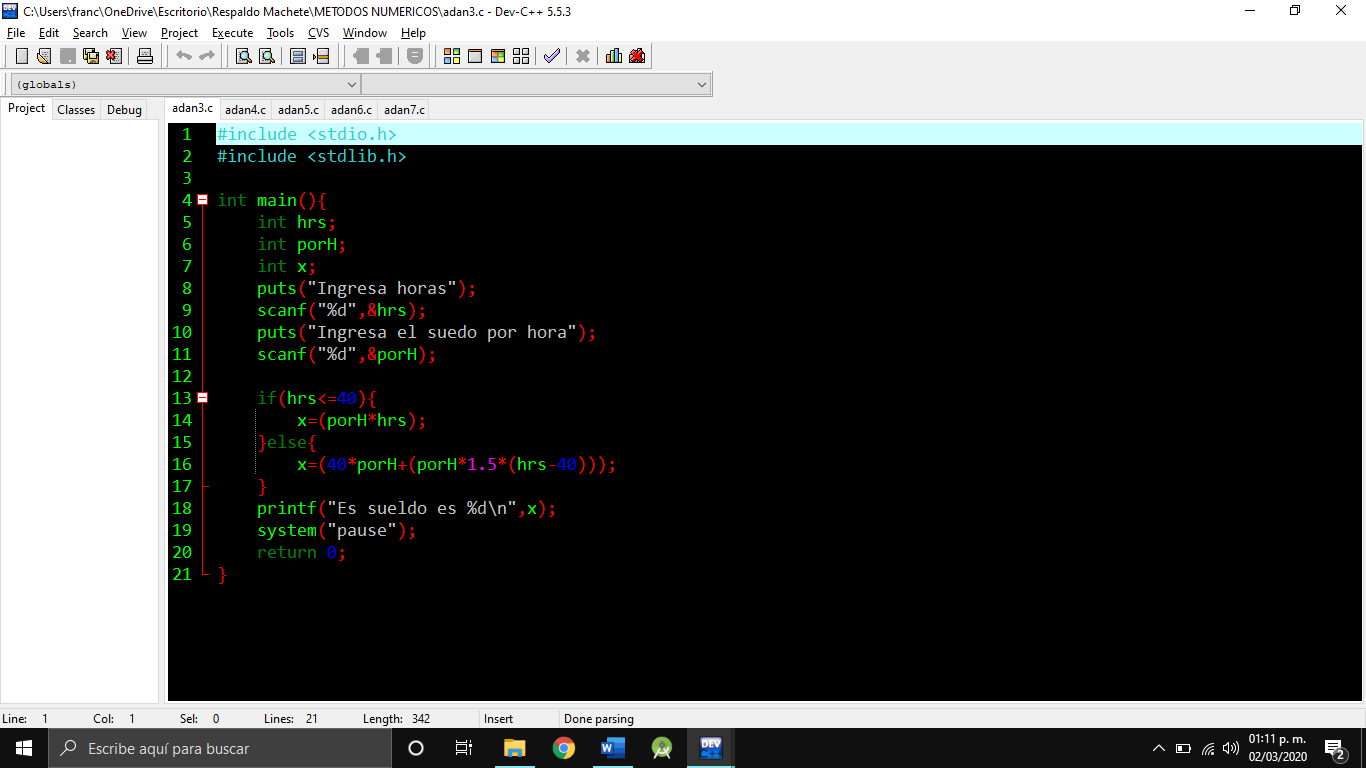
**Ejercicio 2.** Hacer un programa que ingrese un trabajador y años para saber si puede sacar un crédito hipotecario





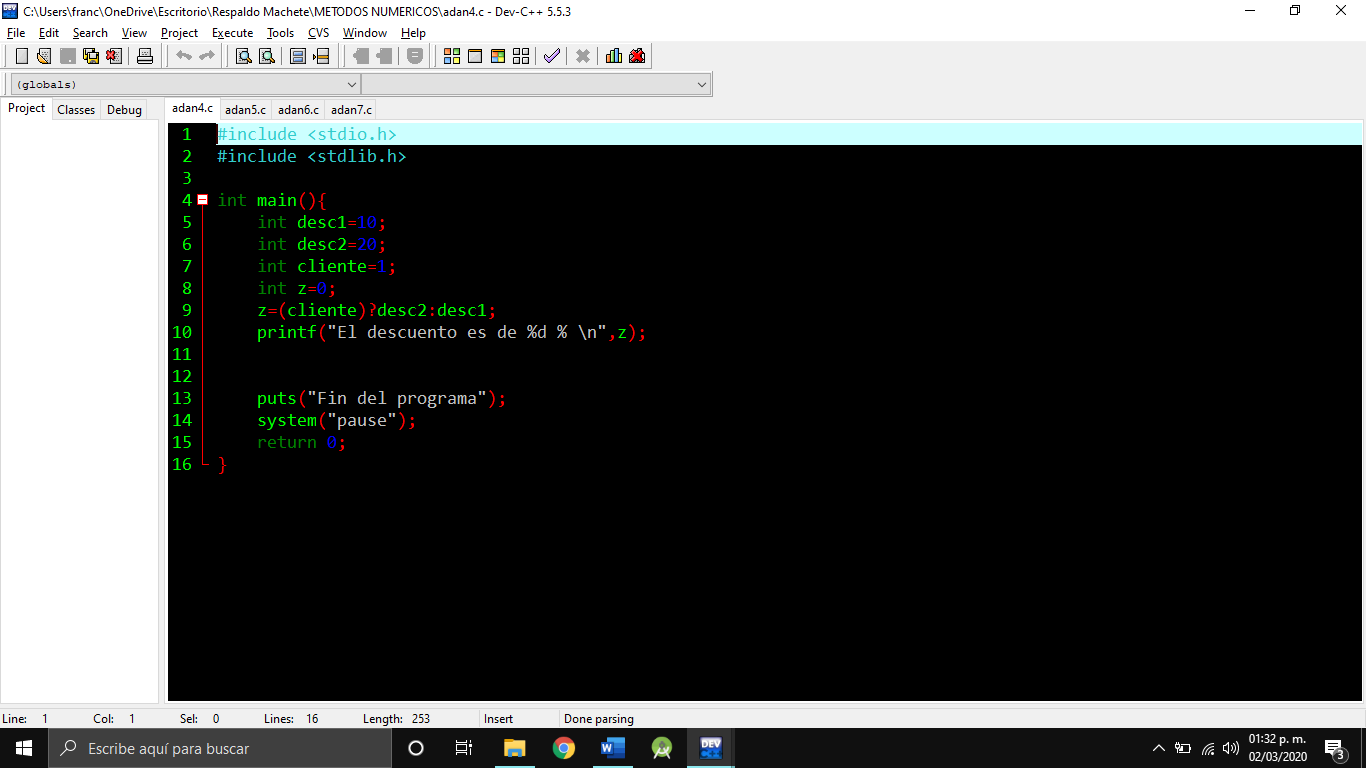
**Ejercicio 3.** Hacer un programa que calcule el sueldo de un trabajador y si trabajo horas extra le pague lo correspondiente sino solo sus horas normales.

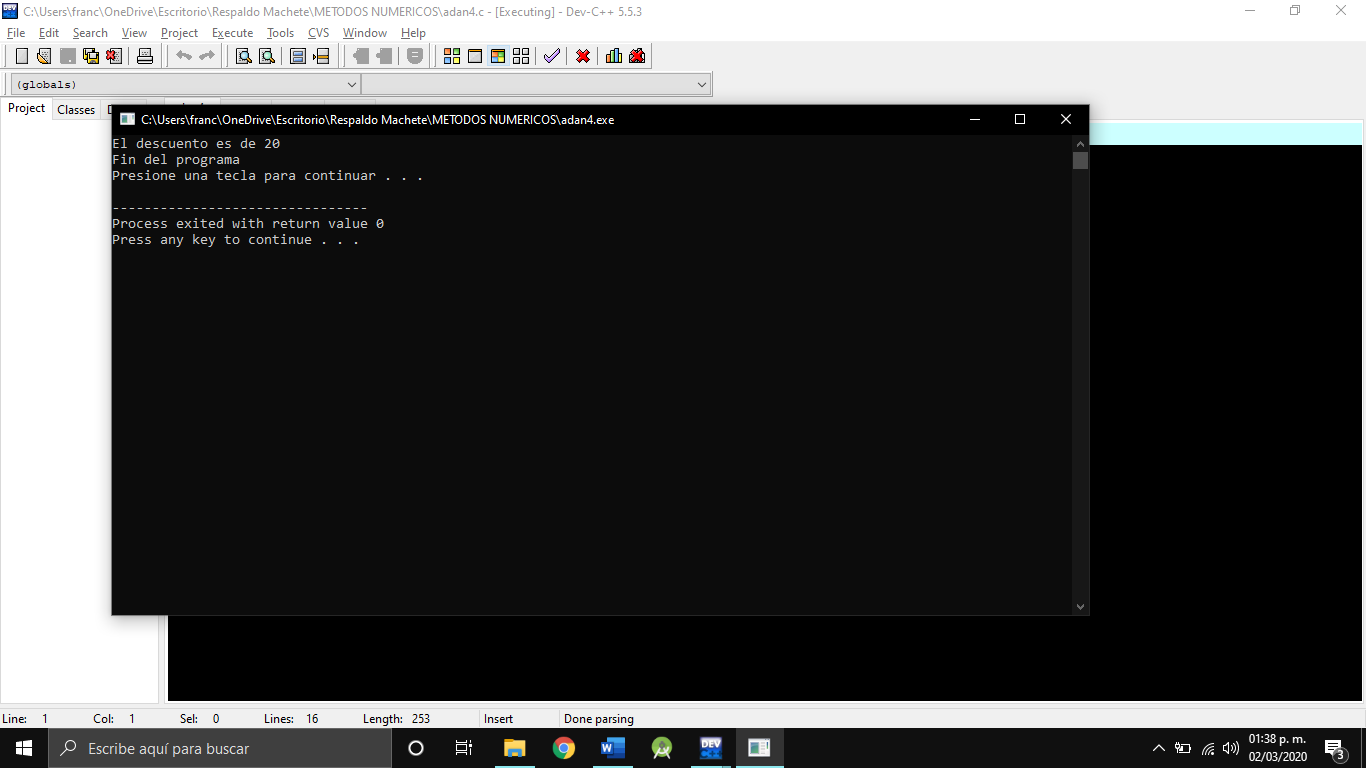




**Ejercicio 4.** En este ejercicio veremos la sintaxis del operador ternario.

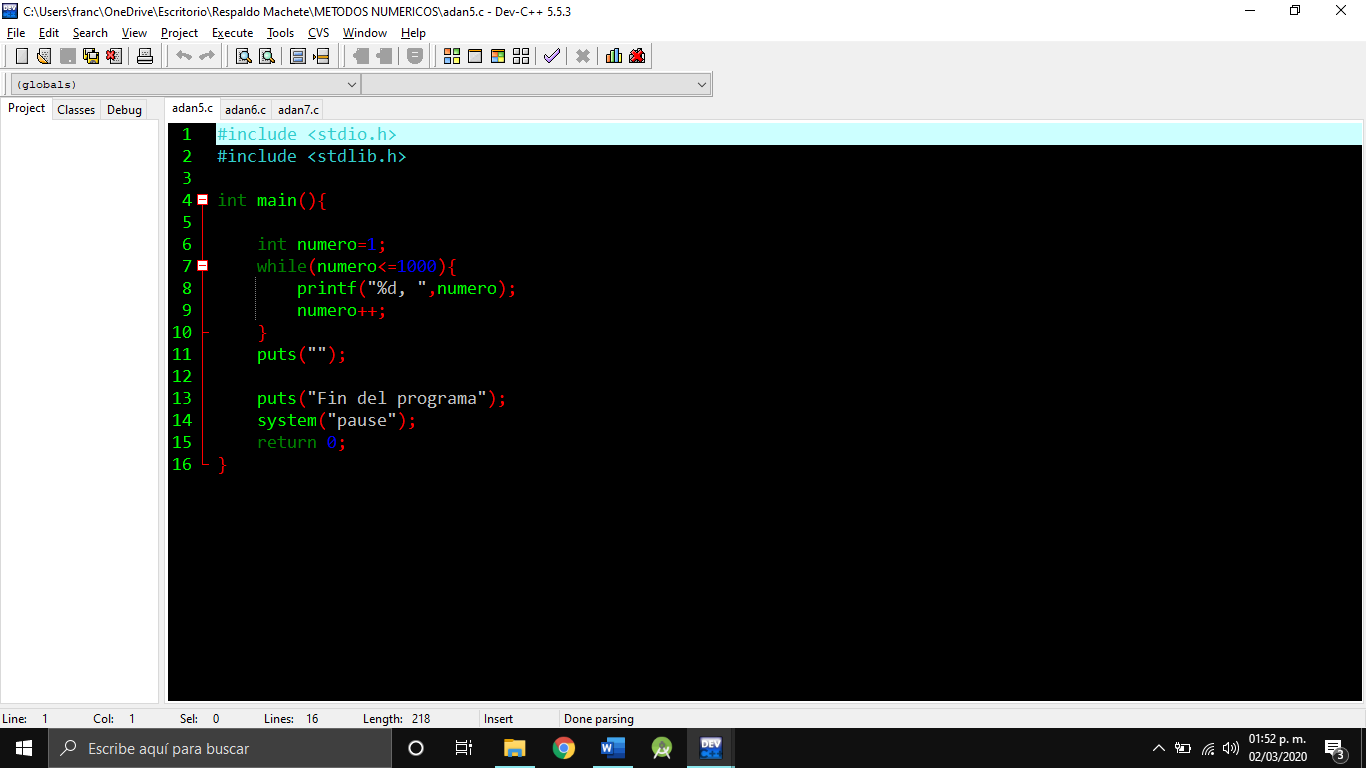
El operador ternario es un if, pero ocupa menos ciclos de reloj que un if

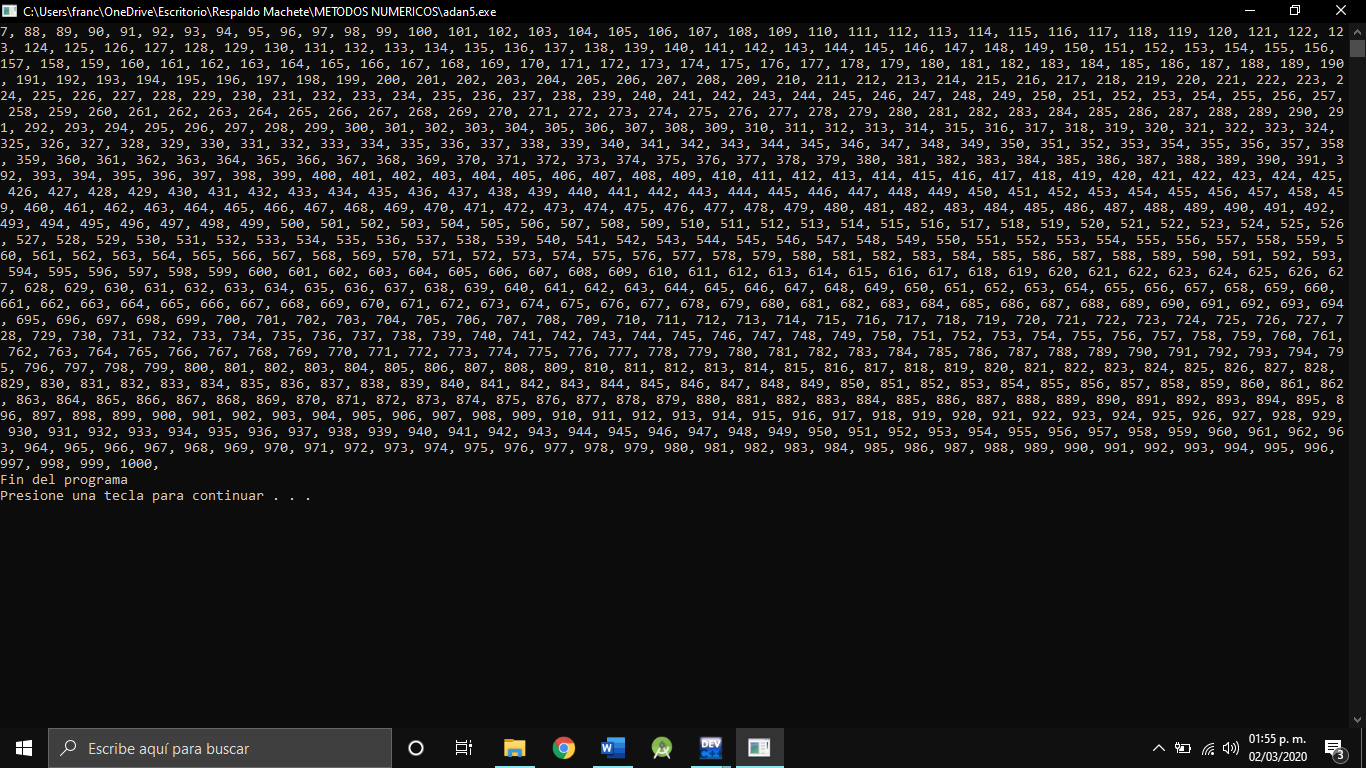




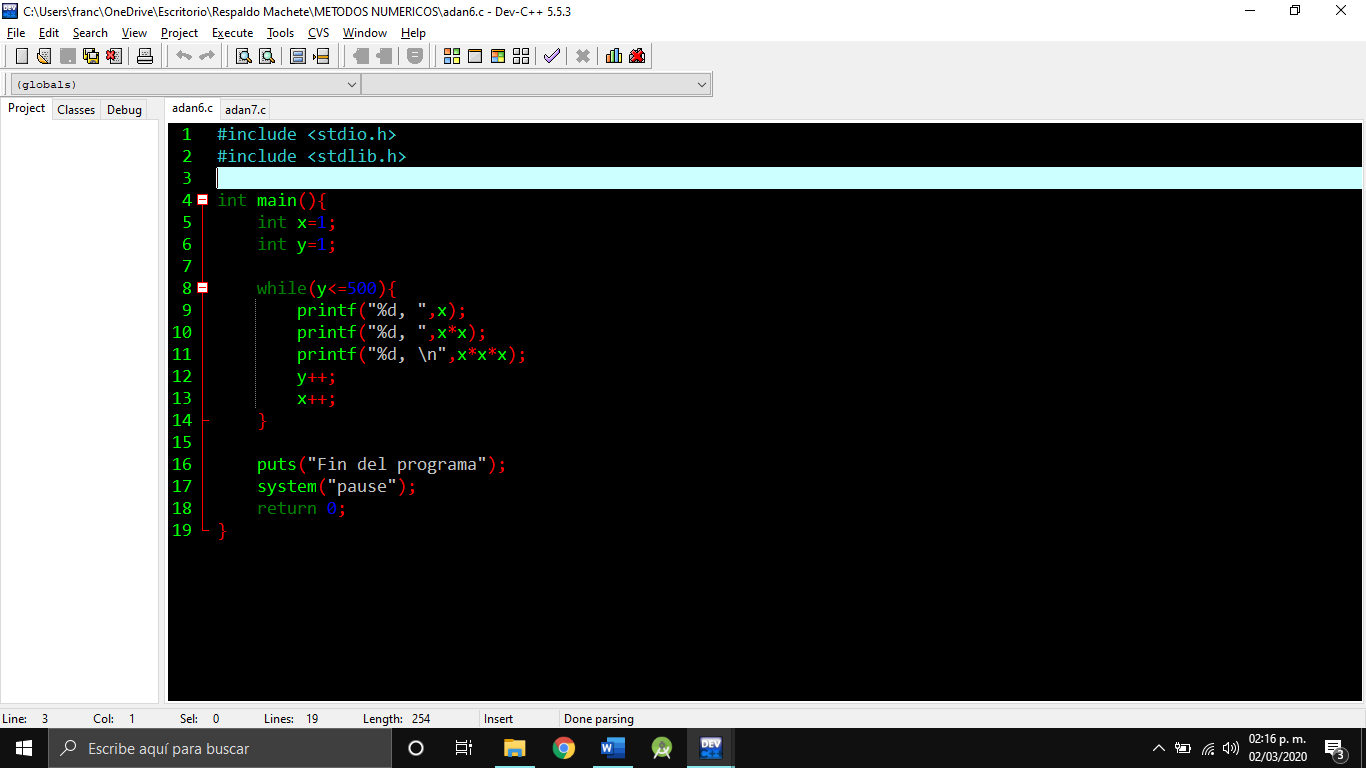
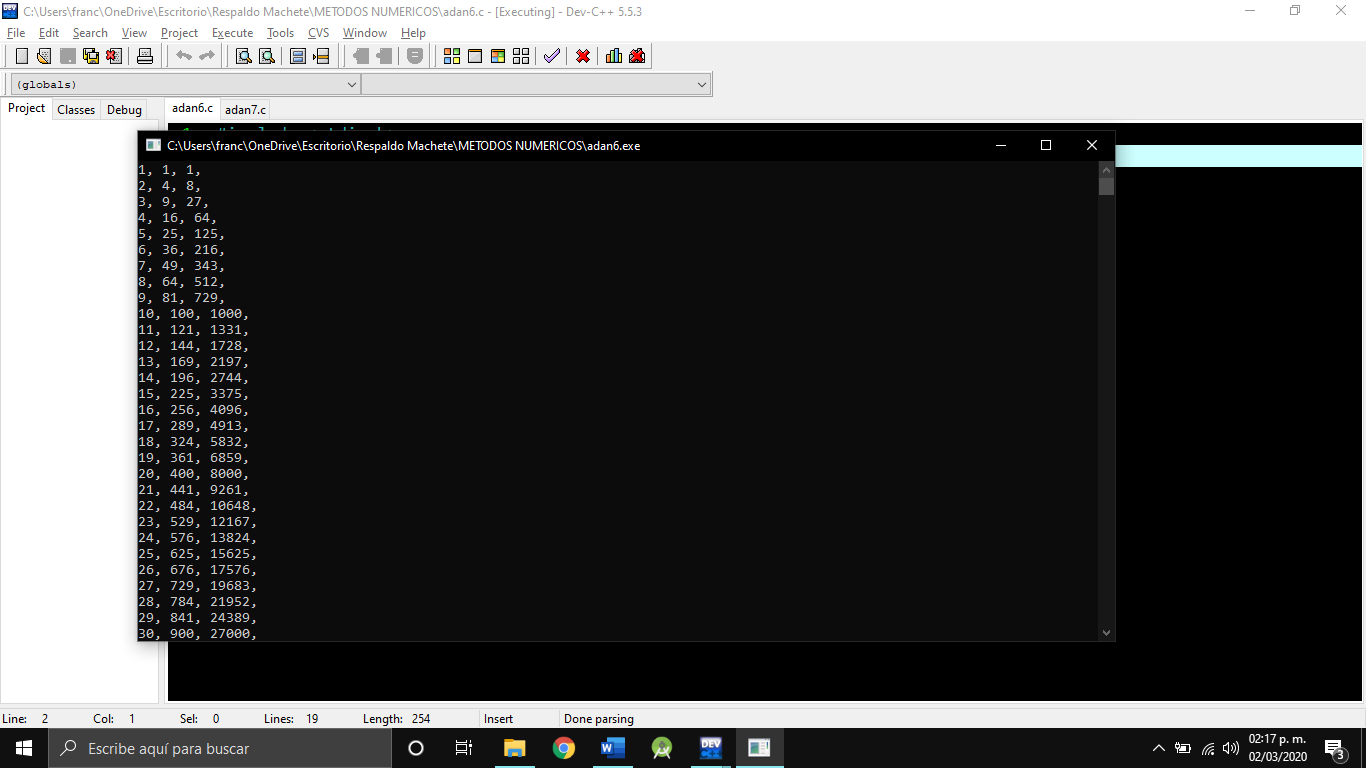
**Ciclos (while, do-while y for)**

**Ejercicio 5.** El primer repetidor es while, la misma sintaxis es como la de java, se debe de tener la variable inicializada, y la condición dentro del while y el incremento dentro de la estructura.



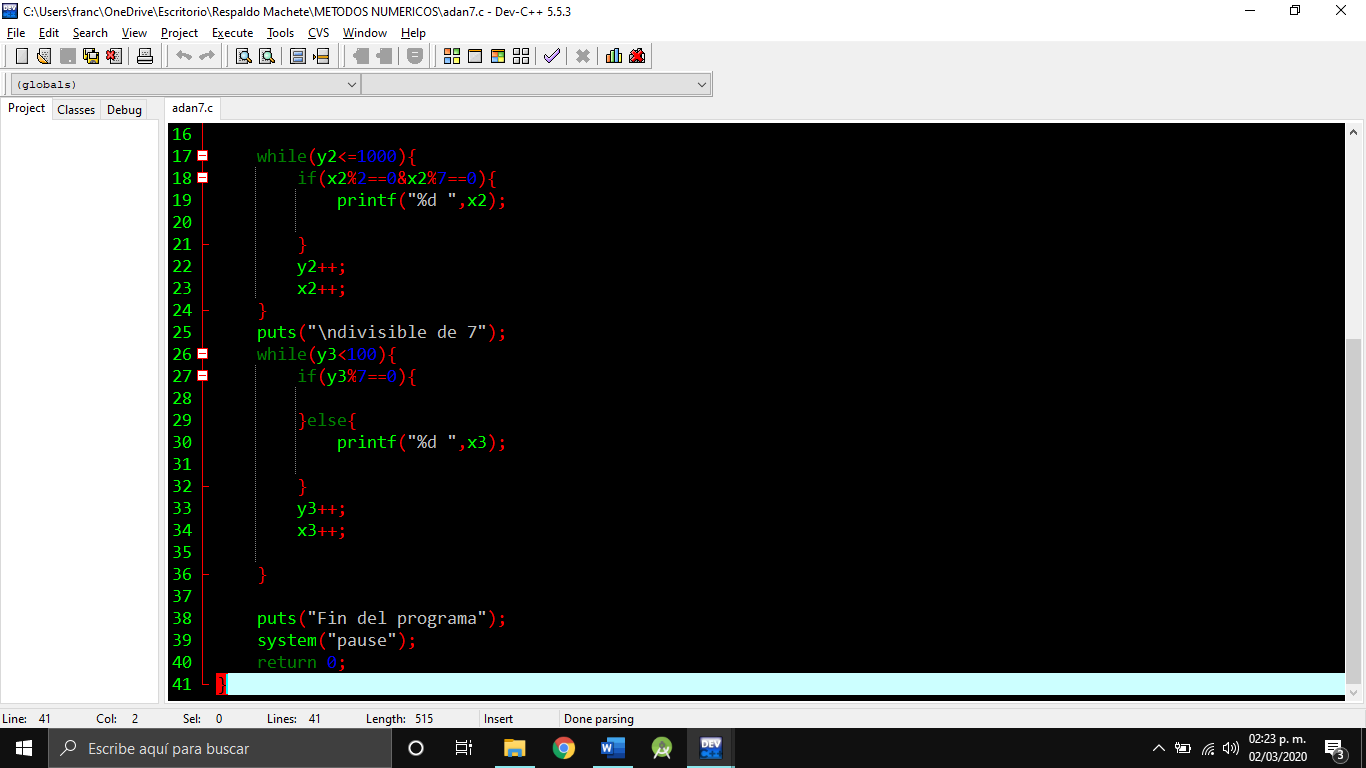
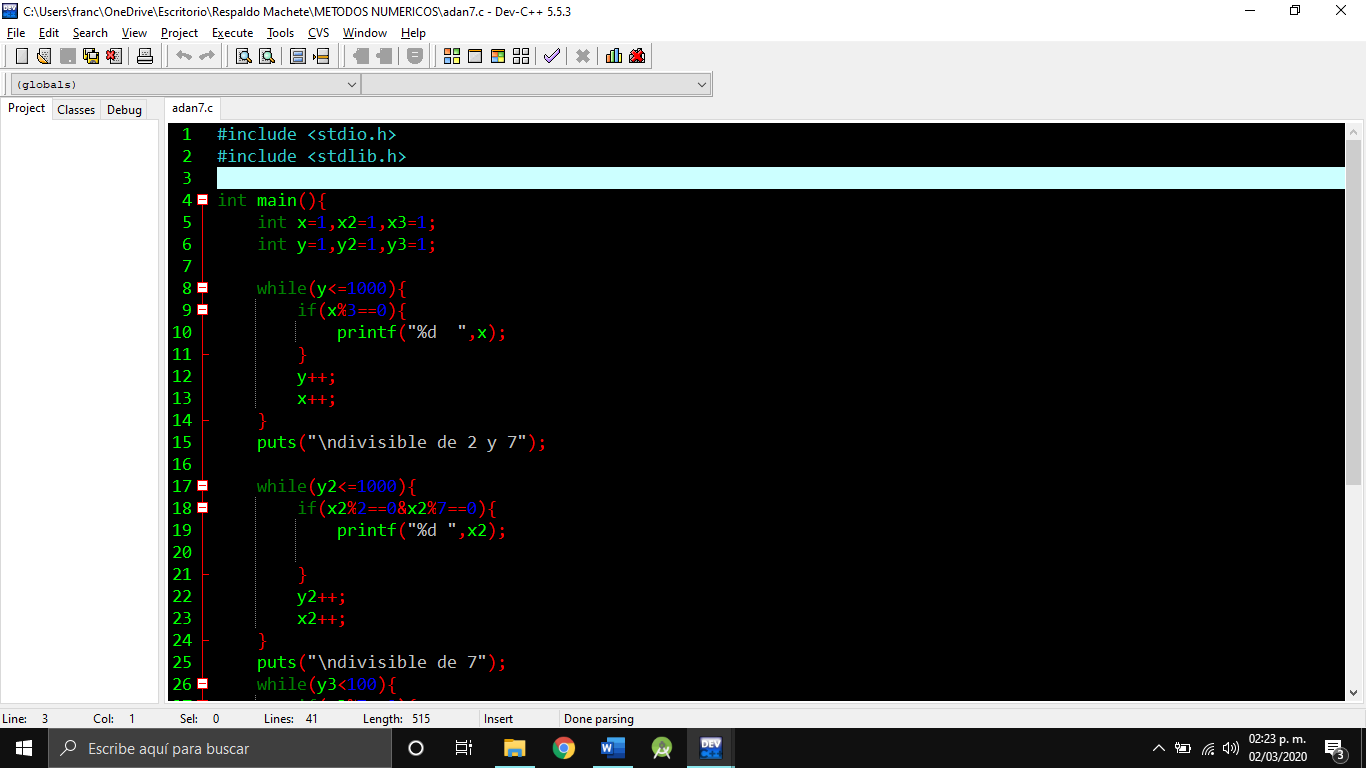


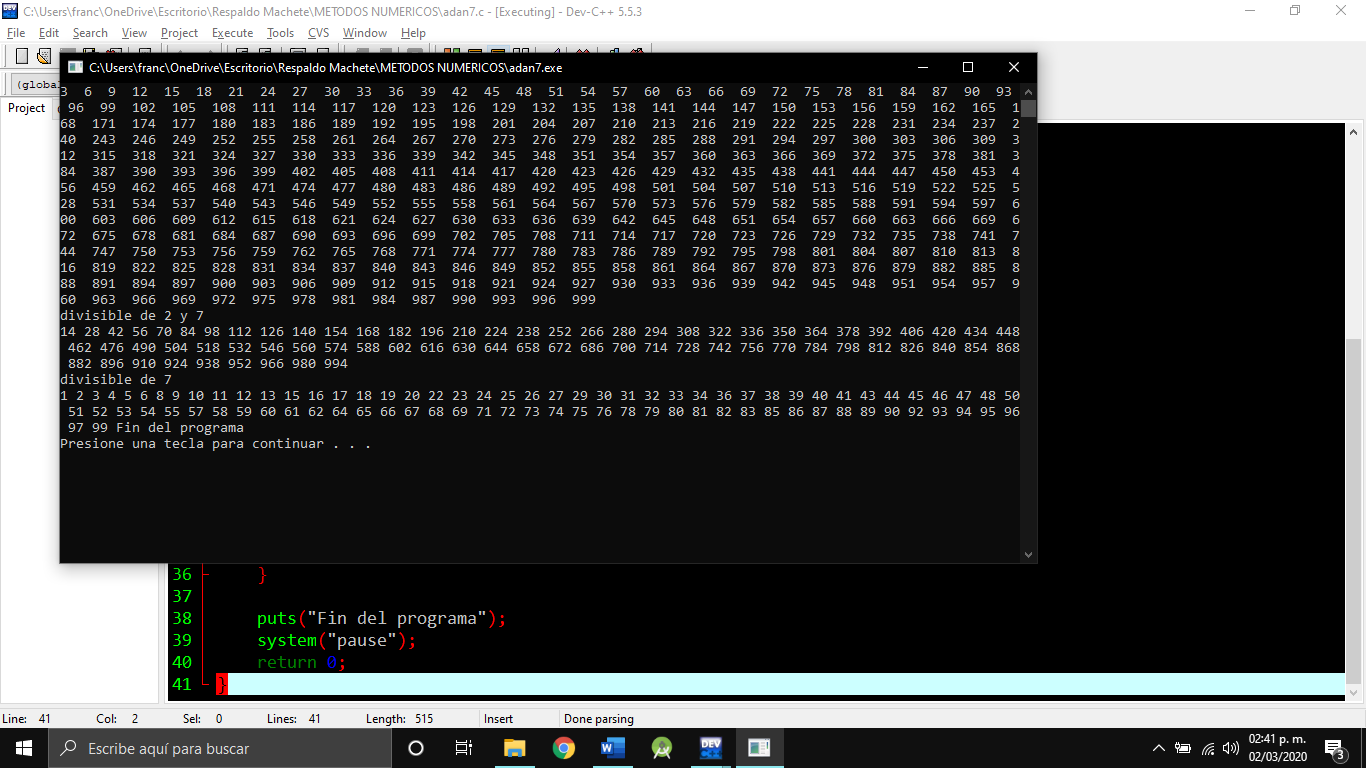
**Ejercicio 6.** Hacer un programa que imprima el numero de 1 a 500 y respectivamente el cuadrado y el cubo

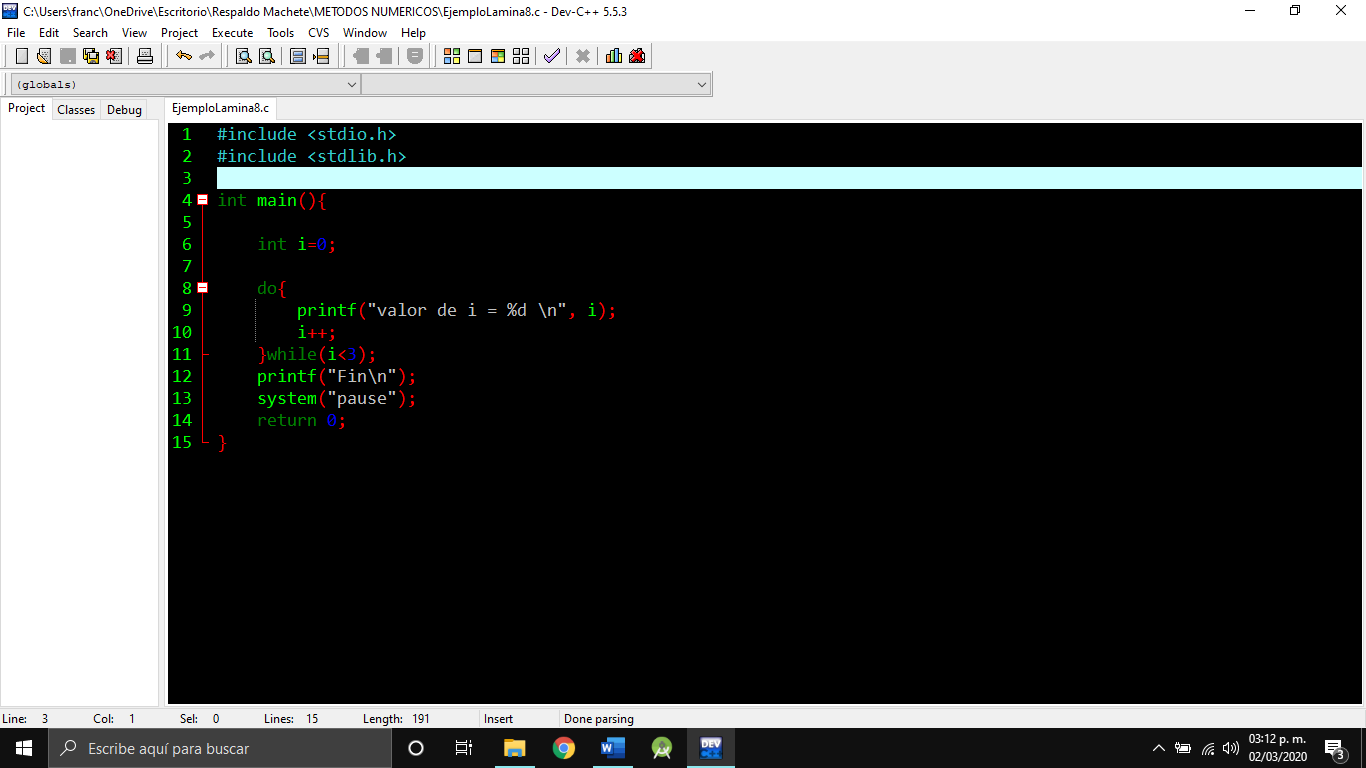
**Ejercicio 7.** Hacer un programa que imprima los primeros 1000 numeros que sean:

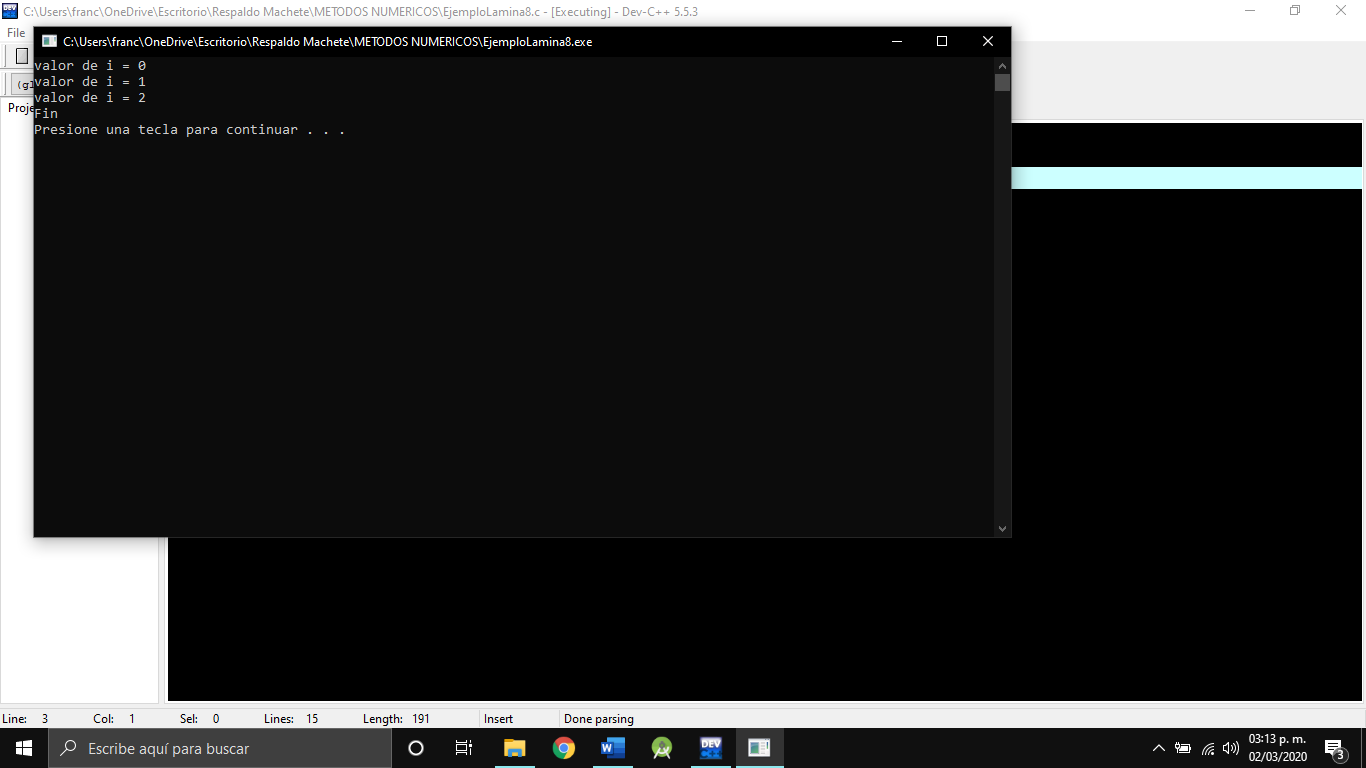
* Sean divisibles entre 3
* Sean divisibles entre 2 y 7
* Sea los primeros 100 y divisible entre 7





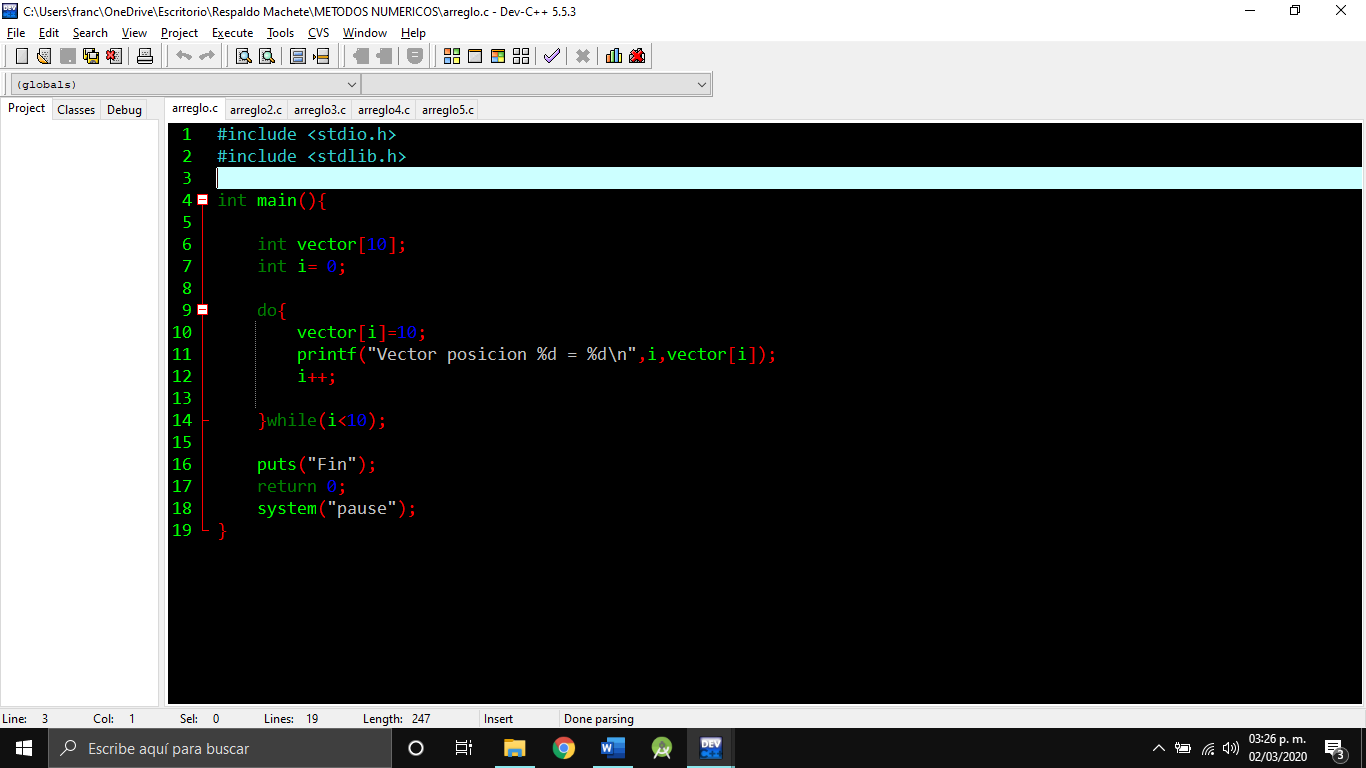
**Ejemplo 1.** Ahora pasamos a los do-while, es parecido al while solo que el ciclo entra por lo menos una vez y si se cumple la condición sigue entrando si no finaliza y sigue las demás instrucciones del código.

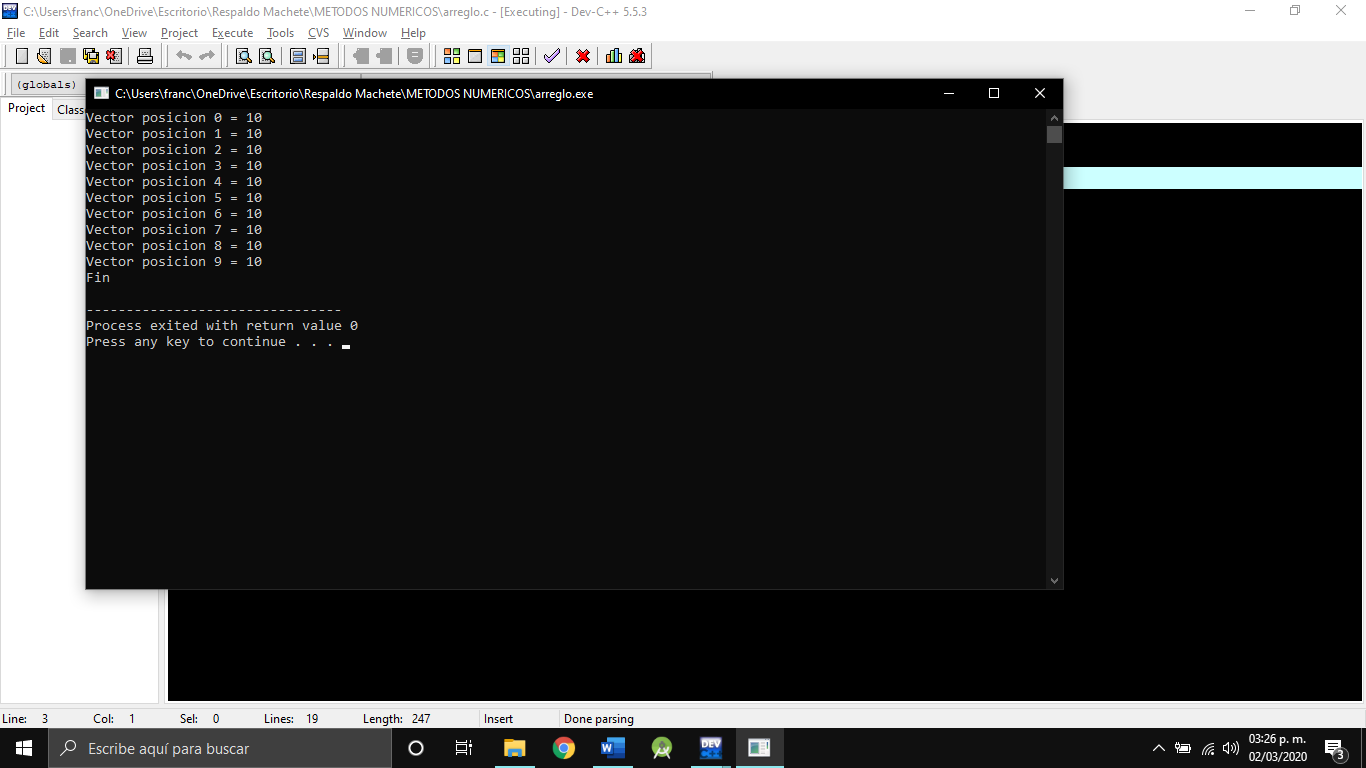




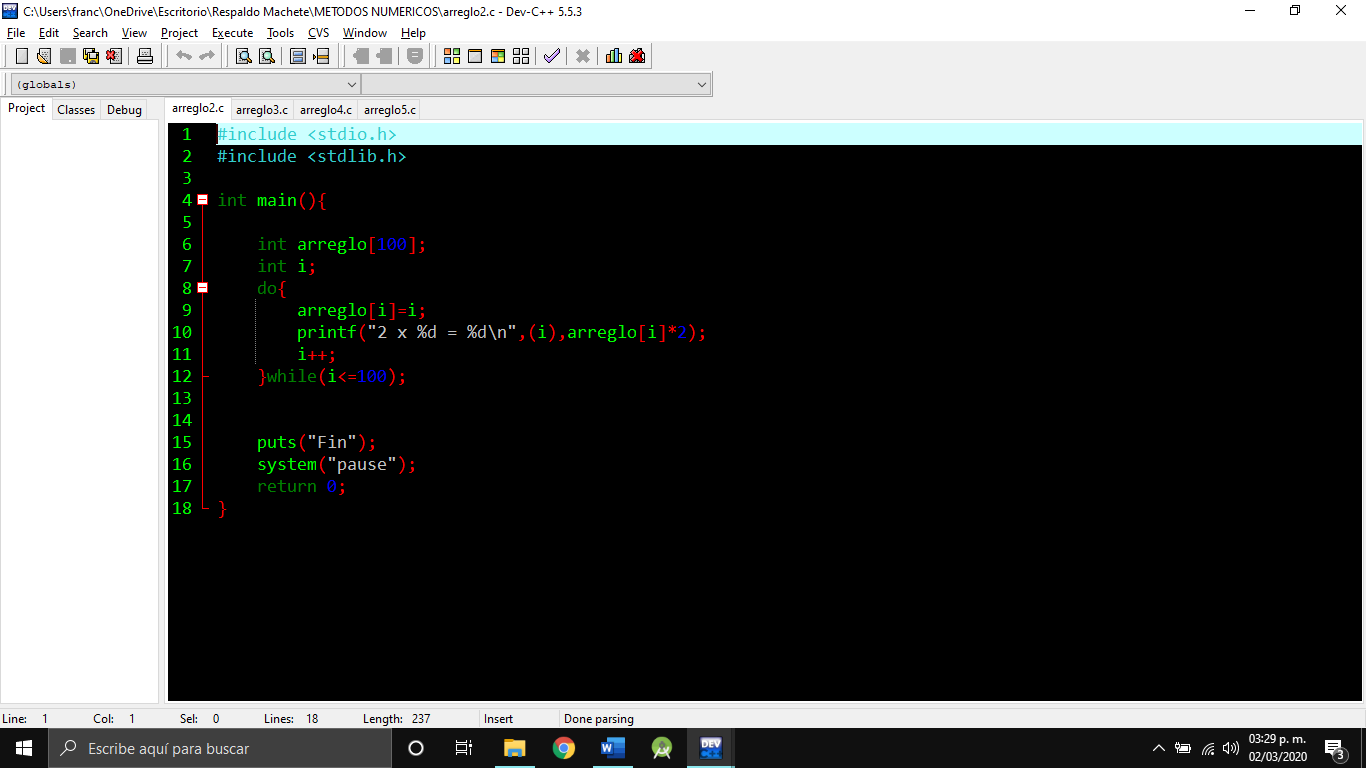
**Arreglos y ciclos for**

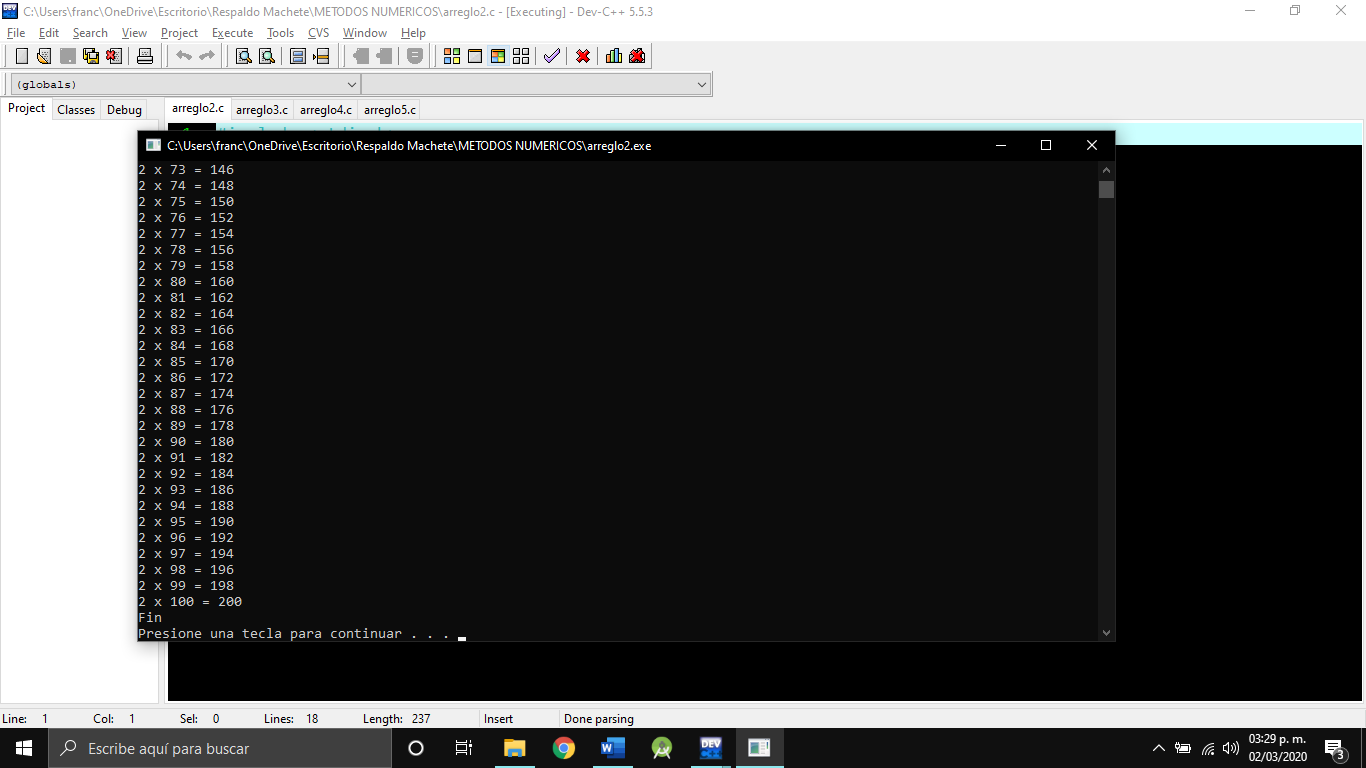
**Ejercicio 8.** Un arreglo por lo regular se maneja con un ciclo for, con su llenado o haciendo operaciones con sus posiciones; dentro de un ciclo for debe de tener los 3 componentes en sus paréntesis (inicialización, condición e incremento).



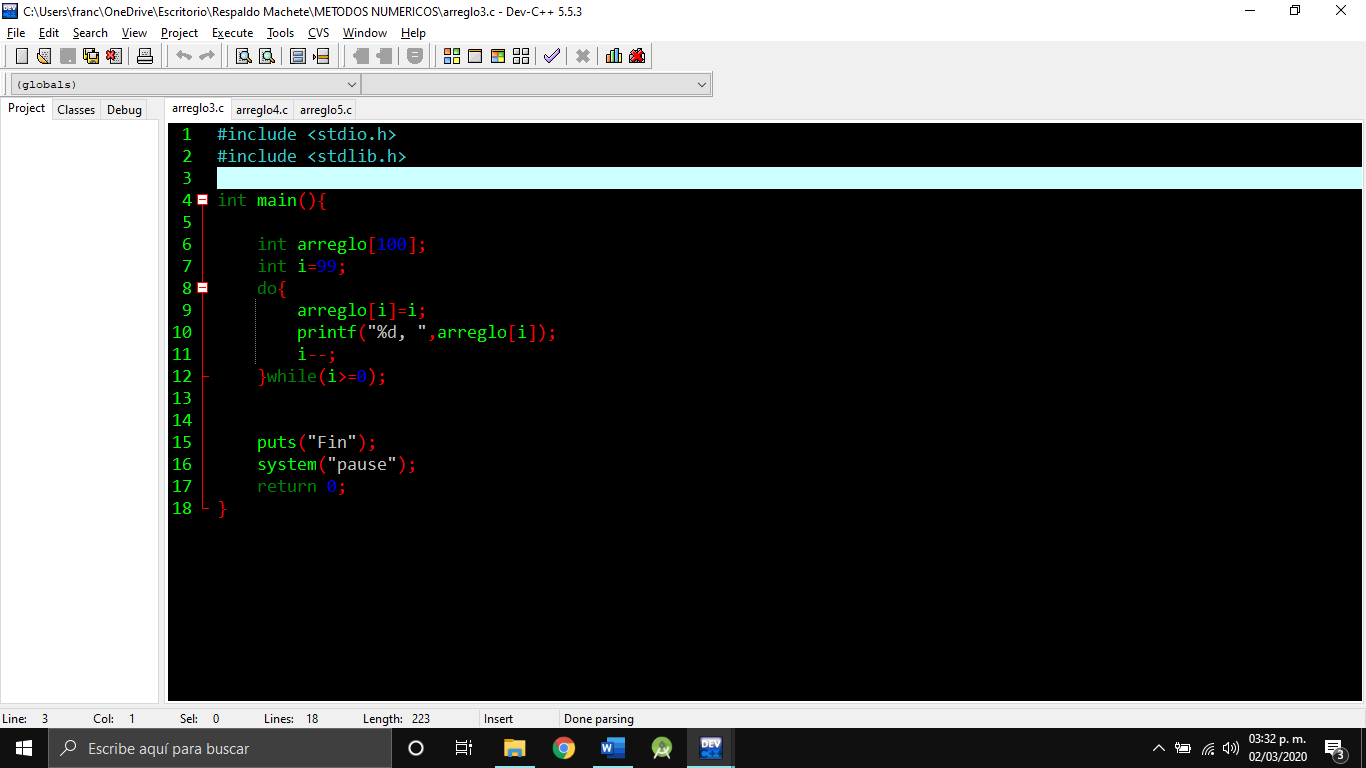


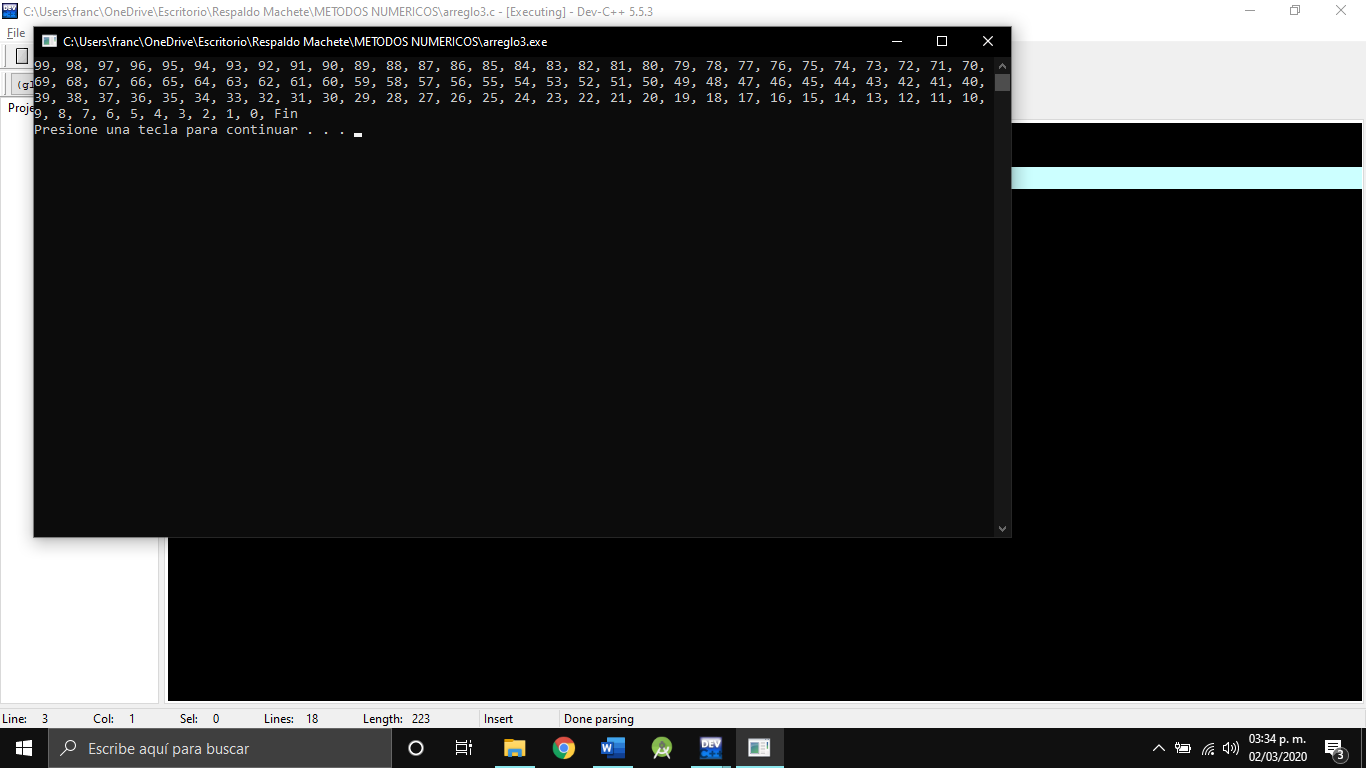
**Ejercicio 9.** Hacer un programa que imprima las tablas de multiplicar.



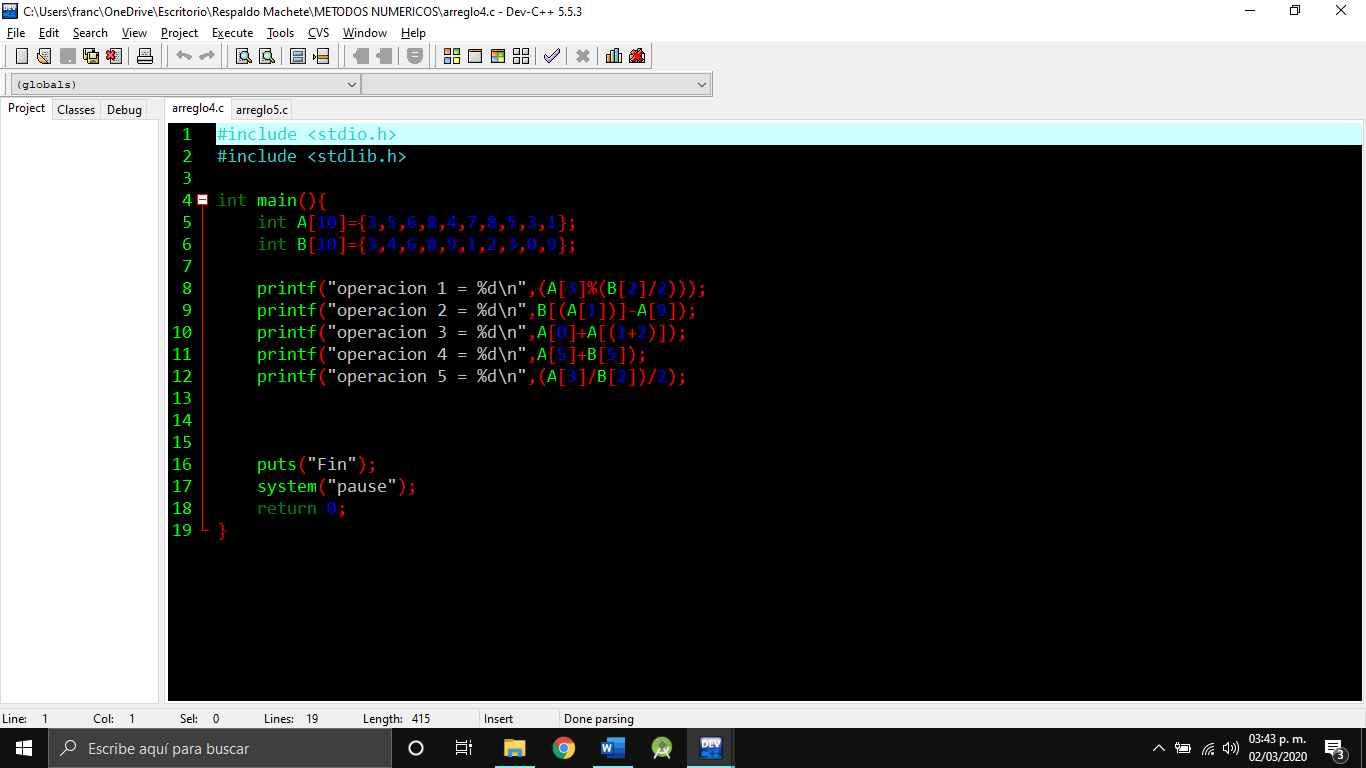


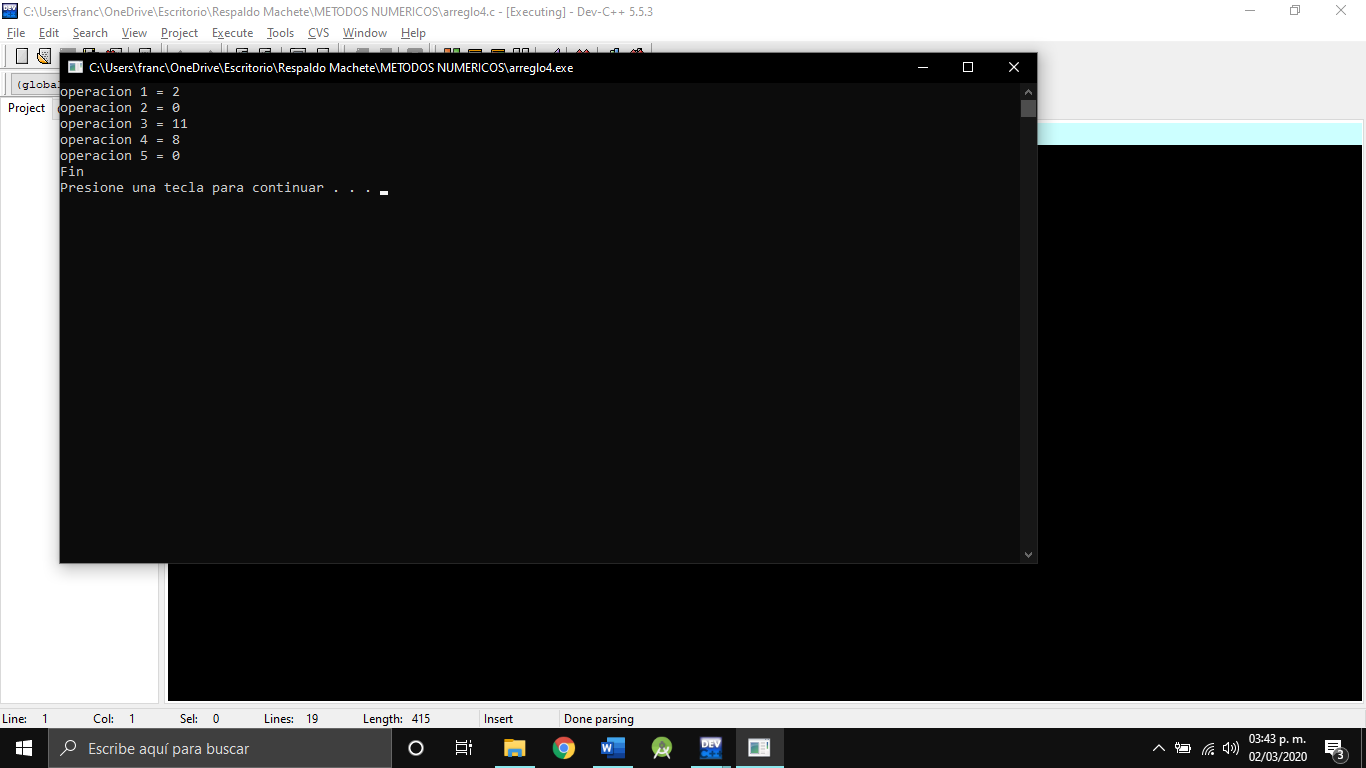
**Ejemplo.** Hacer un programa que muestre del 99 al 0



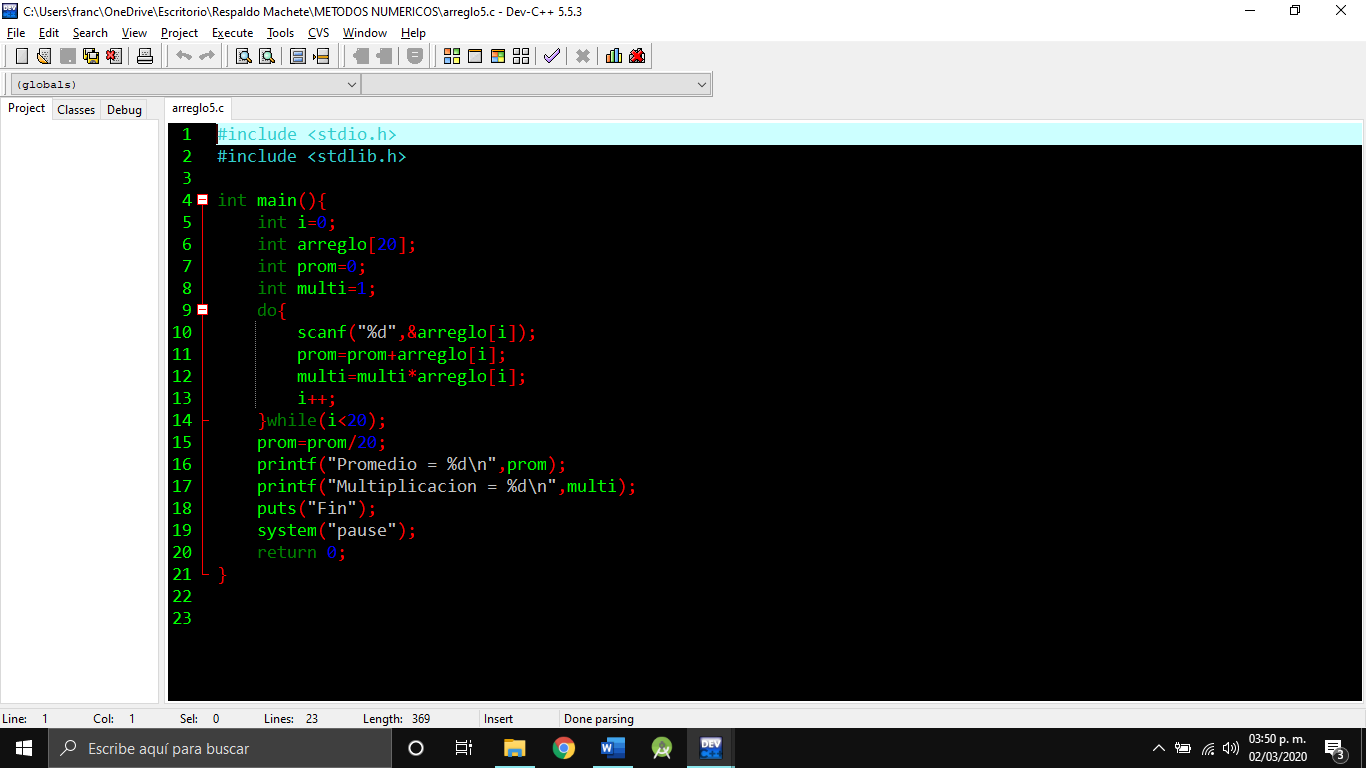


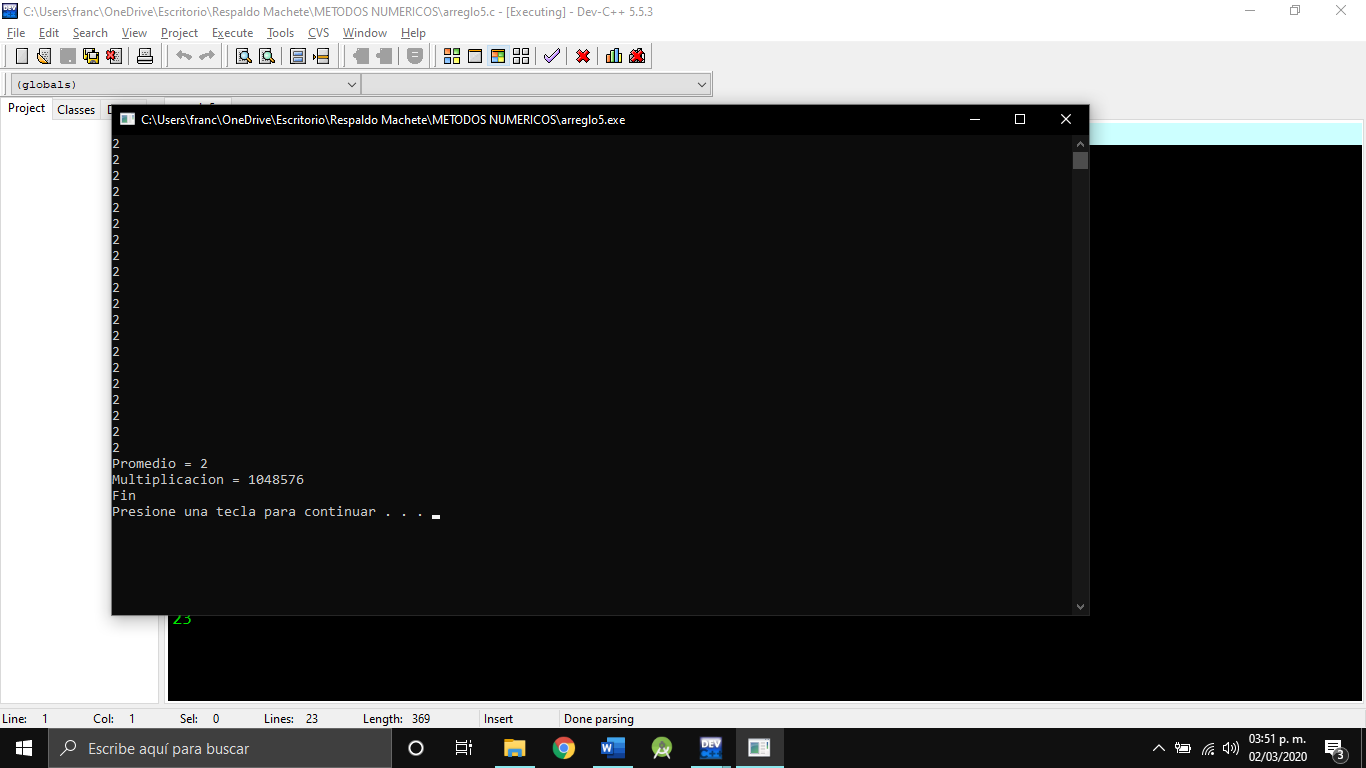
**Ejemplo.** Hacer un programa que juegue con las posiciones de 2 arreglos



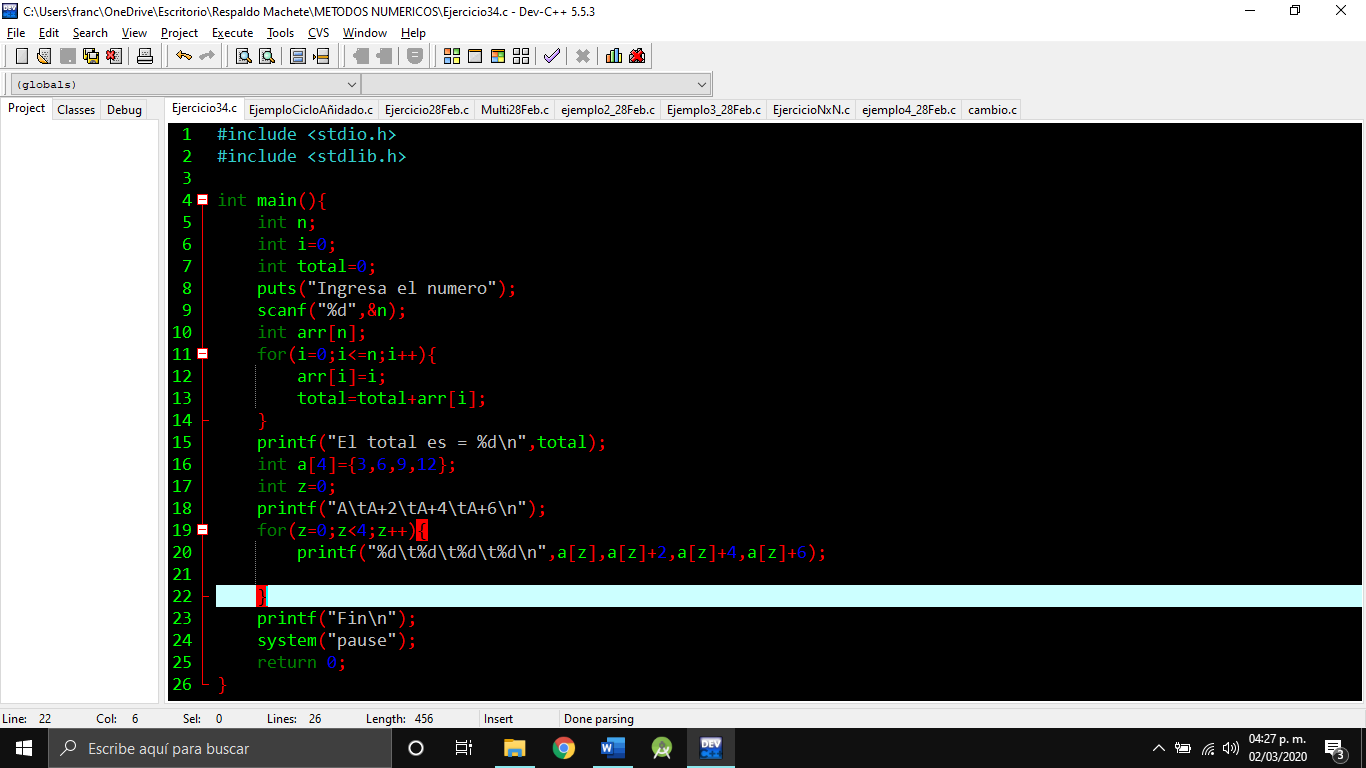


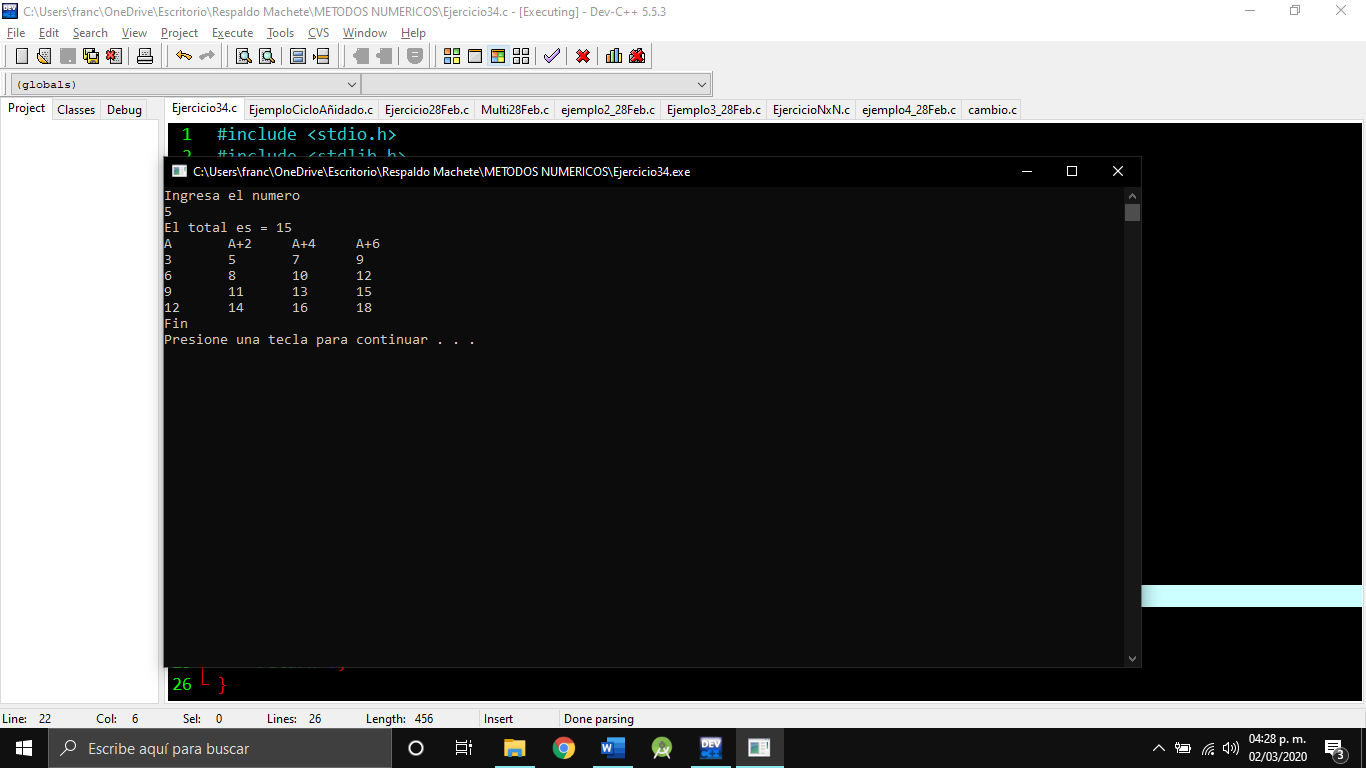
**Ejercicio.** Introducir 20 valores en un arreglo y sacar el promedio de esos valores y multiplicarlos



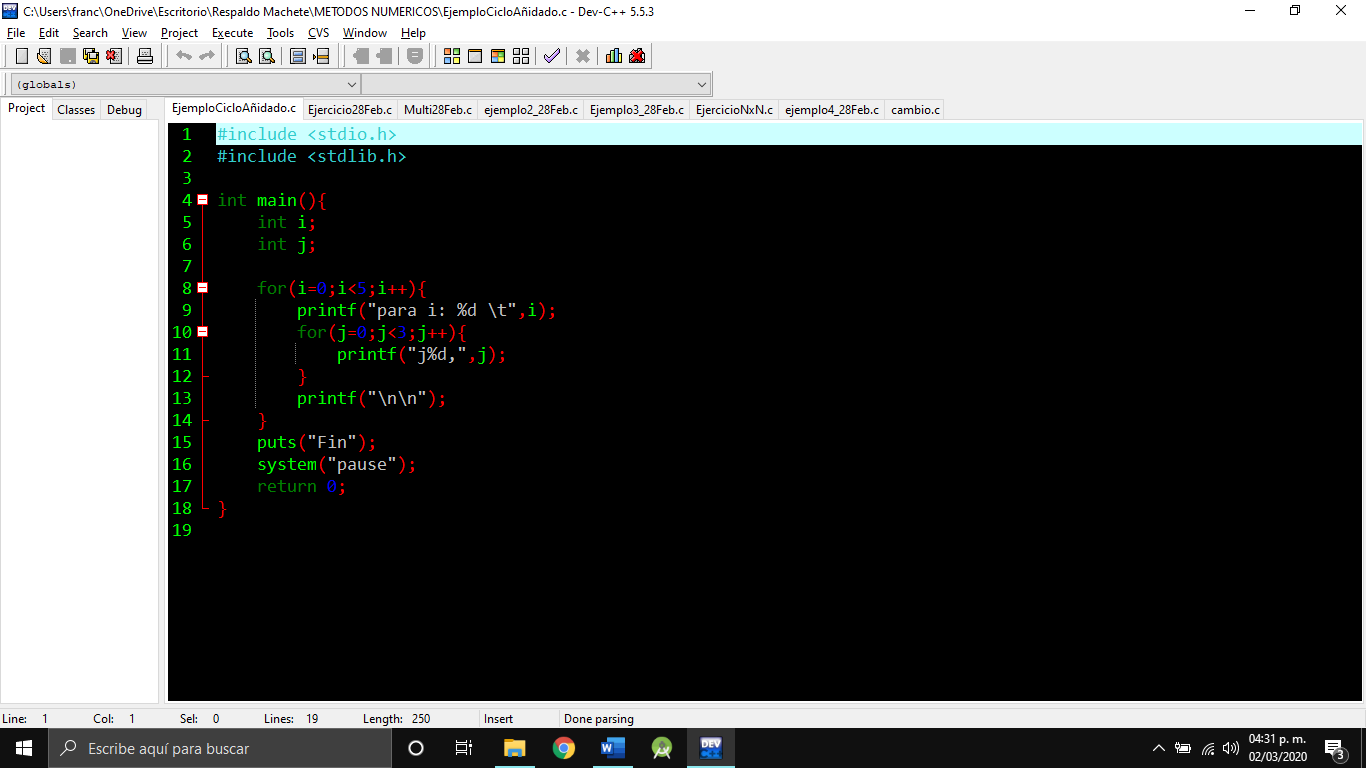


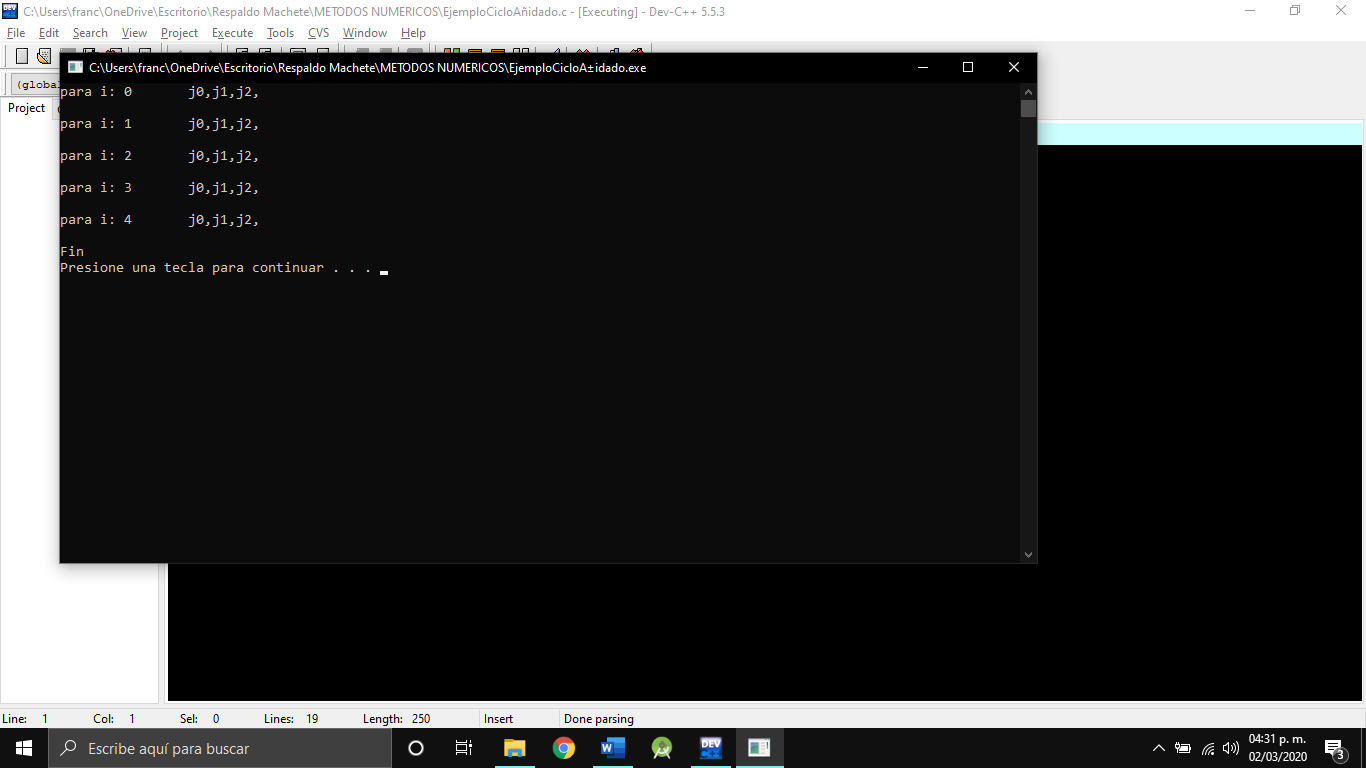
**Ejercicios.** Hacer 2 programas que introduzcas su tamaño y sume todos sus componentes y otro donde los elementos ya están dados y sumarles 2, 4 y 6 y mostrarlo en pantalla como tabla.



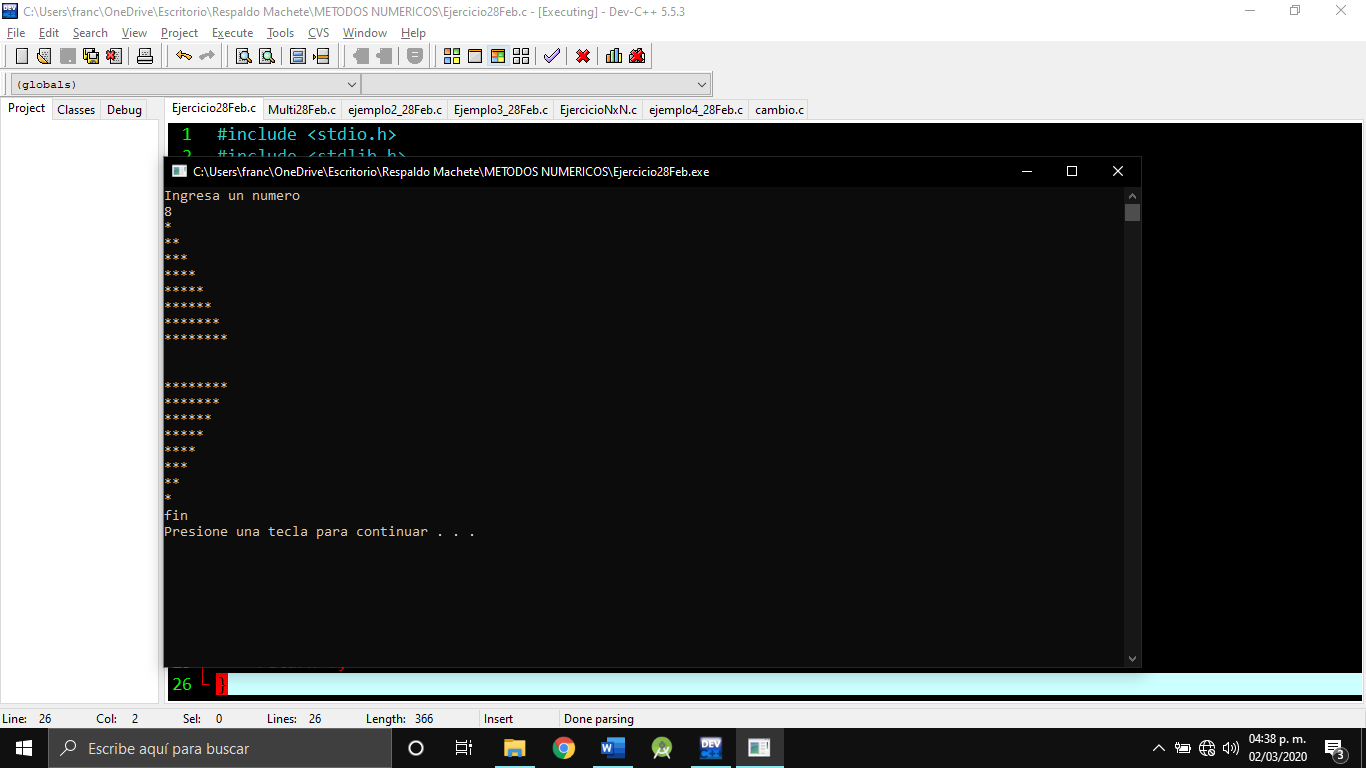
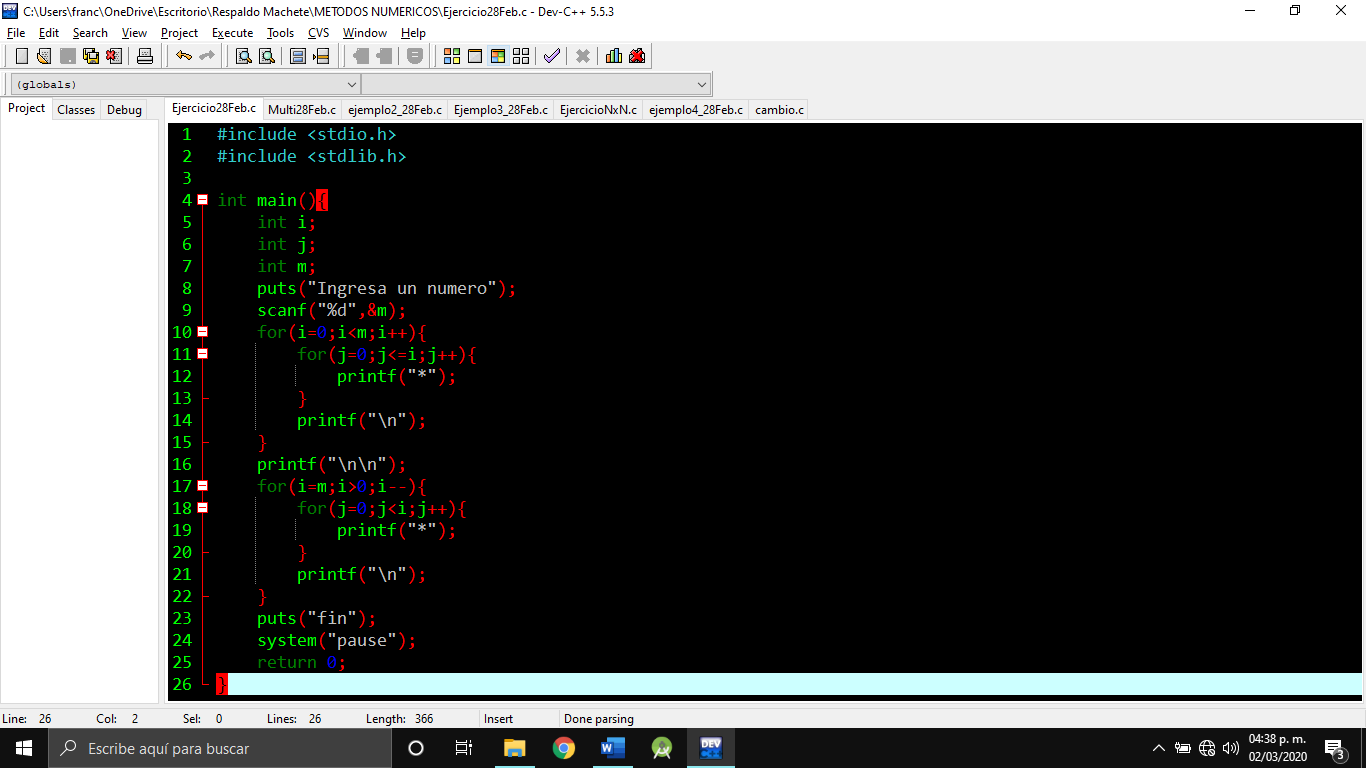


**Ejemplo.** Copiar el programa y ver su funcionamiento de un ciclo anidado.

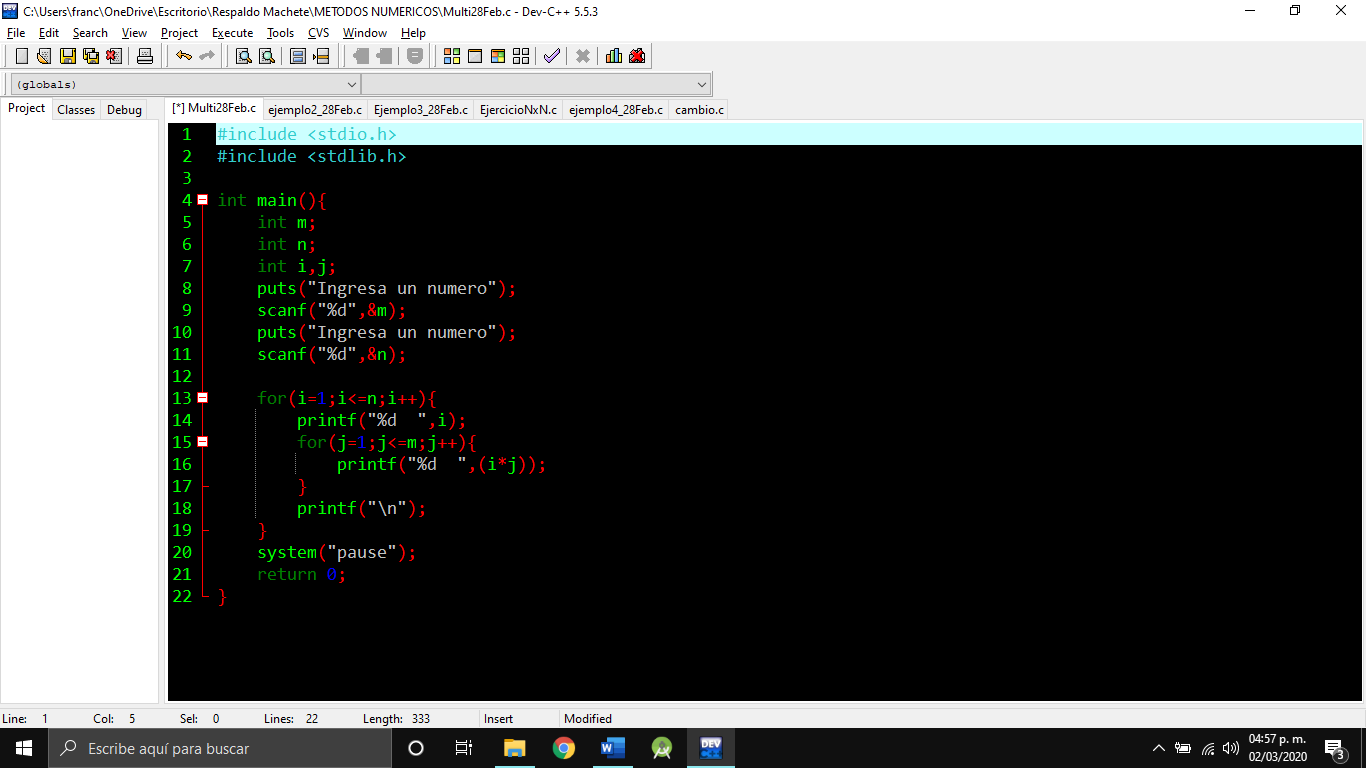


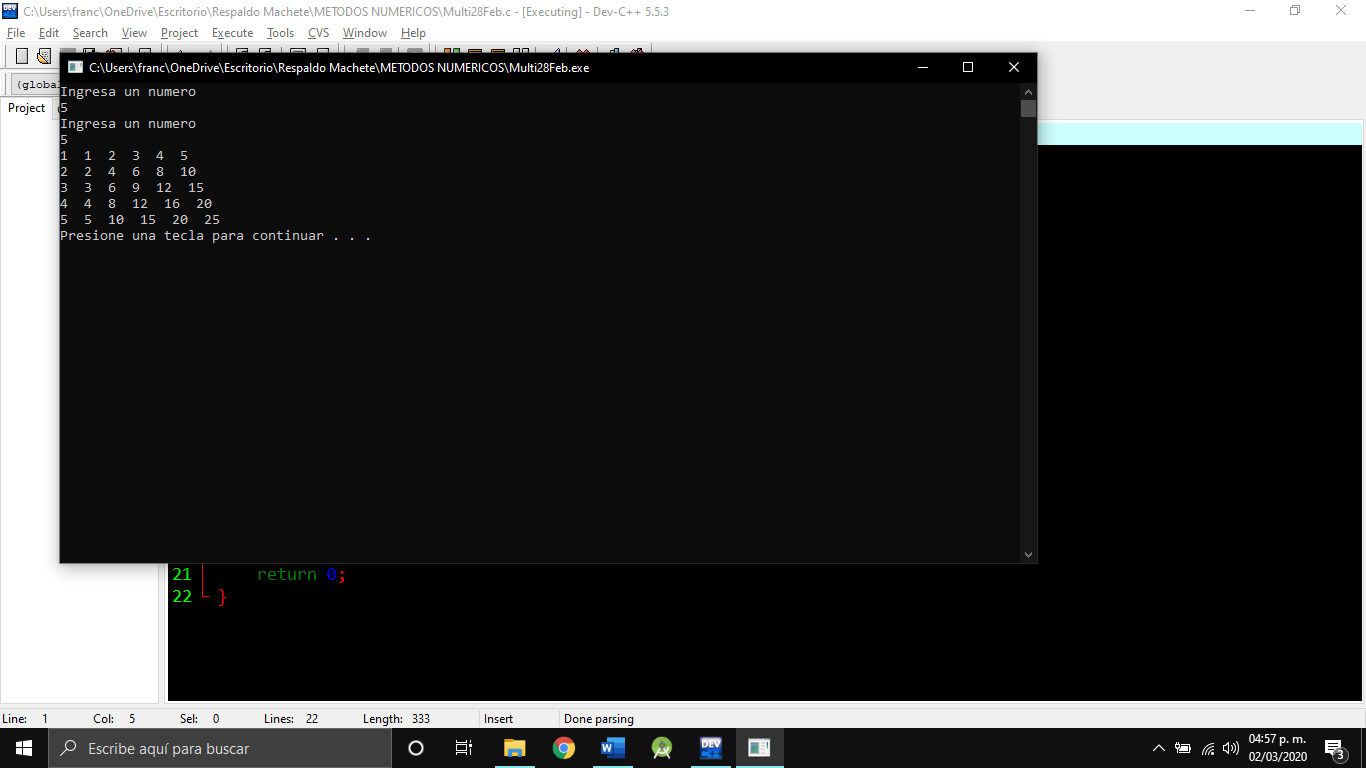


**Ejercicio.** Hacer un programa que introduzcas un numero e imprima una pirámide con eñ numero introducido en filas, al derecho al revés.

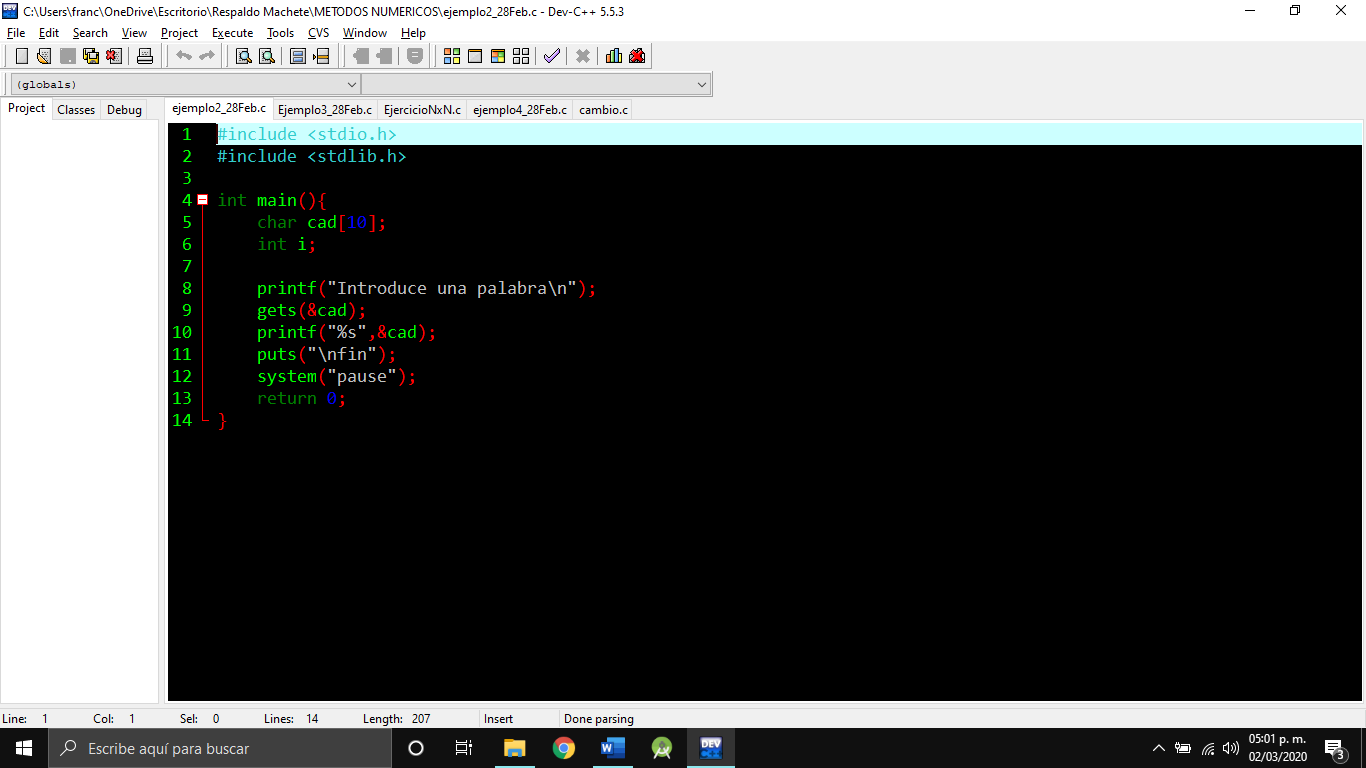


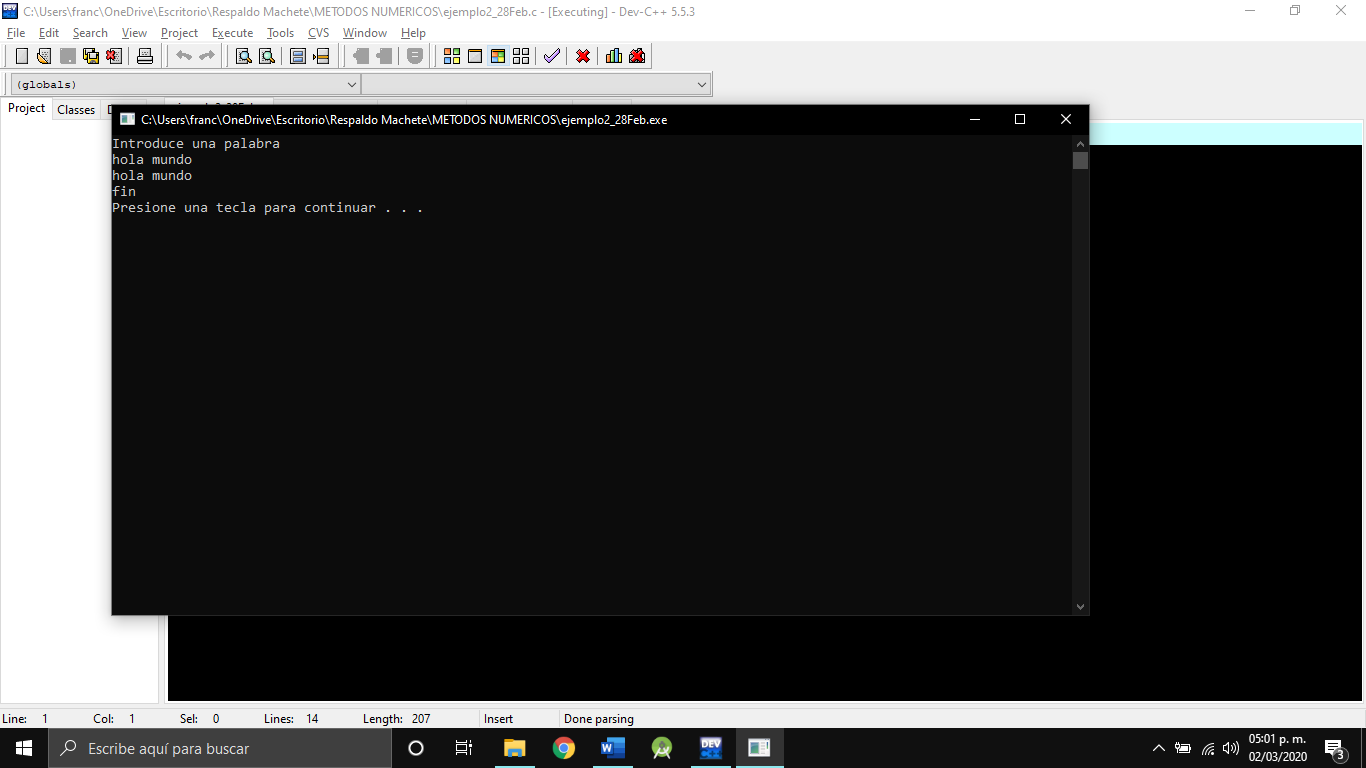
**Ejercicio.** Hacer un programa que introduzcas 2 números y hacer una tabla con la multiplicación de fila por columna



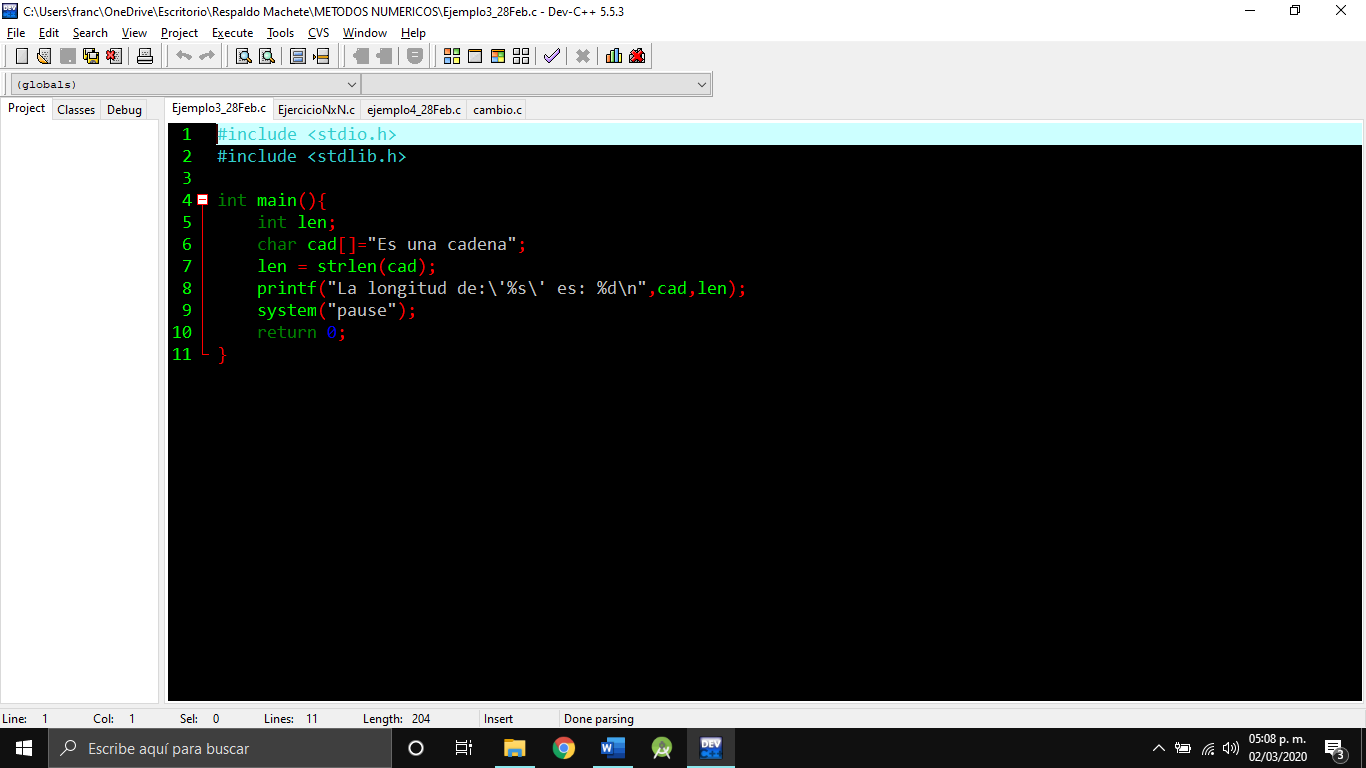


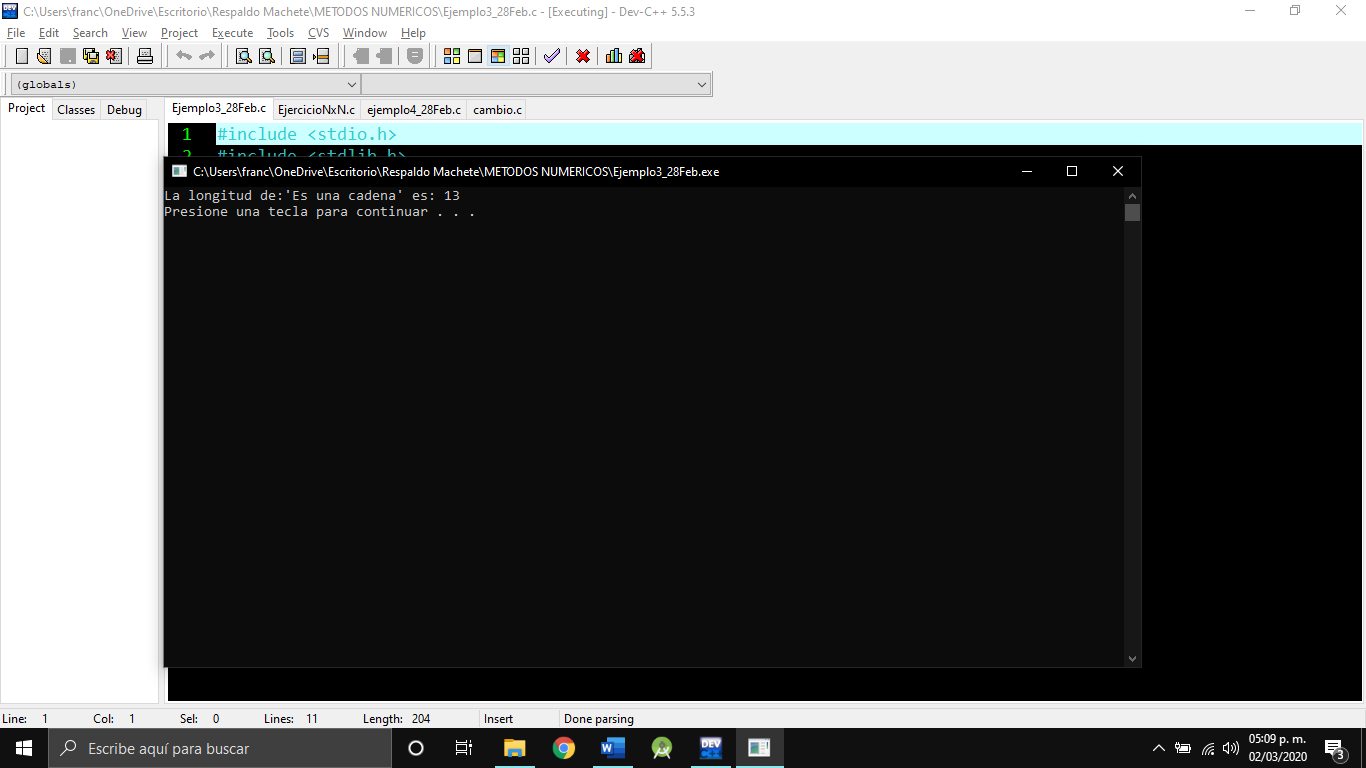
**Ejercicio.** Copiar el ejemplo en las diapositivas, para ver el funcionamiento de un arreglo de caracteres.

**gets ():** lee la variable y guarda la palabra con todo y espacios, ya que con printf solo guarda hasta que encuentra un espacio en blanco

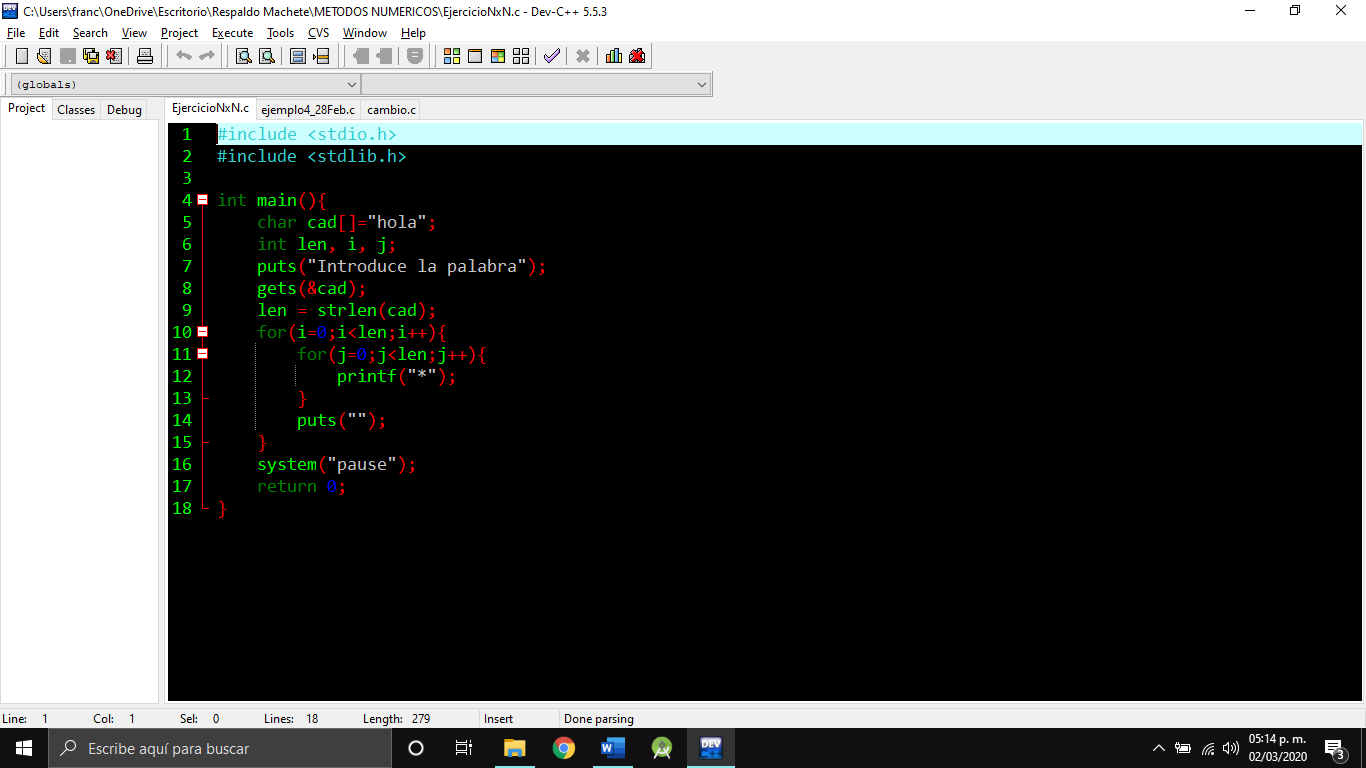


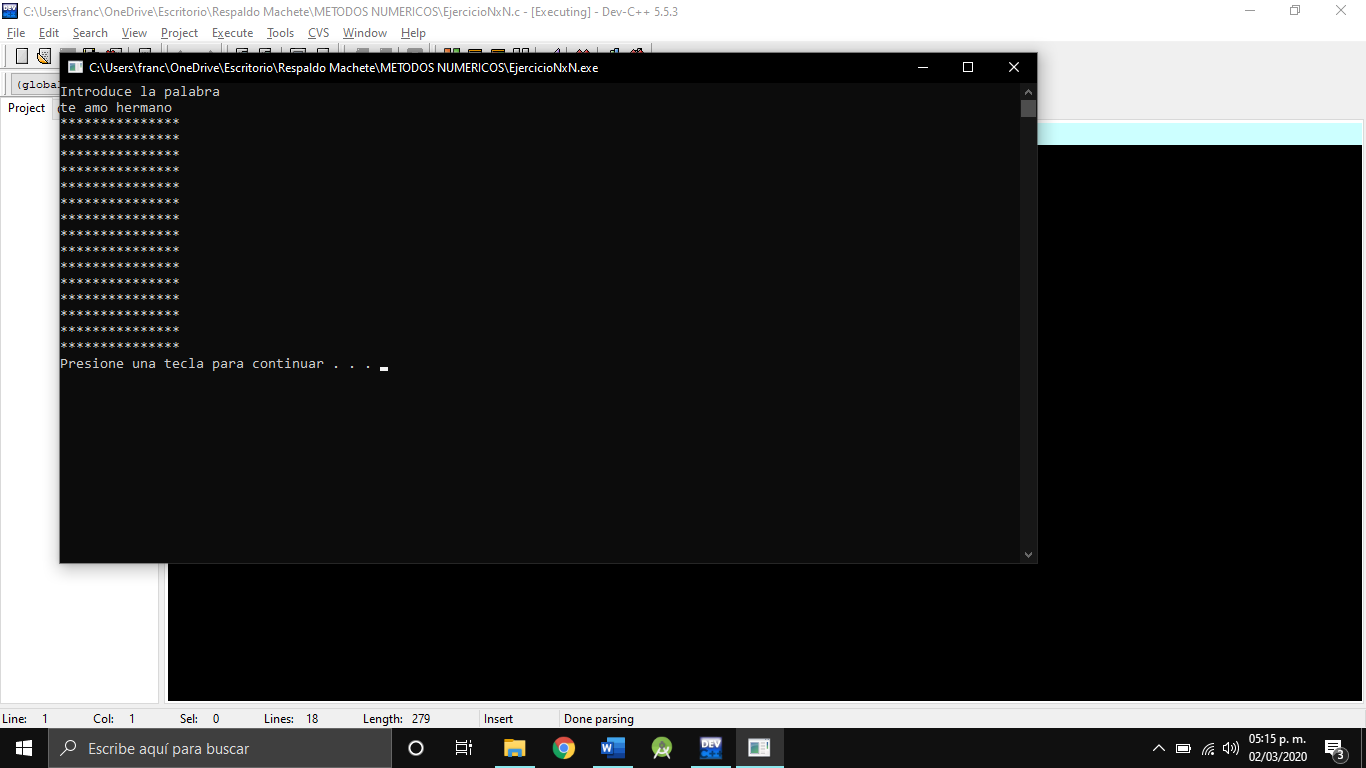
**Ejemplo.** Copiar el siguiente programa de las diapositivas y ver su funcionamiento. Aquí introducimos una nueva palabra que es **strlen ()**, que sirve para saber la longitud de un arreglo de caracteres.



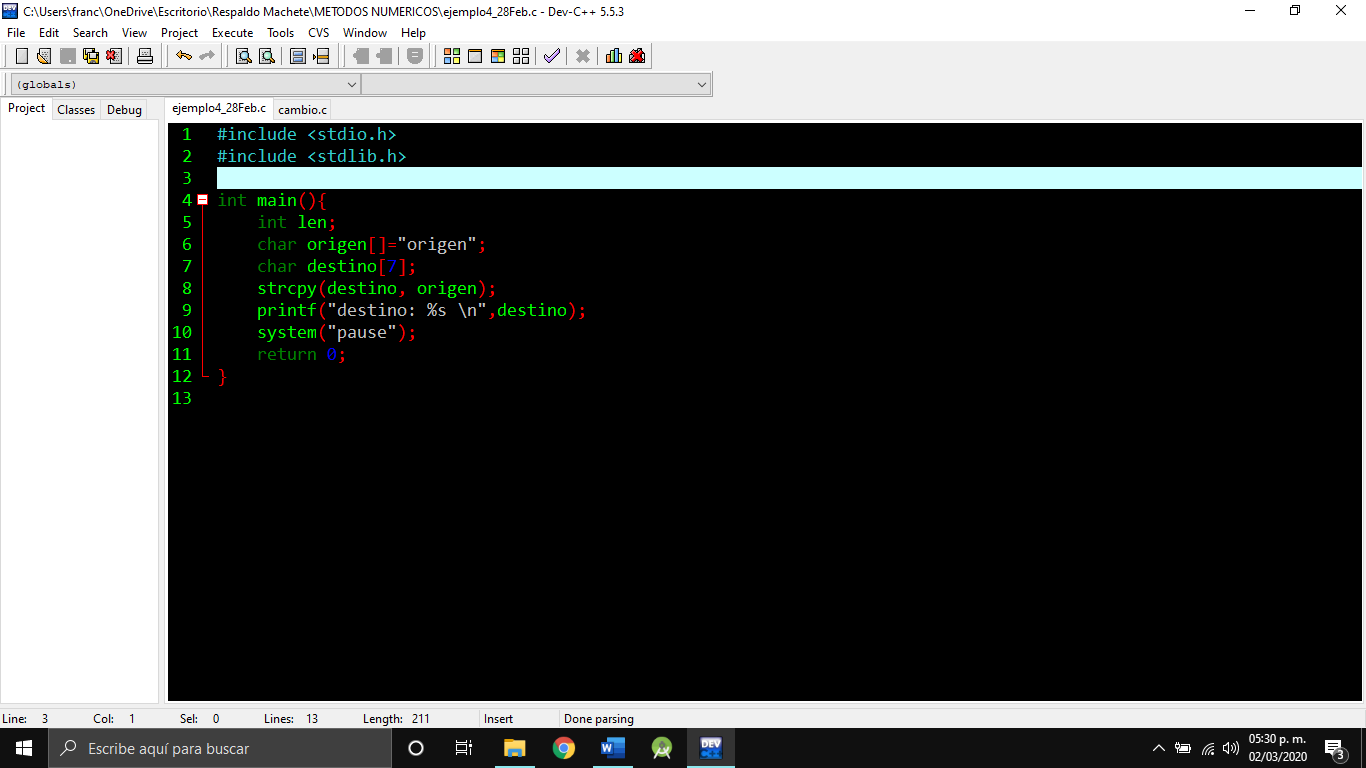


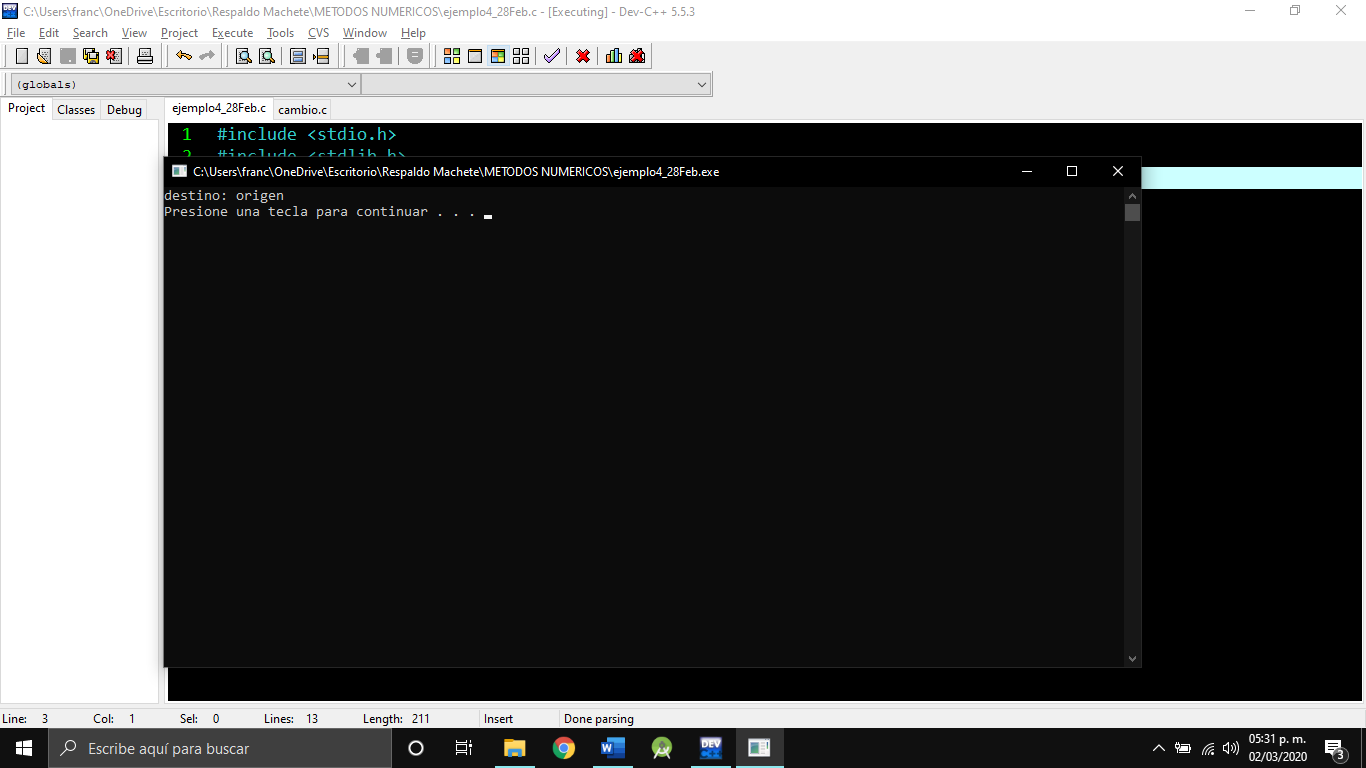
**Ejemplo.** Hacer un programa que introduzcas una palabra y de esa palabra cuente su tamaño y lo imprima n por n en asteriscos.





**Ejemplo.** Copiar el programa que esta en las diapositivas para ver el nuevo comando **strcpy ()** que copea los caracteres de un arreglo y lo guarda en otro arreglo





**Ejercicio.** Hacer un programa que ingreses 2 palabras en 2 arreglos y los cambies de lugar entre ellos.

