|  |
| --- |
| Laboratorio de Computación  Salas A y B |
|  |

|  |  |
| --- | --- |
| Profesor: | GARCIA MORALES KARINA ING. |
| Asignatura: | FUNDAMENTOS DE PROGRAMACION |
| Grupo: | 1121 |
| No de Práctica(s): | 10 |
| Integrante(s): | JOSE DANIEL CALLEJAS SANDOVAL |
|  |  |
|  |  |
|  |  |
| No. de Equipo de cómputo empleado: | 25 |
| Semestre: | 1 |
| Fecha de entrega: | 30-10-2018 |
| Observaciones: |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Guía práctica de estudio 10: 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.

**DESARROLLO**

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.

Hay 3 eventos que pueden llegar a suceder: **Error**. Se refiere a una acción humana que produce o genera un resultado incorrecto.

**Defecto (Fault).** Es la manifestación de un error en el software.

Falla (failure). Es una desviación del servicio o resultado esperado.

Punto de ruptura: también conocido por su traducción al inglés bre*akpoint, s*irve 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 repetir una sección del programa.

Para depurar un programa usando las herramientas desarrolladas por GNU, éste debe compilarse con información para depuración por medio del compilador GCC.

Una vez hecho el paso anterior, debe usarse la herramienta GDB, la cual, es el depurador para cualquier programa ejecutable realizado por GCC

La depuración de un programa es útil cuando:

Se desea optimizar el programa: no basta que el programa se pueda compilar y se

someta a pruebas que demuestren que funciona correctamente. Debe realizarse un

análisis exhaustivo del mismo en ejecución para averiguar cuál es su flujo de

operación y encontrar formas de mejorarlo (reducir el código, utilizar menos

recursos llegando a los mismos resultados, hacer menos rebuscado al algoritmo), o

bien, encontrar puntos donde puede fallar con ciertos tipos de entrada de datos.

El programa tiene algún fallo: el programa no muestra los resultados que se esperan

para cierta entrada de datos debido a que el programador cometió algún error

durante el proceso de diseño. 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: cuando el programa está

ejecutándose, éste se detiene inesperadamente. 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”.

Depuración de programas escritos en C con GCC y GDB

Para depurar un programa usando las herramientas desarrolladas por GNU, éste debe

compilarse con información para depuración por medio del compilador GCC.

Para compilar, por ejemplo, un programa llamado calculadora.c con GCC con información

de depuración, debe realizarse en una terminal con el siguiente comando:

gcc -g -o calculadora calculadora.c

El parámetro -g es quien indica que el ejecutable debe producirse con información de

depuración.

Una vez hecho el paso anterior, debe usarse la herramienta GDB, la cual, es el depurador

para cualquier programa ejecutable realizado por GCC.

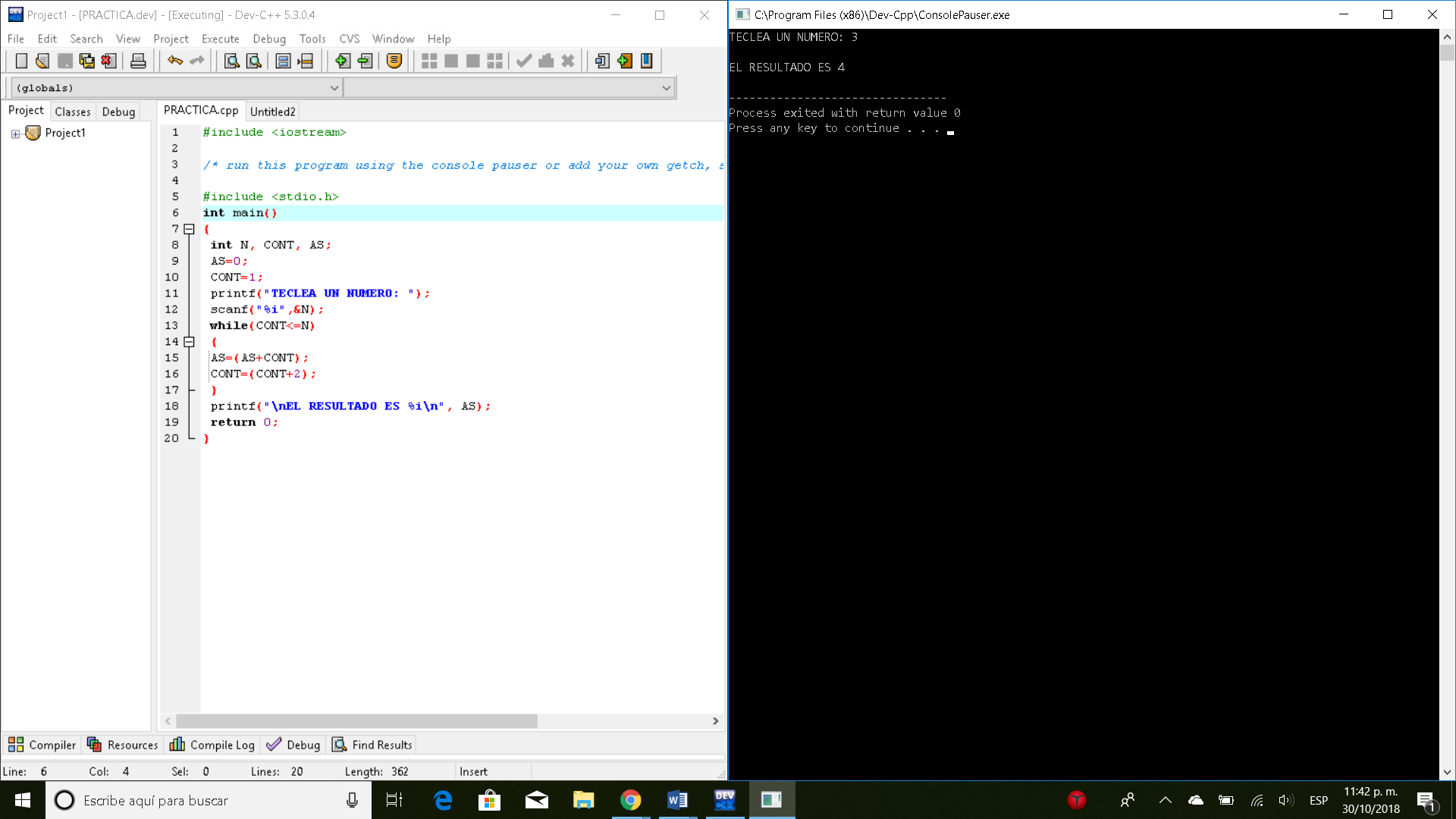
Para depurar un ejecutable debe invocarse a GDB en la terminal indicando cuál es el

programa ejecutable para depurar, por ejemplo, para depurar calculadora:

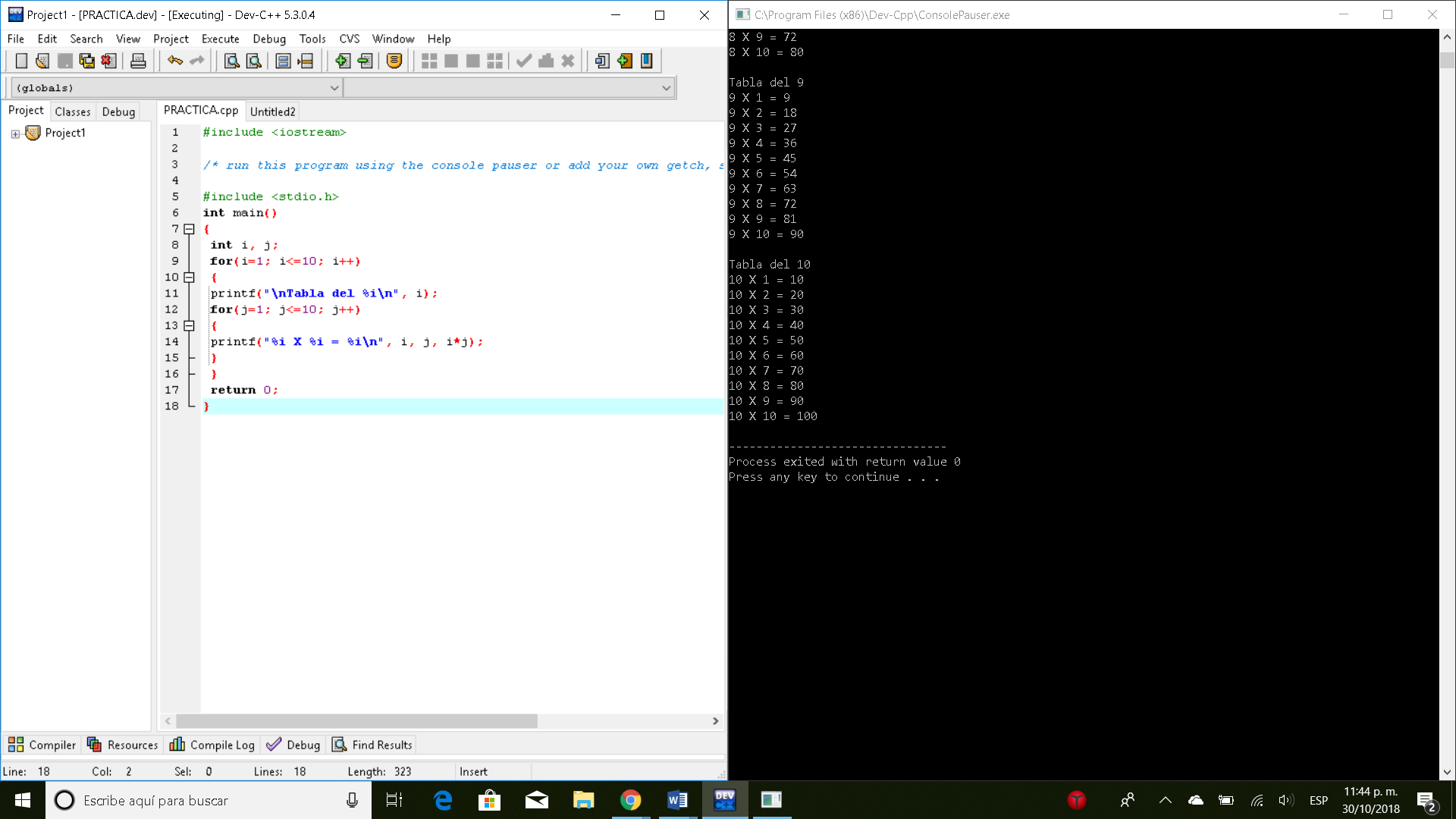
gdb ./calculadora

**TAREAS**

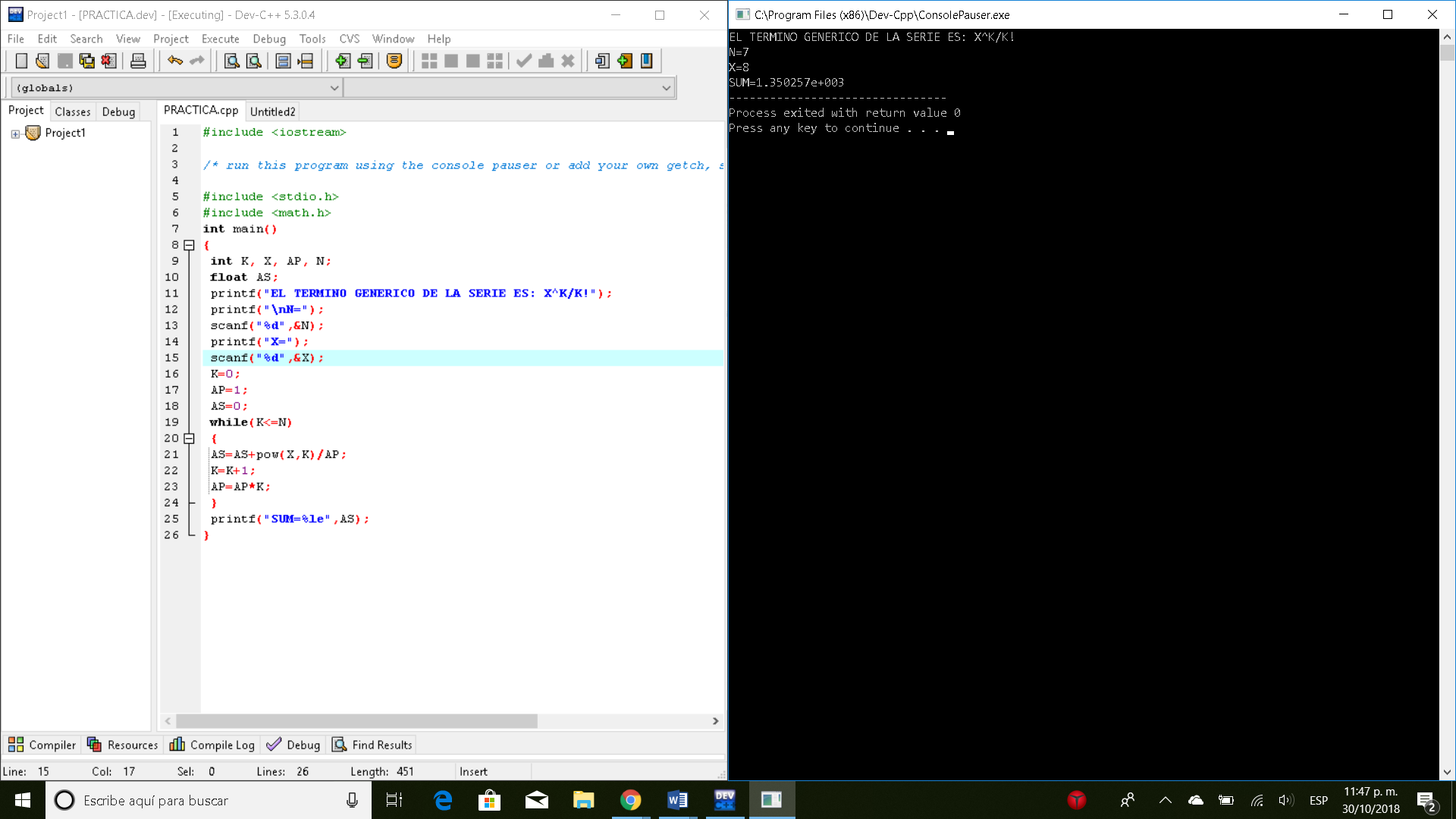
1. EL ERROR ERA QUE NO SE DECLARABA LA FUNCION PRINCIPAL Y HABIA UNA LLAVE ADICIONAL.



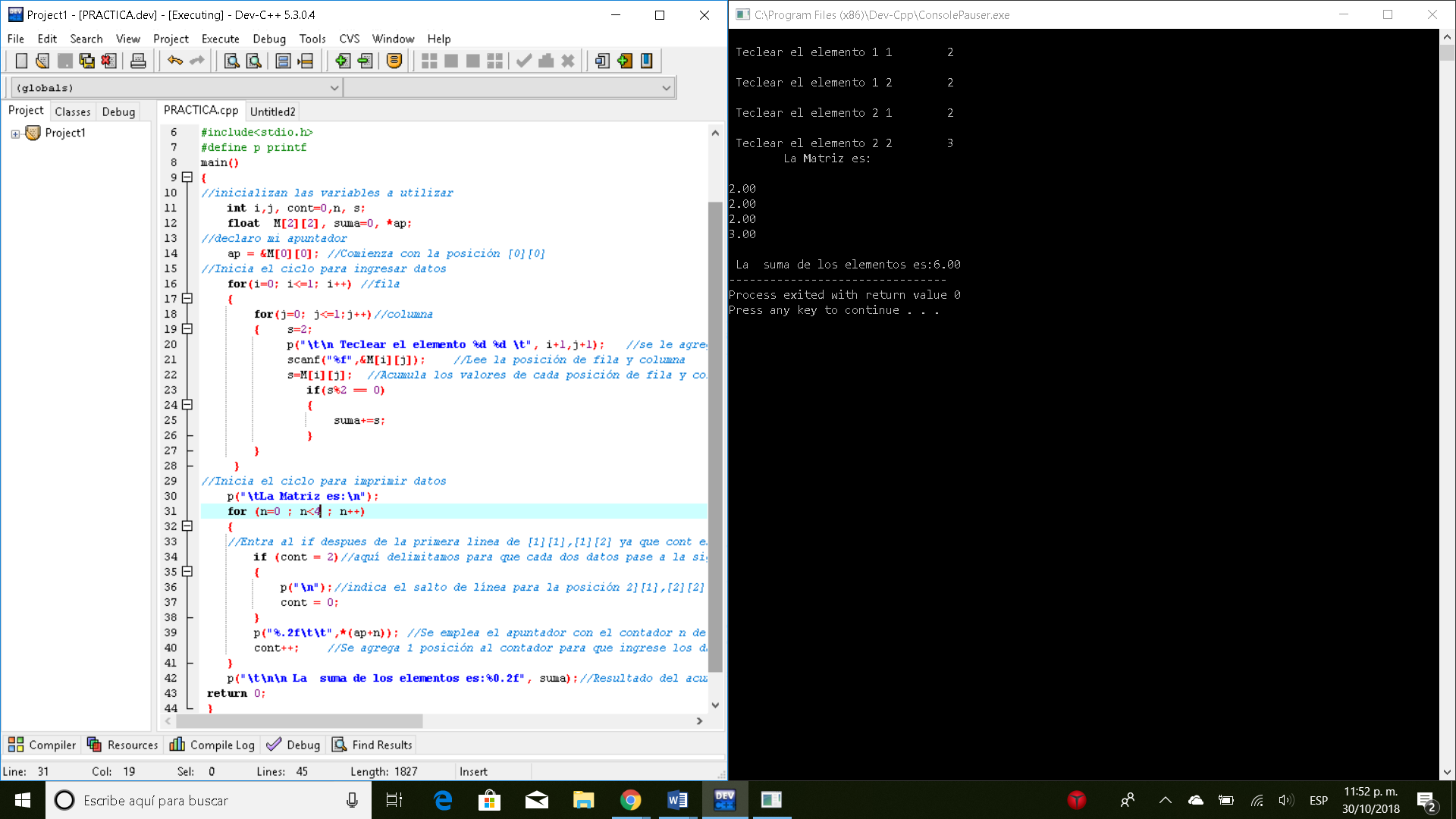
1. EL ERROR ERA QUE SE DEBIA COLOCAR <=10 EN LOS CICLOS Y HABIA UNA LLAVE ADICIONAL



1. EL ERROR ERA EL “&” EN LOS SCANF Y EN NO DECLARAR A LA FUNCION



1. EN LUGAR DE COLOCAR “9” COLOQUE “4” EN EL CICLO PARA QUE LOS DIGITOS FUERAN ENTEROS



**CONCLUSIONES**

Al finalizar la práctica, me percate de la importancia de conocer que es lo que realiza el programa y como poder identificar tanto errores humanos como computacionales para poder ejecutar un programa.

Vimos varios ejemplos, y esto ayudo a poder analizarlos con detenimiento y conocer sus características.

El ritmo de la practica fue el adecuado para poder aprender todos los temas.

**BIBLIOGRAFIA**

* Gutiérrez Rodríguez, Javier Jesús. Primeros pasos con GDB. Consulta: octubre de 2016. Disponible en: http://www.lsi.us.es/~javierj/ssoo\_ficheros/GuiaGDB.htm
* Ferreira, Amelia. Depurador gdb. Consulta: octubre de 2016. Disponible en: http://learnassembler.com/gdbesp.html
* Ferreira, Amelia. Depurador gdb - uso de la opción -g de gcc. Consulta: octubre de 2016. Disponible en: http://learnassembler.com/opc.html
* Gutiérrez, Erik Marín. Depuración de programas Dev C++. Consulta: octubre de 2016. Disponible en: http://programacionymetodos.blogspot.mx/2012/05/depuracionde-programas-dev-c.html
* González Cárdenas, Miguel Eduardo; Marín Lara, Claudia Lorena; Noguerón Pérez, Pedro. Apuntes De Computadoras Y Programación. Universidad Nacional Autónoma de México.
* Pozo Coronado, Salvador. Primeros pasos con GDB. Consulta: octubre de 2016. Disponible en: http://www.c.conclase.net/devcpp/?cap=depurar