|  |
| --- |
| Laboratorio de Computación  Salas A y B |
| |  |  | | --- | --- | | Profesor: | Claudia Rodríguez Espino | | Asignatura: | Fundamentos de Programación | | Grupo: | 1104 | | No de Práctica(s): | 10 | | Integrante(s): | Téllez Torres Lorena Alejandra | |  |  | |  |  | |  |  | | No. de Equipo de cómputo empleado: | 51 | | Semestre: | 1ª | | Fecha de entrega: |  | | Observaciones: |  | |  |  | |

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

**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.

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.
* 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.
* El programa tiene un error de ejecución o defecto: cuando el programa está

ejecutándose, éste se detiene inesperadamente.

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

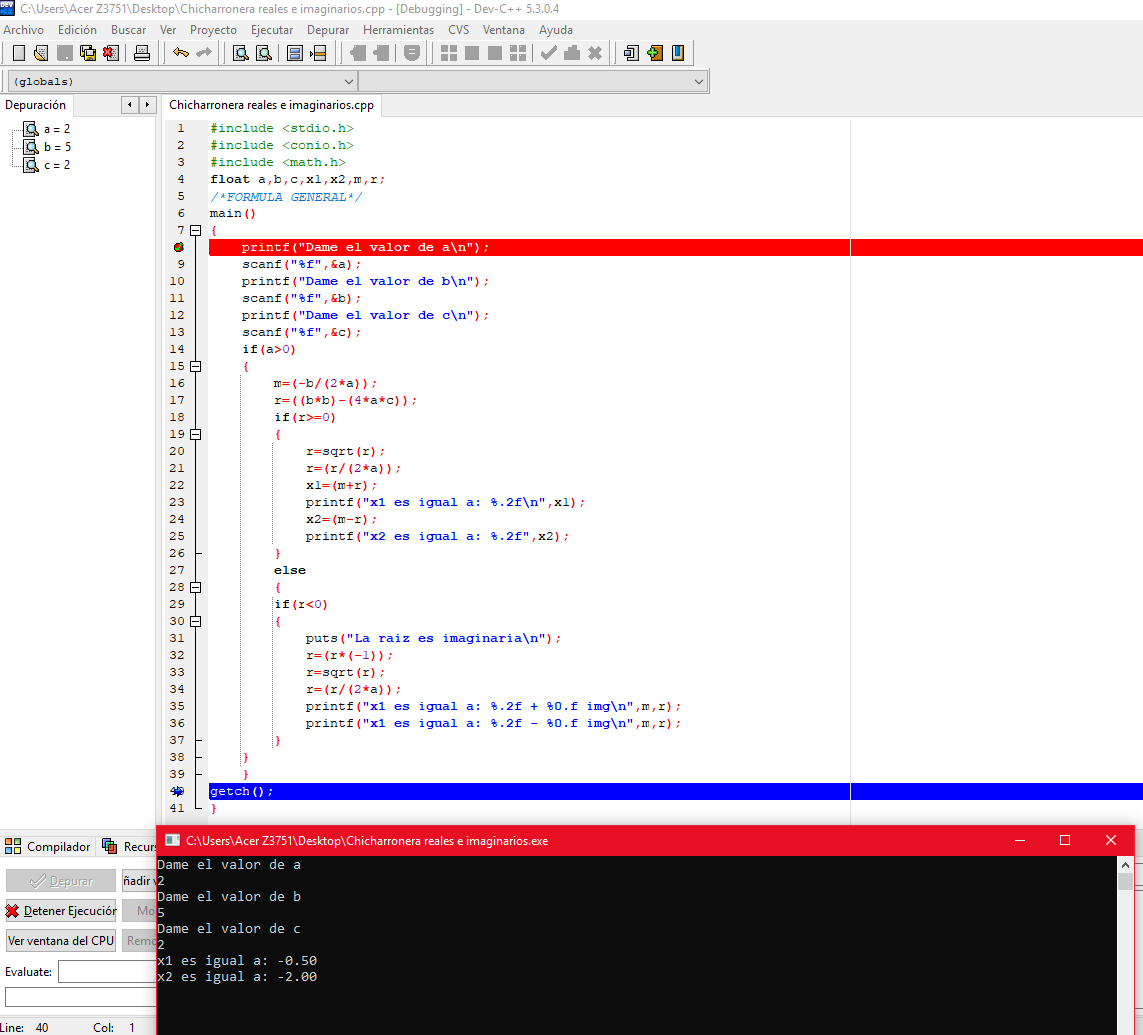
* Ejecutar el programa: se procede a ejecutar el programa en la herramienta de depuración ofreciendo diversas opciones para ello.
* Mostrar el código fuente del programa: muestra cuál fue el código fuente del

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.

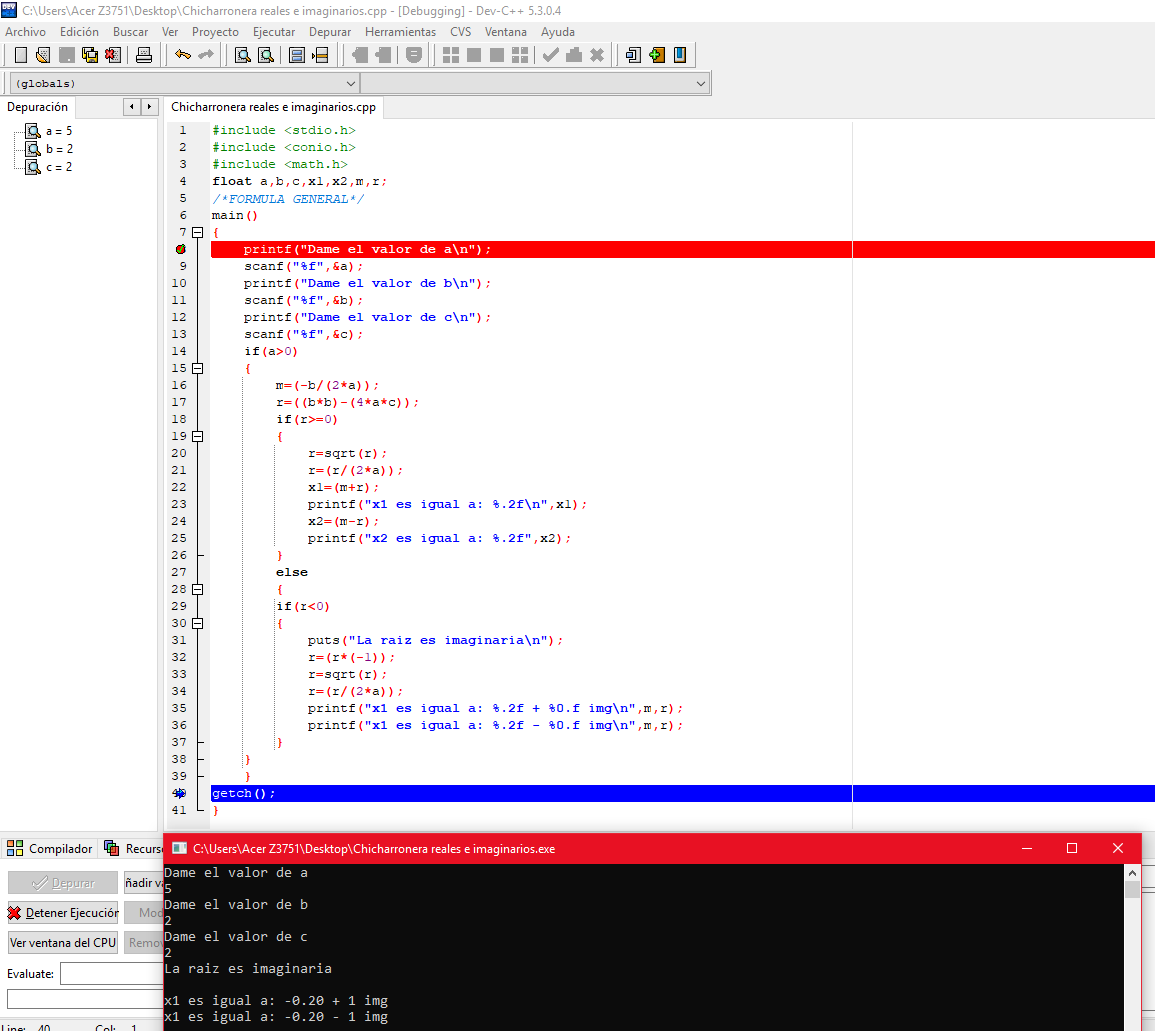
* Punto de ruptura: también conocido por su traducción al inglés breakpoint, sirve para detener la ejecución del programa en algún punto indicado previamente por medio del número de línea.
* Continuar: continúa con la ejecución del programa después del punto de ruptura.
* Ejecutar la siguiente instrucción: 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.
* 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.
* 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.

**Ejercicios**

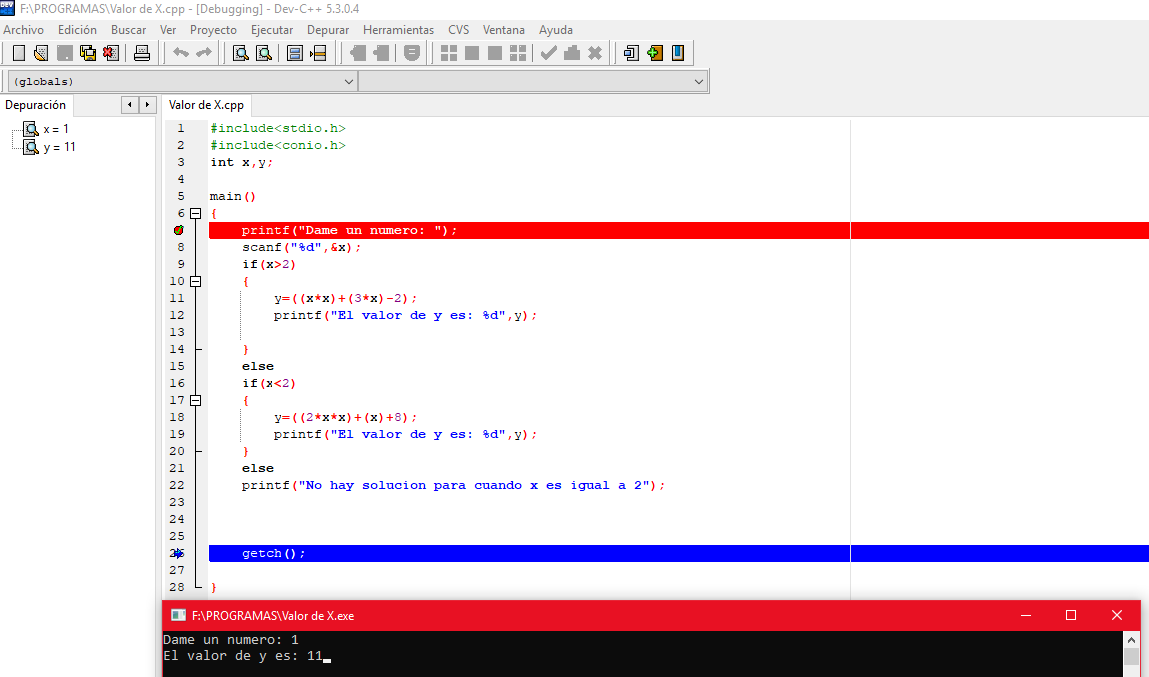
1. **Fórmula General (reales)**

****

1. **Fórmula General (imaginarios)**

****

1. **Ecuación de X (menor a 2)**

****

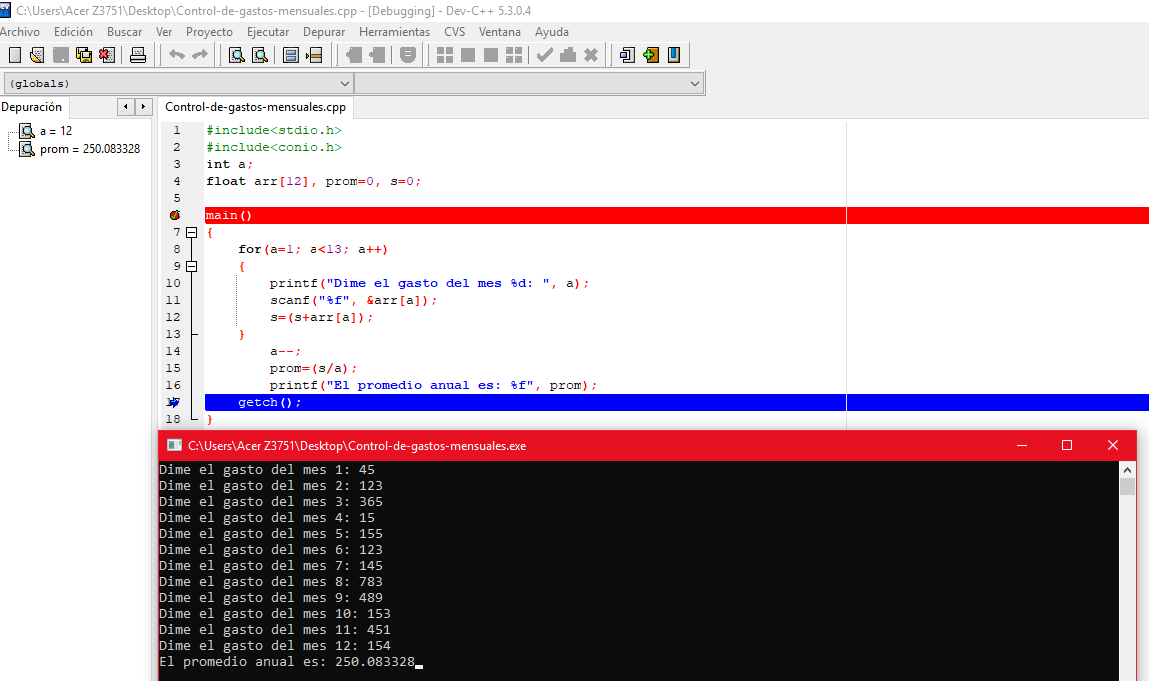
1. **Ecuación de X (igual a 2)**

****

1. **Ecuación de X (mayor a 2)**

****

1. **Gastos Mensuales**

****

**Conclusiones**

No sabía de la existencia de la función de la depuración, me parece muy útil en el caso de tener compilado un programa pero que no haga lo que se espera, hacer uso de la depuración para saber justamente que tienes que modificar para que tu programa no solo corra, sino que también cumpla con el objetivo propuesto desde el inicio, se me hace una gran herramienta de mucha ayuda.