|  |  |  |
| --- | --- | --- |
|  | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorio de docencia |

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| *Profesor:* | Marco Antonio Martínez Quintana |
| *Asignatura:* | Fundamentos de programación |
| *Grupo:* | 3 |
| *No de Práctica(s):* | 10 |
| *Integrante(s):* | Flores Saavedra José Gerardo |
| *No. de Equipo de cómputo empleado:* | No Aplica |
| *No. de Lista o Brigada:* | 13 |
| *Semestre:* | 2021-1 |
| *Fecha de entrega:* | 07/12/2020 |
| *Observaciones:* |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_

**Depuración de programas**

**Objetivo:**

* Aprender las técnicas básicas de depuración de programas en C para revisar de manera precisa el flujo de ejecución de un programa y el valor de las variables; en su caso, corregir posibles errores.

**Introducción:**

Depurar un programa significa someterlo a un ambiente de ejecución controlado por medio de herramientas dedicadas a ello. Este ambiente permite conocer exactamente el flujo de ejecución del programa, el valor que las variables adquieren, la pila de llamadas a funciones, entre otros aspectos. Es importante poder compilar el programa sin errores antes de depurarlo.

La depuración de un programa es útil cuando:

* Se desea optimizar el programa: Debe realizarse un análisis exhaustivo del mismo en ejecución para averiguar cuál es su flujo de operación y encontrar formas de mejorarlo, o bien, encontrar puntos donde puede fallar con ciertos tipos de entrada de datos.
* El programa tiene algún fallo: Muchas veces encontrar este tipo de fallos suele ser difícil, ya sea porque la percepción del programador no permite encontrar la falla en su diseño o porque la errata es muy pequeña, pero crucial. En este caso es de mucha utilidad conocer paso a paso cómo se ejecutan las estructuras de control, qué valor adquieren las variables, etc.
* El programa tiene un error de ejecución o defecto: Suele ocurrir por error en el diseño o implementación del programa en las que no se contemplan las limitaciones del lenguaje de programación o el equipo donde el programa se ejecuta. Como el programa se detiene inesperadamente, no se conoce la parte del programa donde se provoca el defecto, teniendo que recurrir a la depuración para encontrarlo. El más común de este tipo de defecto es la “violación de segmento”.

Algunas funciones básicas que tienen en común la mayoría de los depuradores son las siguientes:

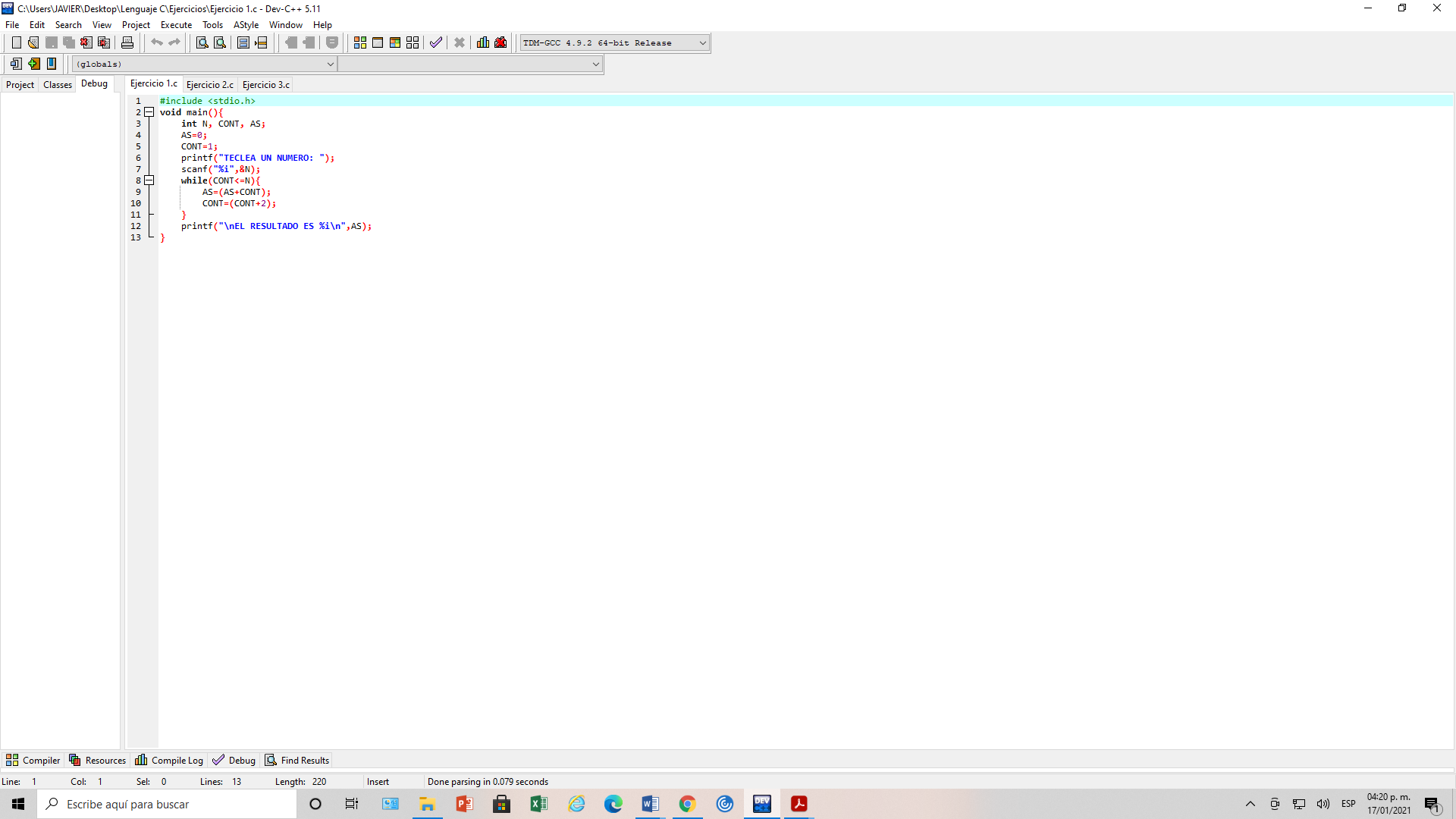
* Ejecutar el programa: se procede a ejecutar el programa en la herramienta de depuración ofreciendo diversas opciones para ello.
* Mostrar el código fuente del programa: muestra cuál fue el código fuente del programa con el número de línea con el fin de emular la ejecución del programa sobre éste, es decir, se indica qué parte del código fuente se está ejecutando a la hora de correr el programa.
* Punto de ruptura o breakpoint: sirve para detener la ejecución del programa en algún punto indicado previamente por medio del número de línea. Como la ejecución del programa es más rápida de lo que podemos visualizar y entender, se suelen poner puntos de ruptura para conocer ciertos parámetros de la ejecución como el valor de las variables en determinados puntos del programa. También sirve para verificar hasta qué punto el programa se ejecuta sin problemas y en qué parte podría existir el error, esto es especialmente útil cuando existe un error de ejecución.
* Continuar: continúa con la ejecución del programa después del punto de ruptura.
* Ejecutar la siguiente instrucción: cuando la ejecución del programa se ha detenido por medio del depurador, esta función permite ejecutar una instrucción más y detener el programa de nuevo. Esto es útil cuando se desea estudiar detalladamente una pequeña sección del programa. Si en la ejecución existe una llamada a función se ingresará a ella.
* Ejecutar la siguiente línea: es muy similar a la función anterior, pero realizará todas las instrucciones necesarias hasta llegar a la siguiente línea de código. Si en la ejecución existe una llamada a función se ignorará.
* Ejecutar la instrucción o línea anterior: deshace el efecto provocado por alguna de las funciones anteriores para volver a repetir una sección del programa.
* Visualizar el valor de las variables: permite conocer el valor de alguna o varias variables.

Dependiendo de la herramienta usada para compilar el programa, si es de consola o de terminal, su uso y las funciones disponibles variarán

En las IDE (Entornos de Desarrollo Interactivo), suelen existir herramientas de depuración integradas de manera gráfica. Existen dos modos de desarrollar un programa y producir el archivo ejecutable que son “Debug” y “Release”. El primer modo se recomienda exclusivamente durante el desarrollo del programa para poder depurarlo continuamente durante cualquier prueba de ejecución. El segundo modo se establece cuando el programa ha sido terminado y totalmente probado.

**Actividades:**

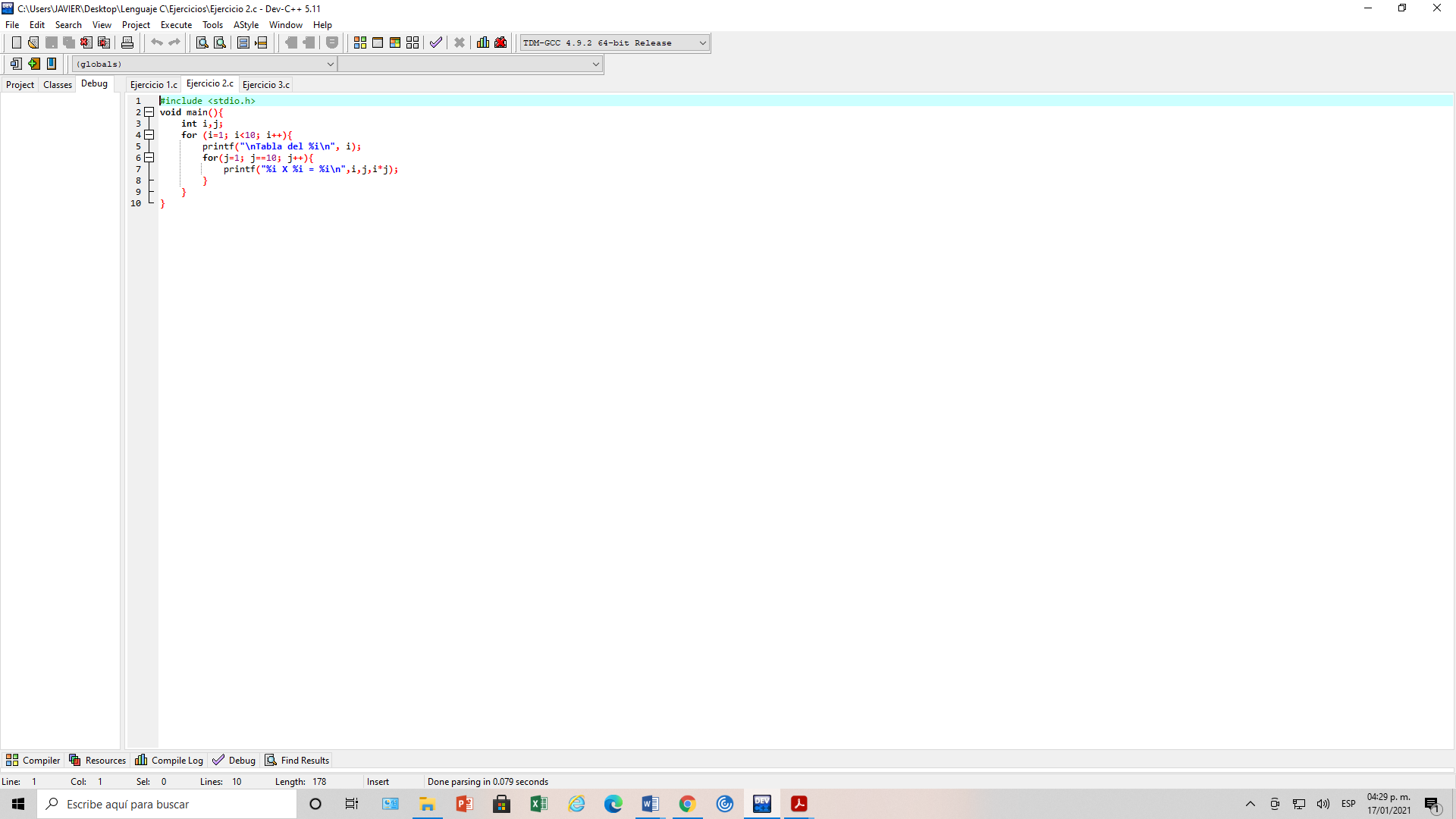
Para el siguiente código fuente, utilizar algún entorno de depuración para encontrar la utilidad del programa y la funcionalidad de los principales comandos de depuración, como punto de ruptura, ejecución de siguiente línea o instrucción.



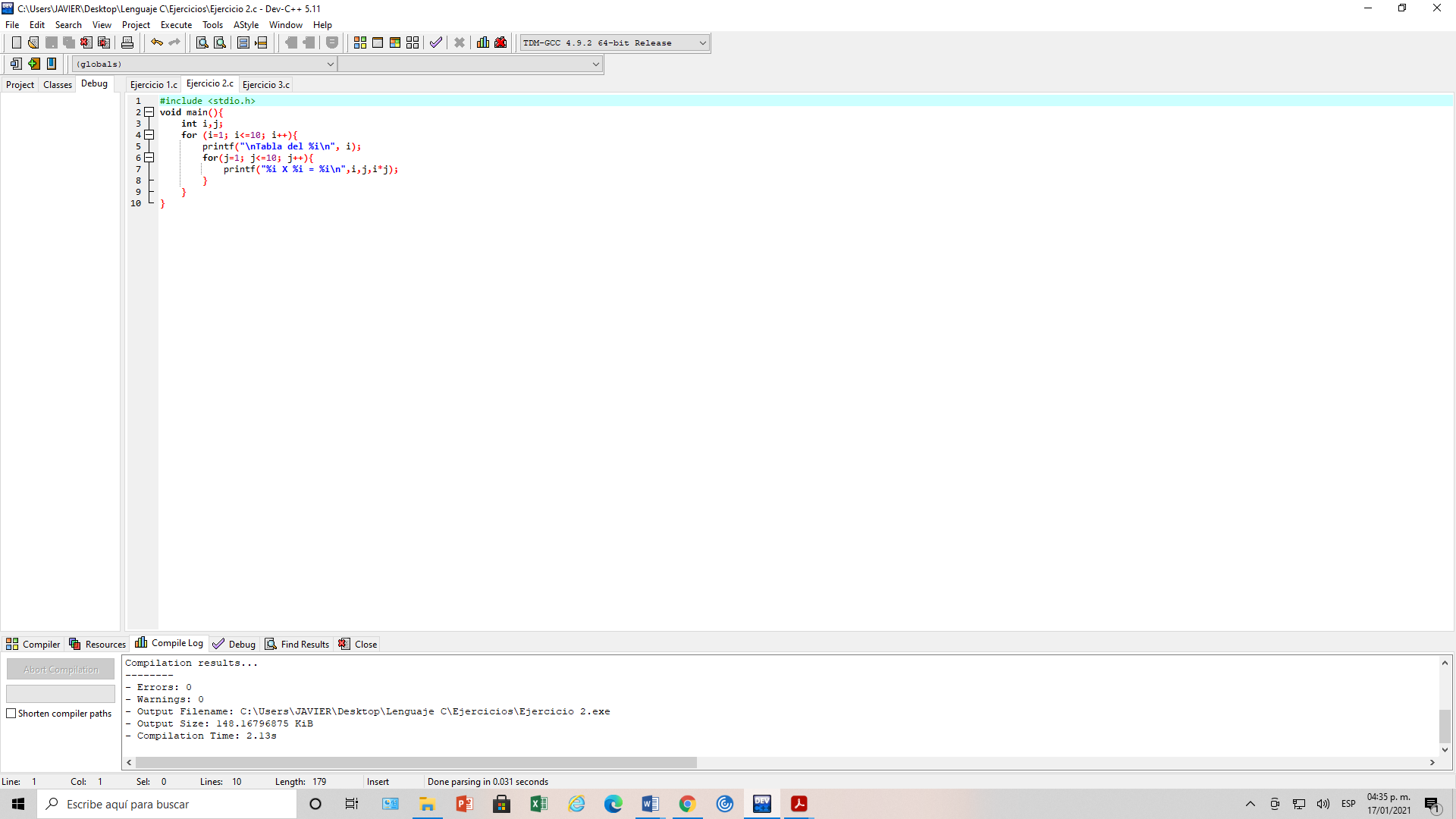
Funcionalidad del programa: El programa guarda un número y le suma un contador siempre y cuando el número sea mayor que el contador, posteriormente, le suma 2 unidades al contador.

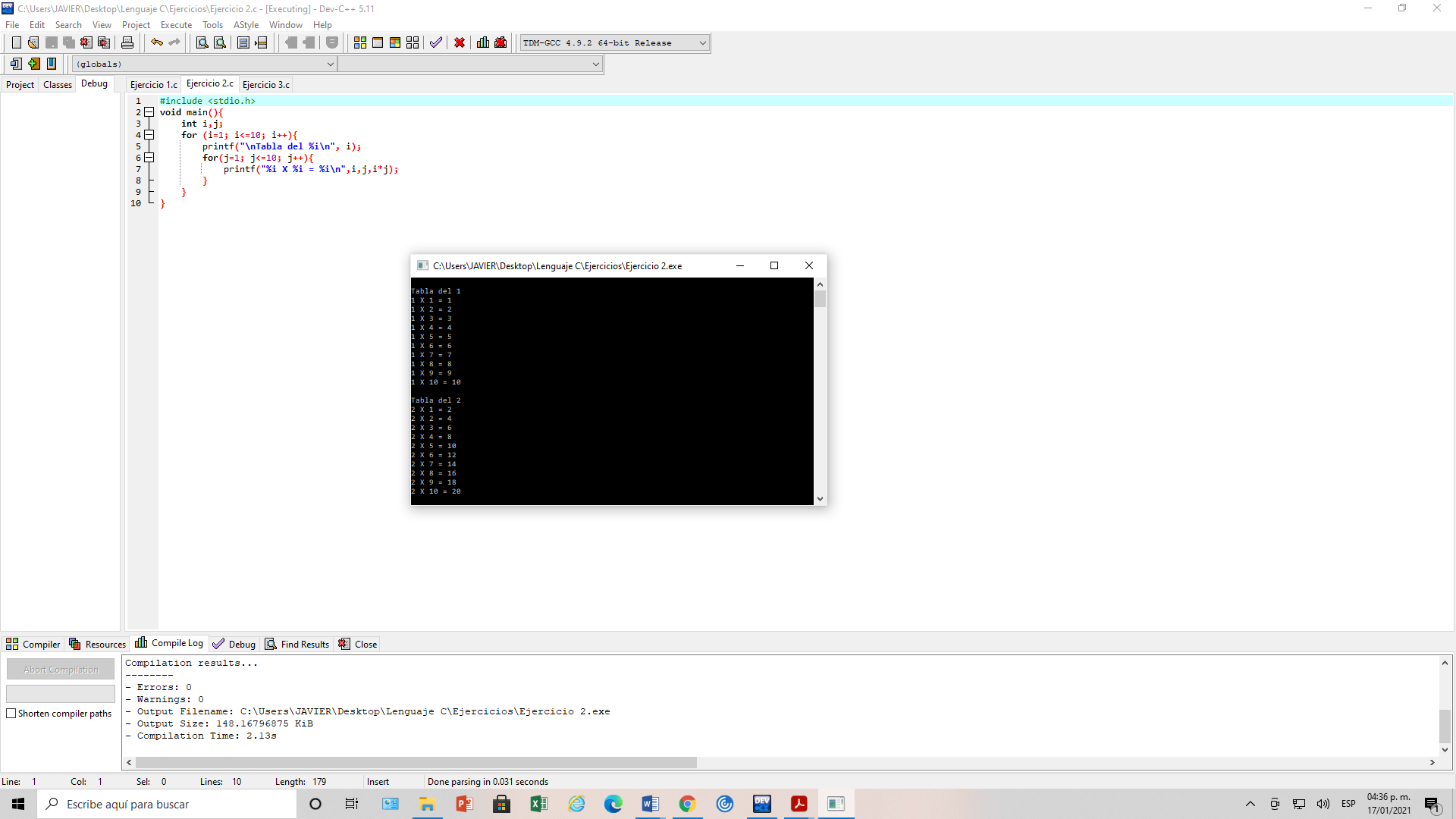
Función de los comandos básicos: Los puntos de ruptura o breakpoints sirven para saber los valores que tiene la variable en cada momento del código y a partir de donde se detuvo se puede ir avanzando, retrocediendo o prácticamente ver a detalle, línea por línea, el programa para lograr la optimización de este.

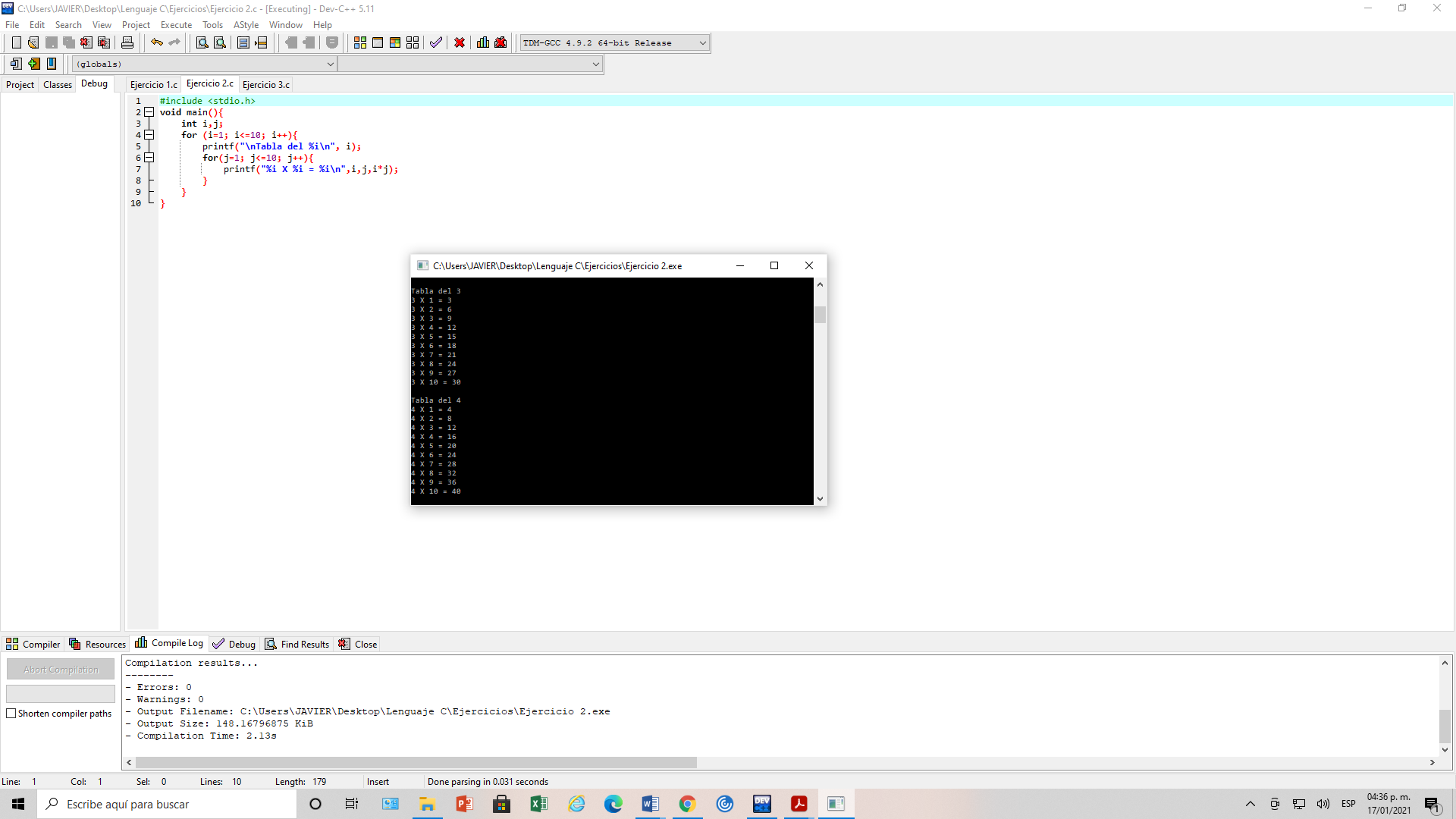
El siguiente programa debe mostrar las tablas de multiplicar desde la del 1 hasta la del 10. En un principio no se mostraba la tabla del 10, luego después de intentar corregirse sin un depurador dejaron de mostrarse el resto de tablas. Usar un depurador de C para averiguar el funcionamiento del programa y corregir ambos problemas.

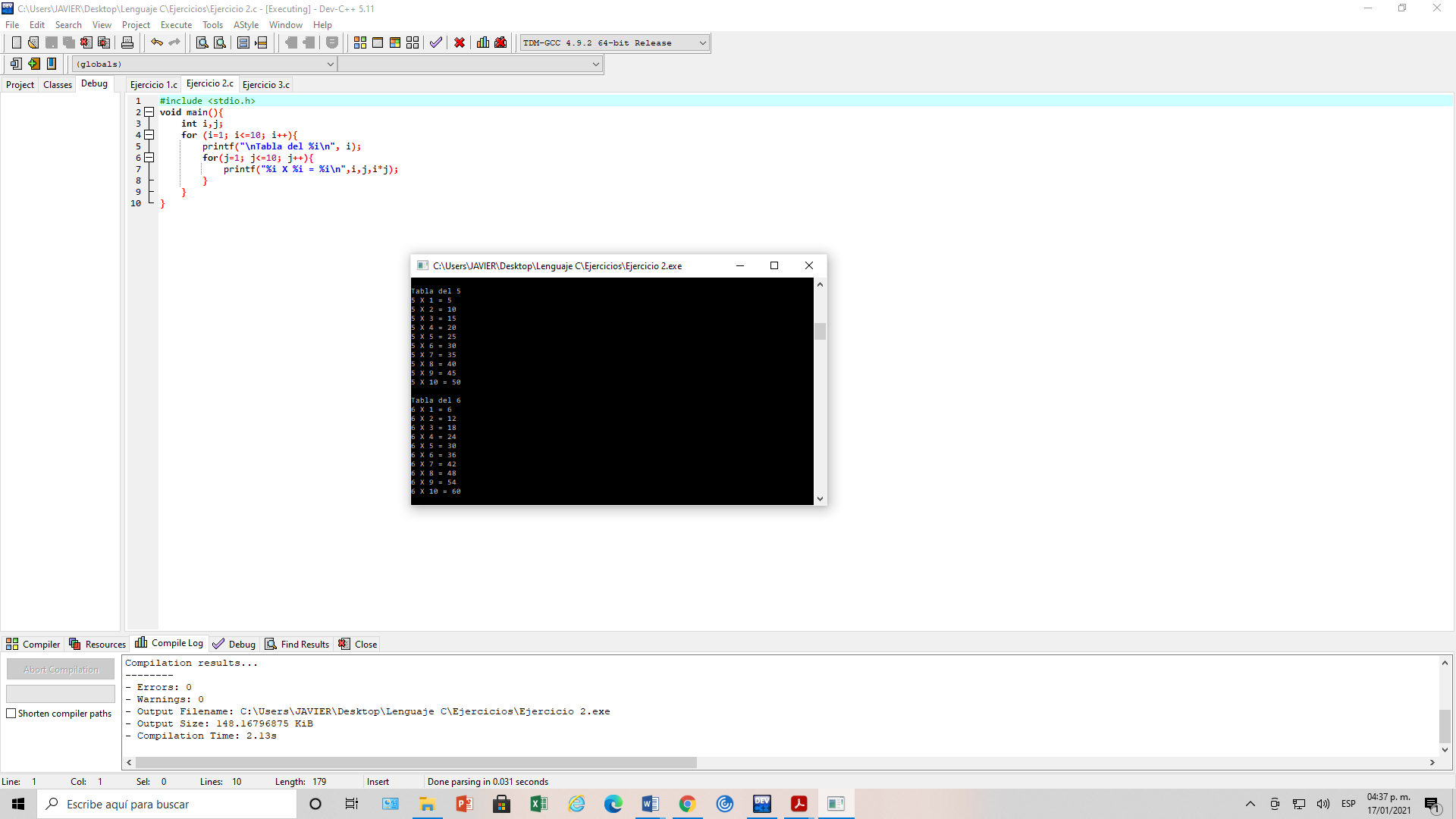


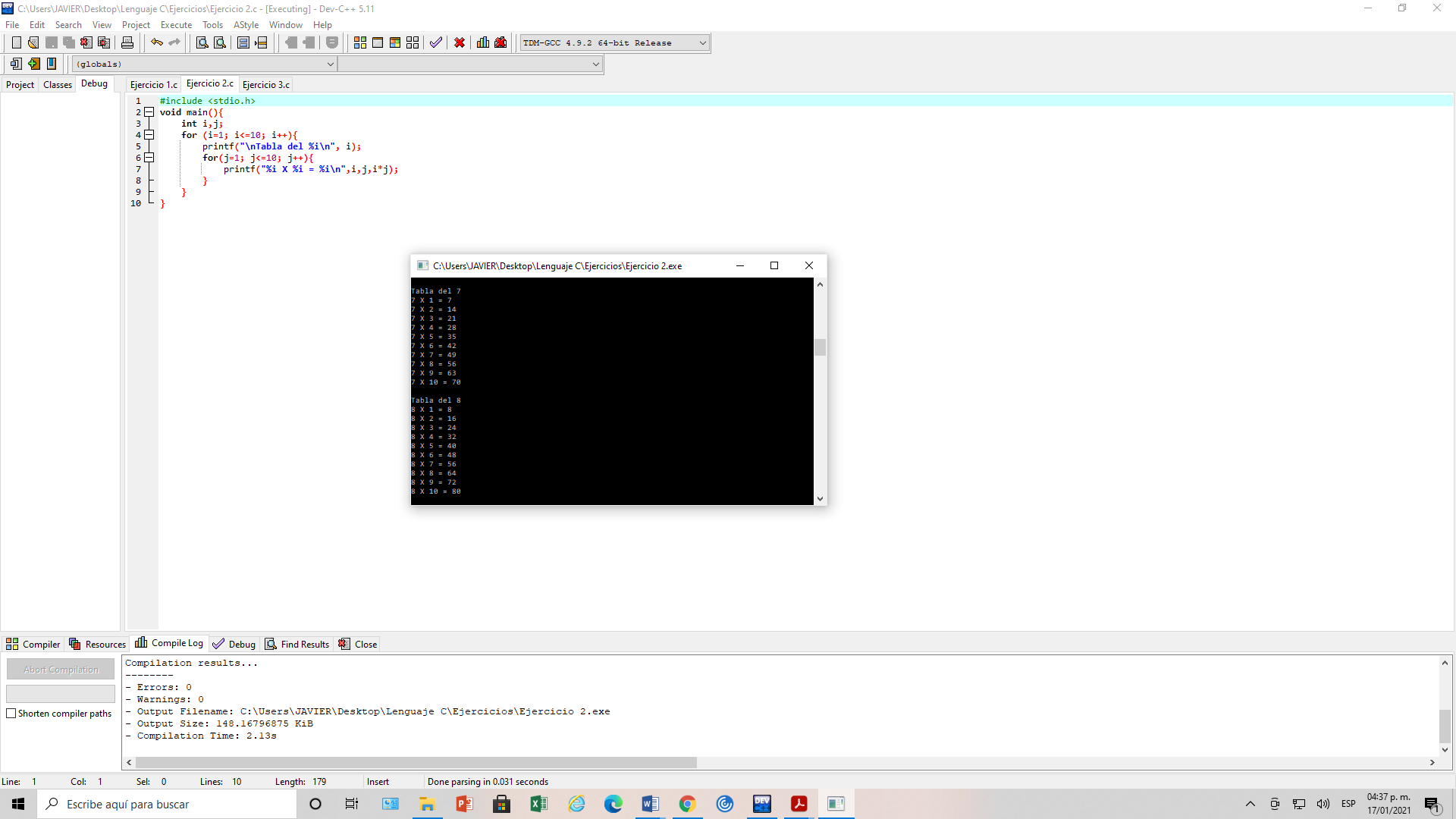
El programa tiene dos códigos de repetición “for” que son los que hacen que se muestren las tablas del 1 al 10 en orden. Para arreglar este programa, tenemos que cambiar el igual a y el menor que 10 en los “for” por un menor o igual a que para que se arregle este problema.

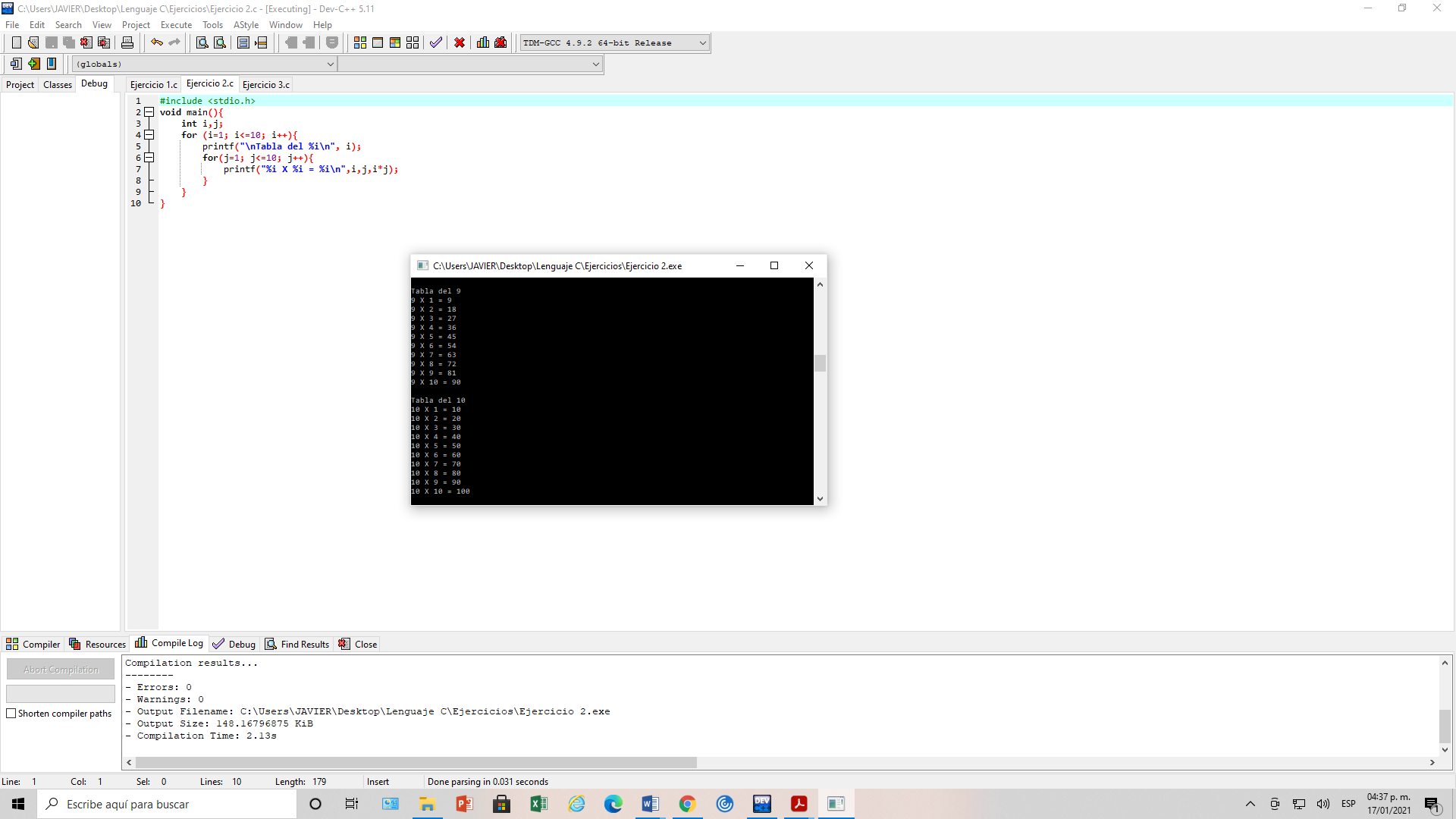




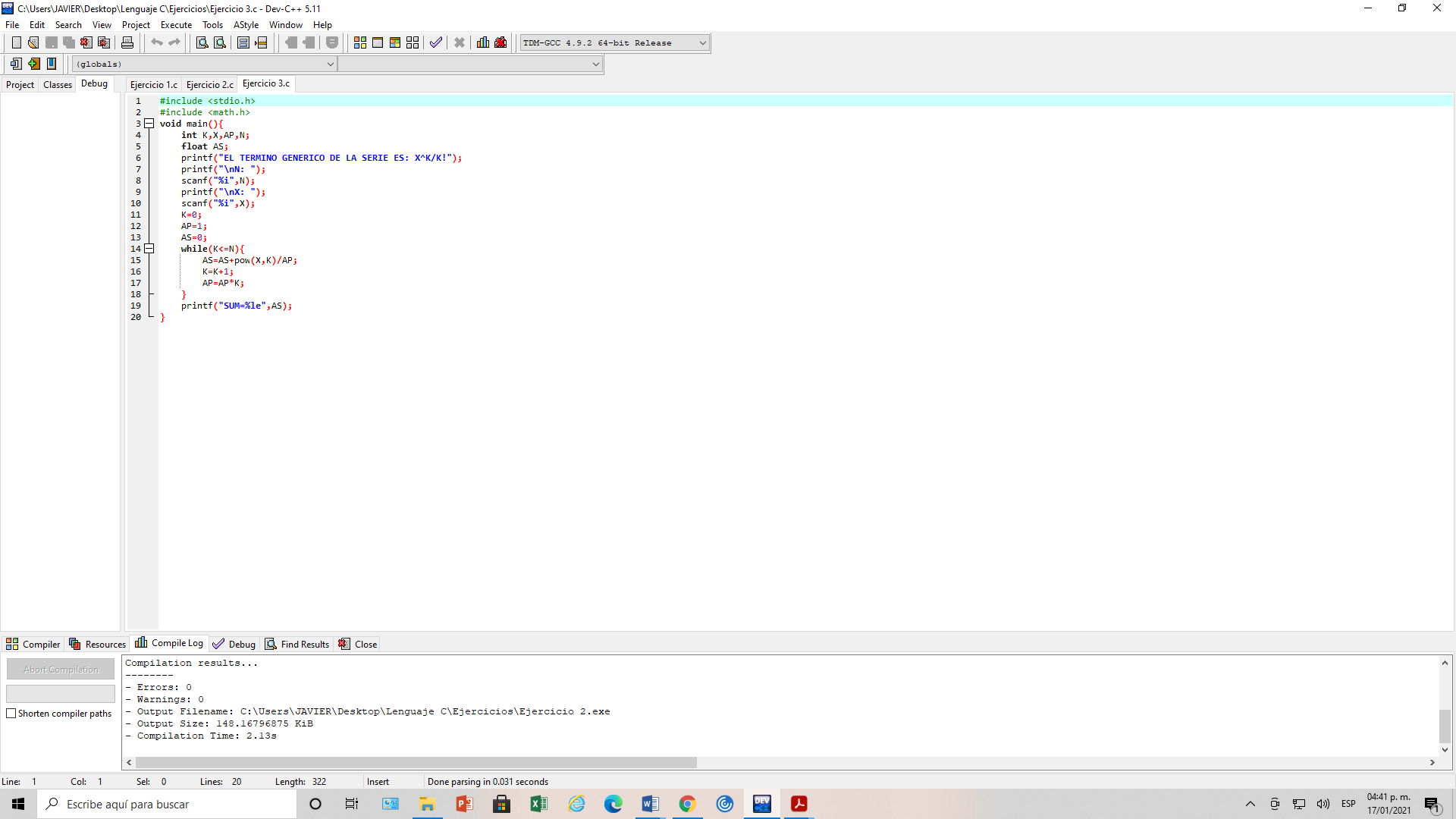




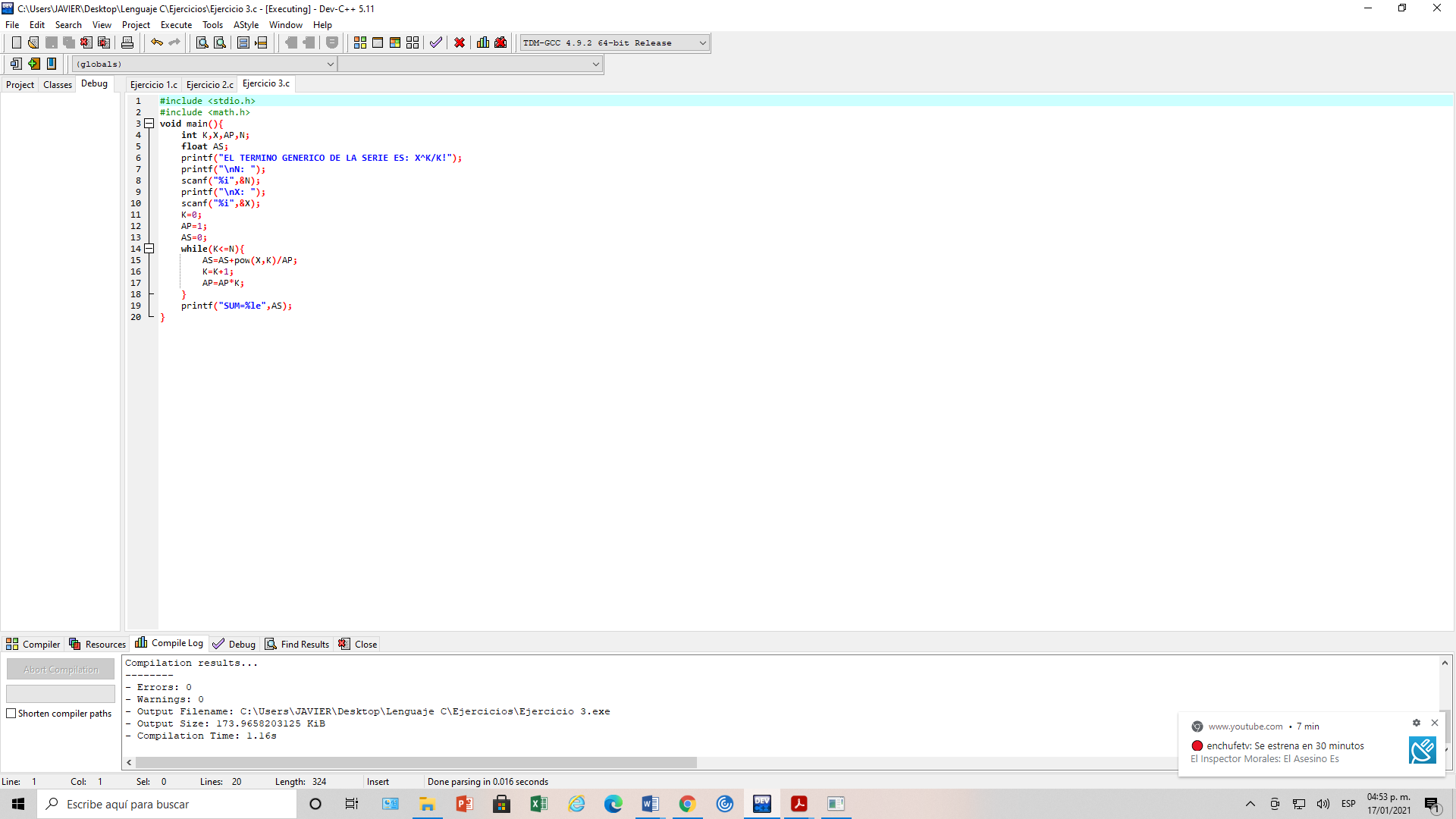


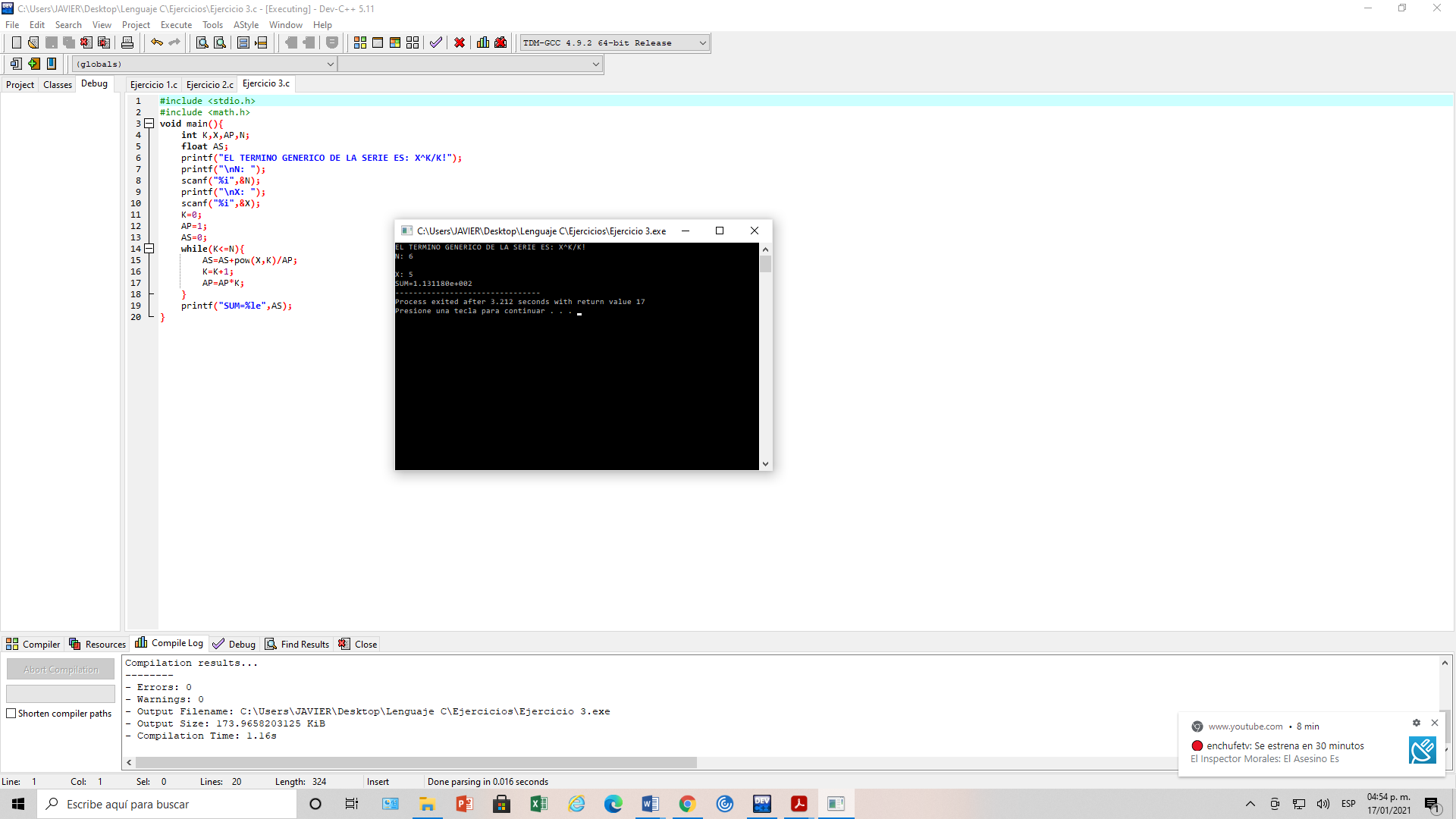


El siguiente programa muestra una violación de segmento durante su ejecución y se interrumpe; usa un depurador para detectar y corregir la falla.



Al programa le falta un símbolo fundamental que es el “&” en dos líneas de código para que el programa tenga su funcionalidad normal y guarde los datos que se está proporcionado al momento de ejecutarlo, quedando así:





**Conclusiones**

Conocer el funcionamiento de un depurador es fundamental para poder encontrar errores en grandes códigos fuentes rápidamente, además de optimizarlo para que funcione mejor que cuando solo se programa y no se revisa qué se puede quitar, agregar, reducir, etc., pero cabe decir que no es recomendado usarlo para programas con muy pocas líneas de código ya que a ojo se puede ver cuál es el error.