	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: M.I. Marco Antonio Martínez Quintana

Asignatura: Fundamentos de Programación

Grupo: 3 Bloque: 136

No de Práctica(s): 10: Depuración de programas

Integrante(s): Carranza Ochoa José David

No. de Equipo de cómputo empleado: No aplica

No. de Lista o Brigada: 6

Semestre: 2021-1

Fecha de entrega: 04/01/2021

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.

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.

Desarrollo:

Al realizar la depuración de un programa es indispensable reconocer el entorno de trabajo donde este se ejecutará, por ello, los códigos escritos cuentan con diferentes herramientas para su depuración, tal es el caso del compilador GCC el cual cuenta con un depurador al ejecutar el comando "*gdb ./nombreArchivo*" en dicho depurador se puede realizar toda la serie de depuraciones con el fin de analizar el desarrollo del programa.

De la misma forma, existen otros compiladores que incluyen la depuración dentro de su sistema como Code::Block, XCode en Mac o el más utilizado DevC++.

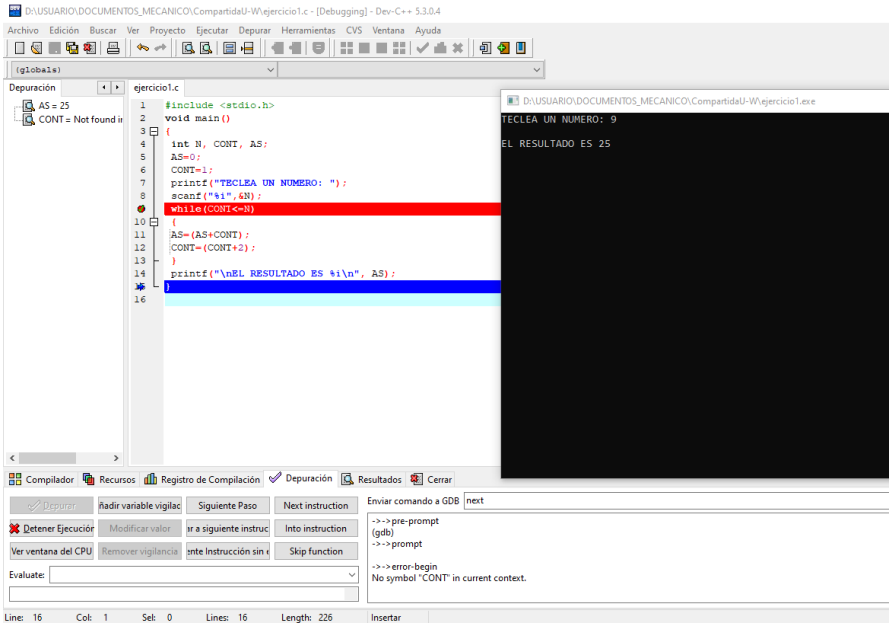
Para el desarrollo de la práctica se empleó DevC++ cuyo IDE es ameno y de la misma manera, retoma el compilador GCC y el depurador GDB.

Ahora, debemos tener en cuenta los pasos que se ejecutan en la depuración:

- Ejecución del programa
- Visualización del código fuente
- Punto de ruptura
- Avanzar en el desarrollo del código
- Ejecución de la siguiente instrucción
- Ejecución de la siguiente línea
- Ejecución de la instrucción o línea anterior
- Visualización de las variables

Ejercicio 1

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 puntos de ruptura, ejecución de siguiente línea o instrucción.



Cuenta con una funcionalidad tal es el caso que los números impares son divididos entre 2 y se elevan al cuadrado mientras que a los impares les ocurre lo mismo siempre y cuando antes se les sume el valor de 1

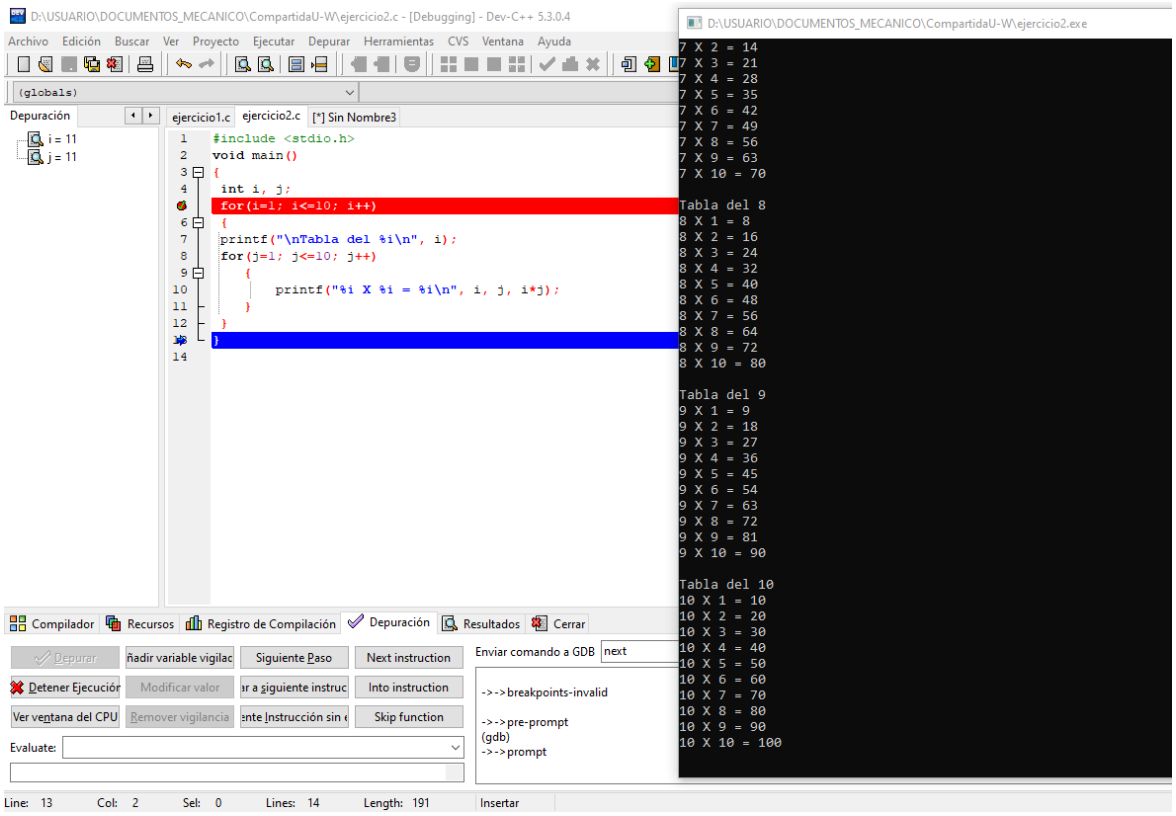
Ejercicio 2

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 las tablas. Usar un depurador de C para averiguar el funcionamiento del programa y corregir ambos problemas.

```c
#include <stdio.h>
void main()
{
 int i, j;
 for(i=1; i<10; i++)
 {
 printf("\nTabla del %i\n", i);
 for(j=1; j==10; j++)
 {
 printf("%i X %i = %i\n", i, j, i*j);
 }
 }
}
```

Al realizar la depuración correspondiente, se observa que la falla está contenida en los operadores de asignación del código; este indica a los valores menores mas no cuenta con aquel que sea igual a dicho valor. De la misma forma, el valor de "j" lo toma como contante mientras que debe ser variable con la misma regla anterior.

El código queda finalmente como la imagen a continuación.



Ejercicio 3

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

```
1   #include <stdio.h>
2   #include <math.h>
3   void main()
4   {
5    int K, X, AP, N;
6    float AS;
7    printf("EL TERMINO GENERICO DE LA SERIE ES: X^K/K!");
8    printf("\nN=");
     scanf("%d",N);
10   printf("X=");
     scanf("%d",X);
12   K=0;
13   AP=1;
14   AS=0;
15   while(K<=N)
16   {
17   AS=AS+pow(X,K)/AP;
18   K=K+1;
19   AP=AP*K;
20   }
21   printf("SUM=%le",AS);
22  }
```

Su falla se presenta al escanear el valor de las variables, esto se denota en el punto de ruptura donde el programa se detiene repentinamente al igual que al momento de agregar una variable de asignación para ver su desarrollo, esta no es reconocida por carecer de “&” en la sintaxis del código.

```c
#include <stdio.h>
#include <math.h>
void main()
{
 int K, X, AP, N;
 float AS;

 K=0;
 AP=1;
 AS=0;

 printf("EL TERMINO GENERICO DE LA SERIE ES: X^K/K!");
 printf("\nN=");
     scanf("%i",&N);
 printf("\nX=");
     scanf("%i",&X);

 while(K<=N)
 {
 AS=AS+pow(X,K)/AP;
 K=K+1;
 AP=AP*K;
 }
 printf("SUM=%le",AS);
}
```

```
D:\Programas\Dev-Cpp\ConsolePauser.exe
EL TERMINO GENERICO DE LA SERIE ES: X^K/K!
N=2

X=2
SUM=5.000000e+000
--------------------------------
Process exited with return value 17
Press any key to continue . . .
```

Conclusiones

La depuración es una excelente herramienta para el desarrollo óptimo de Software, aprendí a manejarla por medio de un IDE el cual desconocía hasta esta práctica; de la misma forma, manejé los pasos que ocurren durante la depuración notando los errores que se presentan humanamente y que a simple vista no son reconocidos a menos que se retome uso de un buen interprete. Para estos casos sencillos se diría que es fácil detectar los errores, sin embargo, para códigos más extensos el revisar línea por línea tomaría bastante tiempo al igual que se pasarían errores por alto.

En dicha práctica, observé como las variables cambian de valores en el transcurso del código, es por ello que la depuración facilita incluso la manera de entender un código al comprender la funcionalidad del mismo.

Bibliografía

♣ 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