



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: M.I. MARCO ANTONIO MARTINEZ QUINTANA

Asignatura: FUNDAMENTOS DE PROGRAMACIÓN

Grupo: 3

No de Práctica(s): 10

Integrante(s): CARRILLO CERVANTES IVETTE ALEJANDRA

*No. de Equipo de
cómputo empleado:* NO APLICA

No. de Lista o Brigada: 7

Semestre: PRIMER SEMESTRE

Fecha de entrega: 07 DICIEMBRE 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.

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. Un defecto es encontrado porque causa una Falla (failure).

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

La depuración de un programa es útil, cuando: se desea optimizar el programa, el programa tiene algún fallo, el programa tiene un error de ejecución o defecto.

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

- Ejecutar el programa.
- Mostrar el código fuente del programa.
- Punto de ruptura.
- Continua.
- Ejecutar la siguiente instrucción.
- Ejecutar la siguiente línea.
- Ejecutar la instrucción o línea anterior.
- Visualizar el valor de las variables.

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

Depuración de programas escritos en C con GCC y GDB

- ✓ **list o l:** Permite listar diez líneas del código fuente del programa, si se desea visualizar todo el código fuente debe invocarse varias veces este comando para mostrar de diez en diez líneas. Se puede optar por colocar un número separado por un espacio para indicar a partir de qué línea desea mostrarse el programa. También es posible mostrar un rango de líneas introduciendo el comando y de qué línea a qué línea separadas por una coma.

Ejemplo: list 4,6

- ✓ **b:** Establece un punto de ruptura para lo cual debe indicarse en qué línea se desea establecer o bien también acepta el nombre de la función donde se desea realizar dicho paso.

Ejemplo: b 5

- ✓ `d` o `delete`: Elimina un punto de ruptura, indicando cuál es el que debe eliminarse usando el número de línea.

Ejemplo: `d 5`

- ✓ `clear`: Elimina todos los puntos de ruptura.

Ejemplo: `clear`

- ✓ `info line`: Permite mostrar información relativa a la línea que se indique después del comando.

Ejemplo: `info line 8`

- ✓ `run` o `r`: Ejecuta el programa en cuestión. Si el programa tiene un punto de ruptura se ejecutará hasta dicho punto, de lo contrario se ejecutará todo el programa.

- ✓ `c`: Continúa con la ejecución del programa después de un punto de ruptura.

- ✓ `s`: Continúa con la siguiente instrucción después de un punto de ruptura.

- ✓ `n`: Salta hasta la siguiente línea de código después de un punto de ruptura.

- ✓ `p` o `print`: Muestra el valor de una variable, para ello debe escribirse el comando y el nombre de la variable separados por un espacio.

Ejemplo: `p suma_acumulada`

- ✓ `ignore`: Ignora un determinado punto de ruptura indicándolo con el número de línea de código.

Ejemplo: `ignore 5`

- ✓ `q` o `quit`: Termina la ejecución de GDB.

Actividades:

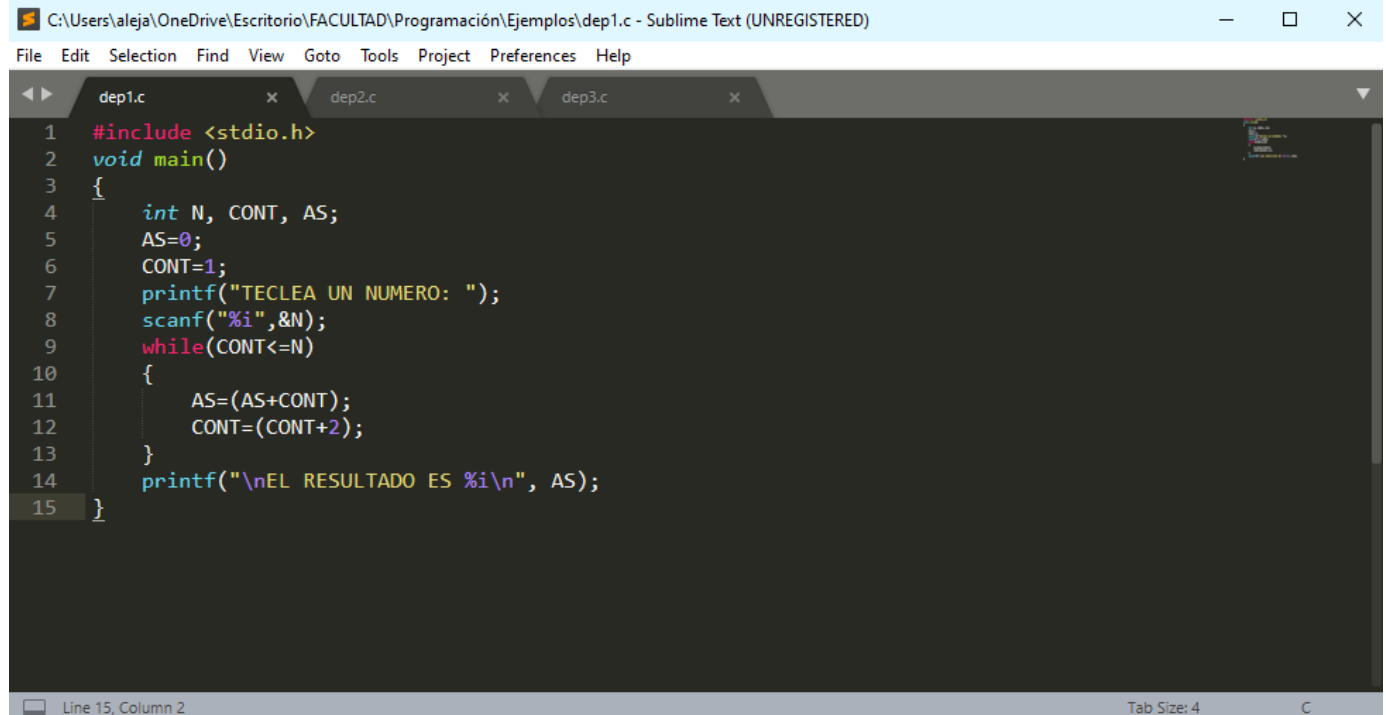
- Revisar, a través de un depurador, los valores que va tomando una variable en un programa escrito en C, al momento de ejecutarse.
- Utilizando un depurador, revisar el flujo de instrucciones que se están ejecutando en un programa en C, cuando el flujo depende de los datos de entrada.

Ejercicios Propuestos

Para los ejercicios hechos en esta práctica se utilizará depuración de programas escritos en C en GCC y GDB

- 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.

dep1.c

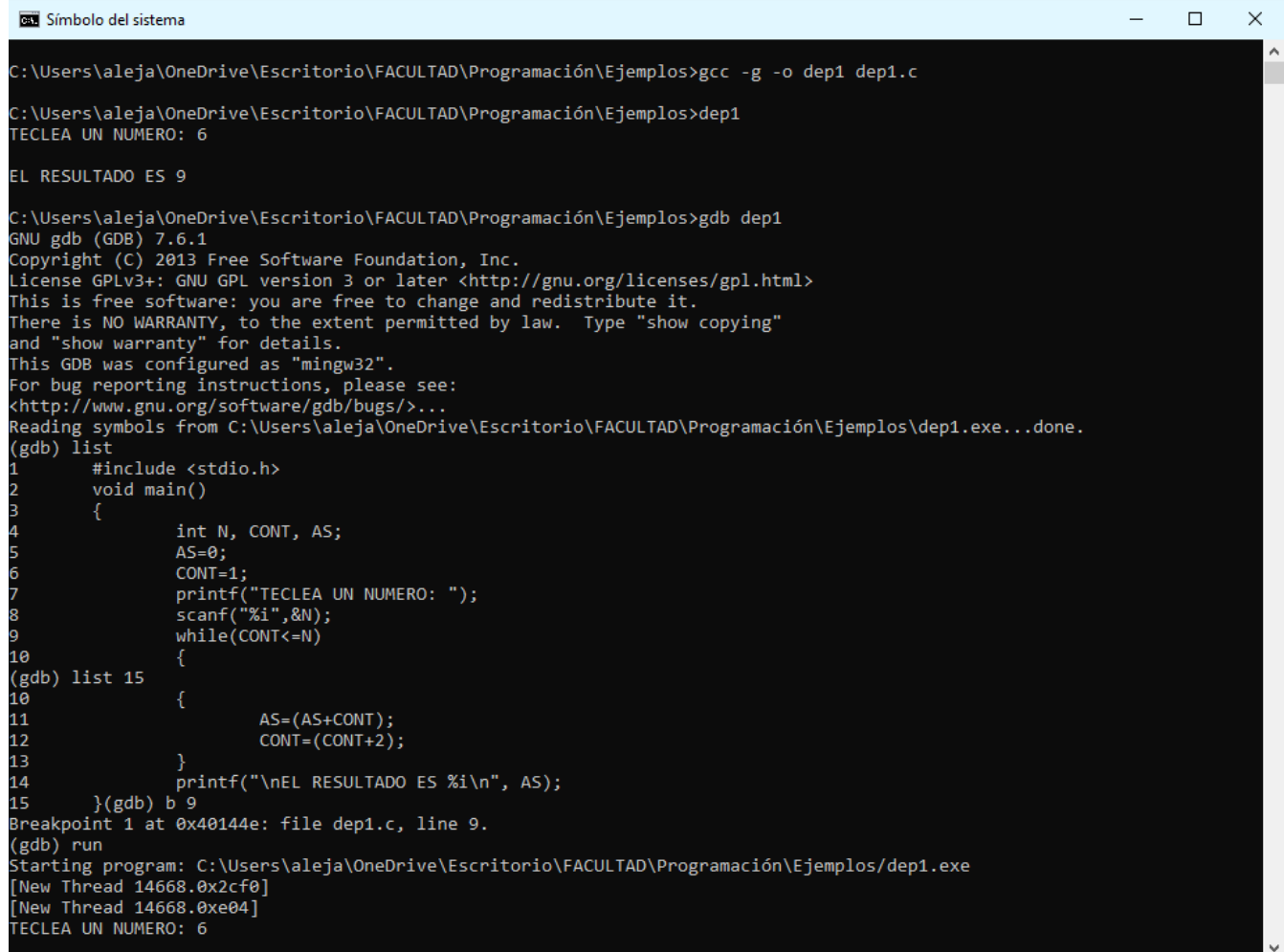


```

1  #include <stdio.h>
2  void main()
3  {
4      int N, CONT, AS;
5      AS=0;
6      CONT=1;
7      printf("TECLEA UN NUMERO: ");
8      scanf("%i",&N);
9      while(CONT<=N)
10     {
11         AS=(AS+CONT);
12         CONT=(CONT+2);
13     }
14     printf("\nEL RESULTADO ES %i\n", AS);
15 }

```

La función que tiene este programa es hacer una operación matemática, primero te pide un número N, el cual debe ser mayor o igual a 1, para después así poder realizar la operación.



```

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc -g -o dep1 dep1.c

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>dep1
TECLEA UN NUMERO: 6

EL RESULTADO ES 9

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gdb dep1
GNU gdb (GDB) 7.6.1
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "mingw32".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\dep1.exe...done.
(gdb) list
1      #include <stdio.h>
2      void main()
3      {
4          int N, CONT, AS;
5          AS=0;
6          CONT=1;
7          printf("TECLEA UN NUMERO: ");
8          scanf("%i",&N);
9          while(CONT<=N)
10         {
(gdb) list 15
10         {
11             AS=(AS+CONT);
12             CONT=(CONT+2);
13         }
14         printf("\nEL RESULTADO ES %i\n", AS);
15     }(gdb) b 9
Breakpoint 1 at 0x40144e: file dep1.c, line 9.
(gdb) run
Starting program: C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\dep1.exe
[New Thread 14668.0x2cf0]
[New Thread 14668.0xe04]
TECLEA UN NUMERO: 6

```

```

Breakpoint 1, main () at dep1.c:9
9          while(CONT<=N)
(gdb) c
Continuing.

EL RESULTADO ES 9
[Inferior 1 (process 14668) exited with code 023]
(gdb) info line 9
Line 9 of "dep1.c" starts at address 0x40144e <main+62> and ends at 0x401450 <main+64>.
(gdb) clear
Deleted breakpoint 1
(gdb) quit

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>

```

Al momento de ejecutarlo desde la terminal, usando gdb, utilice el comando “list” para ver el código del programa; después, utilice un punto de ruptura en la línea 9, donde se encuentra “while”, al ejecutar el programa con ese punto de ruptura, se para el programa justo en la línea 9; pero al utilizar el comando “c”, continua el programa partiendo ahora desde el punto de ruptura hasta finalizarlo.

En este programa también vi el funcionamiento del comando “clear” el cual borra todos los puntos de ruptura hechos anteriormente, y también el comando “quit” para salirme de gdb.

- 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

dep2.c

```

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\dep2.c - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

dep1.c x dep2.c x dep3.c x
1  #include <stdio.h>
2  void main()
3  {
4      int i, j;
5      for(i=1; i<10; i++)
6      {
7          printf("\nTabla del %i\n", i);
8          for(j=1; j==10; j++)
9          {
10             printf("%i X %i = %i\n", i, j, i*j);
11         }
12     }
13 }

```

Line 13, Column 2; Saved C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\dep2.c (UTF-8) Tab Size: 4 C

La función que tiene este programa, como ya se dijo en las instrucciones, es mostrar cada tabla de multiplicar; sin embargo, al ejecutar el programa solo aparece el nombre de cada una de las tablas ej “Tabla del 2” y no se muestra ninguna operación.

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc -g -o dep2 dep2.c
```

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>dep2
```

```
Tabla del 1
```

```
Tabla del 2
```

```
Tabla del 3
```

```
Tabla del 4
```

```
Tabla del 5
```

```
Tabla del 6
```

```
Tabla del 7
```

```
Tabla del 8
```

```
Tabla del 9
```

```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gdb dep2
```

```
GNU gdb (GDB) 7.6.1
```

```
Copyright (C) 2013 Free Software Foundation, Inc.
```

```
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
```

```
This is free software: you are free to change and redistribute it.
```

```
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
```

```
and "show warranty" for details.
```

```
This GDB was configured as "mingw32".
```

```
For bug reporting instructions, please see:
```

```
<http://www.gnu.org/software/gdb/bugs/>...
```

```
Reading symbols from C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\dep2.exe...done.
```

```
(gdb) list
```

```
1      #include <stdio.h>
2      void main()
3      {
4          int i, j;
5          for(i=1; i<10; i++)
6          {
7              printf("\nTabla del %i\n", i);
8              for(j=1; j==10; j++)
9              {
10                 printf("%i X %i = %i\n", i, j, i*j);
```

```
(gdb) list 16
```

```
11                 }
12             }
13         }(gdb) b 5
```

```
Breakpoint 1 at 0x40141e: file dep2.c, line 5.
```

```
(gdb) b 8
```

```
Breakpoint 2 at 0x40143c: file dep2.c, line 8.
```

```
(gdb) run
```

```
Starting program: C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\dep2.exe
```

```
[New Thread 8140.0x38dc]
```

```
[New Thread 8140.0x25a4]
```

```
Breakpoint 1, main () at dep2.c:5
```

```
5          for(i=1; i<10; i++)
```

```
(gdb) c
```

```
Continuing.
```

```
Tabla del 1
```

```
Breakpoint 2, main () at dep2.c:8
```

```
8          for(j=1; j==10; j++)
```

```
(gdb) c
```

```
Continuing.
```

```
Tabla del 2
```

```
Breakpoint 2, main () at dep2.c:8
```

```
8          for(j=1; j==10; j++)
```

```
(gdb) c
```

```
Continuing.
```

```
Tabla del 3
```

```
Breakpoint 2, main () at dep2.c:8
```

```
8          for(j=1; j==10; j++)
```

```
(gdb) c
```

```
Continuing.
```

```
Tabla del 4
```

```
Breakpoint 2, main () at dep2.c:8
```

```
8          for(j=1; j==10; j++)
```

```
(gdb) c
```

```

Continuing.
Tabla del 5
Breakpoint 2, main () at dep2.c:8
8      for(j=1; j==10; j++)
(gdb) s
5      for(i=1; i<10; i++)
(gdb) s
7      printf("\nTabla del %i\n", i);
(gdb) s
Tabla del 6
Breakpoint 2, main () at dep2.c:8
8      for(j=1; j==10; j++)
(gdb) s
5      for(i=1; i<10; i++)
(gdb) s
7      printf("\nTabla del %i\n", i);
(gdb) c
Continuing.
Tabla del 7
Breakpoint 2, main () at dep2.c:8
8      for(j=1; j==10; j++)
(gdb) c
Continuing.
Tabla del 8
Breakpoint 2, main () at dep2.c:8
8      for(j=1; j==10; j++)
(gdb) c
Continuing.
Tabla del 9
Breakpoint 2, main () at dep2.c:8
8      for(j=1; j==10; j++)
(gdb) c
Continuing.
[Inferior 1 (process 8140) exited with code 015]

```

Al momento de ejecutarlo desde la terminal, usando gdb, utilice el comando “list” para ver el código del programa; después, utilice un punto de ruptura en la línea 5, correspondiente a “for”, y otro en la línea 8 correspondiente otra vez a otra acción “for”, al ejecutar el programa con estos puntos de ruptura, primero el programa se detiene en la línea 5, al continuar se detiene en la línea 8, al continuar me di cuenta que tenia un fallo en for, así que lo cambie, los cambios que hice fueron:

En la línea 5

```
for(i=1; <10; i++)
```

lo cambie por:

```
for(i=1; <=10; i++)
```

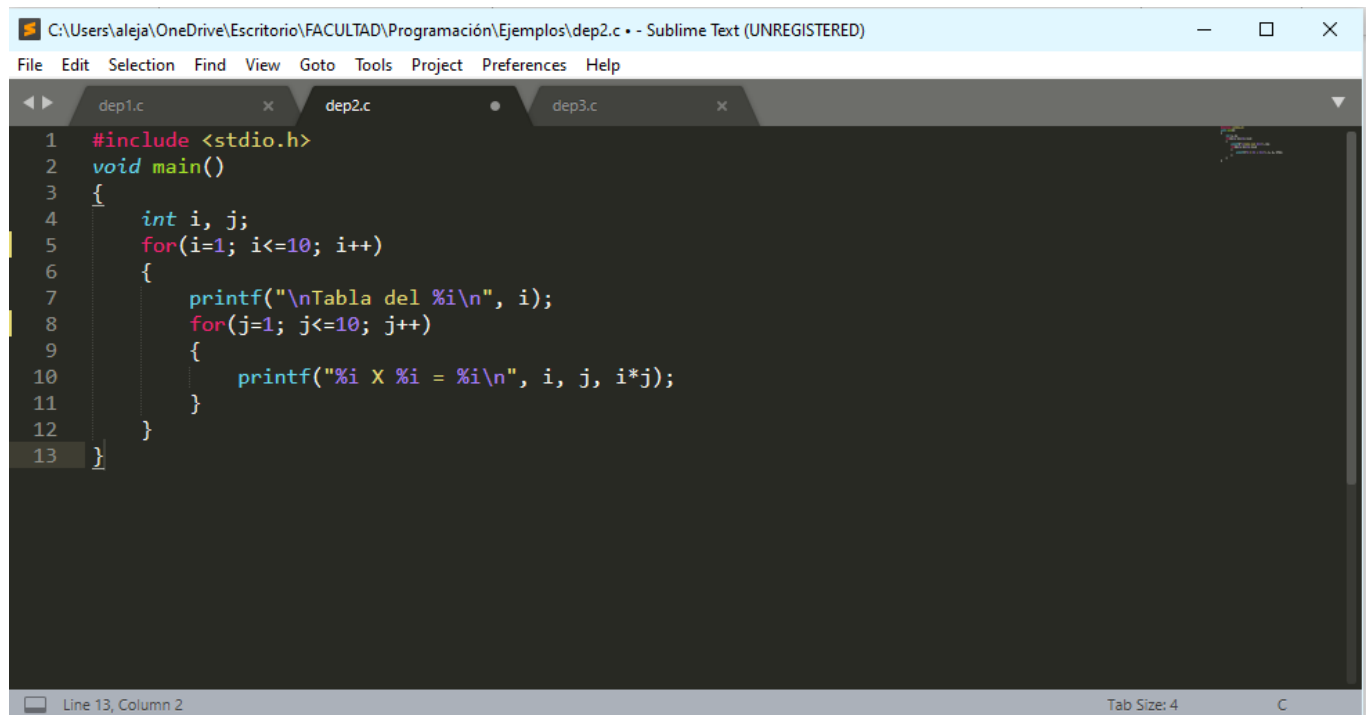
En la línea 8

```
for(j=1; j==10; j++)
```

lo cambie por

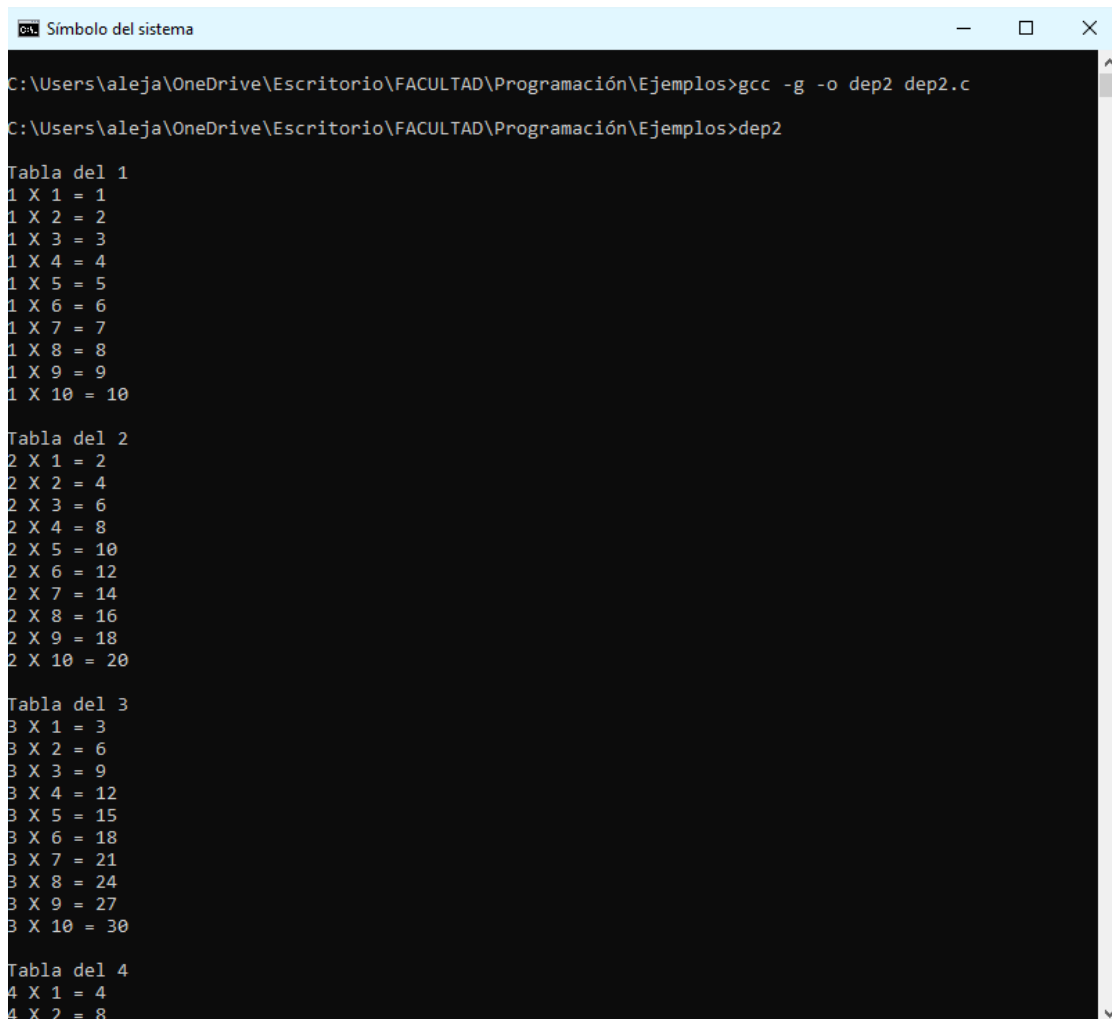
```
for(j=1; j<=10; j++)
```

Con estos cambios, el código del programa quedo de la siguiente manera:



```
1 #include <stdio.h>
2 void main()
3 {
4     int i, j;
5     for(i=1; i<=10; i++)
6     {
7         printf("\nTabla del %i\n", i);
8         for(j=1; j<=10; j++)
9         {
10             printf("%i X %i = %i\n", i, j, i*j);
11         }
12     }
13 }
```

Al volver a ejecutar ya nos mostraba cada una de las tablas de multiplicar



```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc -g -o dep2 dep2.c
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>dep2

Tabla del 1
1 X 1 = 1
1 X 2 = 2
1 X 3 = 3
1 X 4 = 4
1 X 5 = 5
1 X 6 = 6
1 X 7 = 7
1 X 8 = 8
1 X 9 = 9
1 X 10 = 10

Tabla del 2
2 X 1 = 2
2 X 2 = 4
2 X 3 = 6
2 X 4 = 8
2 X 5 = 10
2 X 6 = 12
2 X 7 = 14
2 X 8 = 16
2 X 9 = 18
2 X 10 = 20

Tabla del 3
3 X 1 = 3
3 X 2 = 6
3 X 3 = 9
3 X 4 = 12
3 X 5 = 15
3 X 6 = 18
3 X 7 = 21
3 X 8 = 24
3 X 9 = 27
3 X 10 = 30

Tabla del 4
4 X 1 = 4
4 X 2 = 8
```



```
4 X 3 = 12
4 X 4 = 16
4 X 5 = 20
4 X 6 = 24
4 X 7 = 28
4 X 8 = 32
4 X 9 = 36
4 X 10 = 40
```

Tabla del 5

```
5 X 1 = 5
5 X 2 = 10
5 X 3 = 15
5 X 4 = 20
5 X 5 = 25
5 X 6 = 30
5 X 7 = 35
5 X 8 = 40
5 X 9 = 45
5 X 10 = 50
```

Tabla del 6

```
6 X 1 = 6
6 X 2 = 12
6 X 3 = 18
6 X 4 = 24
6 X 5 = 30
6 X 6 = 36
6 X 7 = 42
6 X 8 = 48
6 X 9 = 54
6 X 10 = 60
```

Tabla del 7

```
7 X 1 = 7
7 X 2 = 14
7 X 3 = 21
7 X 4 = 28
7 X 5 = 35
7 X 6 = 42
7 X 7 = 49
7 X 8 = 56
7 X 9 = 63
7 X 10 = 70
```

Tabla del 8

```
8 X 1 = 8
8 X 2 = 16
8 X 3 = 24
8 X 4 = 32
8 X 5 = 40
8 X 6 = 48
8 X 7 = 56
8 X 8 = 64
8 X 9 = 72
8 X 10 = 80
```

Tabla del 9

```
9 X 1 = 9
9 X 2 = 18
9 X 3 = 27
9 X 4 = 36
9 X 5 = 45
9 X 6 = 54
9 X 7 = 63
9 X 8 = 72
9 X 9 = 81
9 X 10 = 90
```

Tabla del 10

```
10 X 1 = 10
10 X 2 = 20
10 X 3 = 30
10 X 4 = 40
10 X 5 = 50
10 X 6 = 60
10 X 7 = 70
10 X 8 = 80
10 X 9 = 90
10 X 10 = 100
```

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>

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

Dep3.c

```

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=");
9      scanf("%d",N);
10     printf("X=");
11     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 }

```

En este programa, se realizará una operación matemática con el fin de mostrarte como resultado, el número genérico de una serie; sin embargo, al ejecutarlo, solo pedía el valor de una sola de las variables del programa, cuando se debe de pedir 2.

```

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc -g -o dep3 dep3.c
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>dep3
EL TERMINO GENERICO DE LA SERIE ES: X^K/K!
N=6
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gdb dep3
GNU gdb (GDB) 7.6.1
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "mingw32".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\dep3.exe...
done.
(gdb) list
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=");
9          scanf("%d",N);
10         printf("X=");
11         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     }
(gdb) b 7

```

```

Breakpoint 1 at 0x40141e: file dep3.c, line 7.
(gdb) b 11
Breakpoint 2 at 0x401456: file dep3.c, line 11.
(gdb) b 15
Breakpoint 3 at 0x401480: file dep3.c, line 15.
(gdb) run
Starting program: C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\dep3.exe
[New Thread 2140.0x41c8]
[New Thread 2140.0x1190]

Breakpoint 1, main () at dep3.c:7
7      printf("EL TERMINO GENERICO DE LA SERIE ES: X^K/K!");
(gdb) c
Continuing.
EL TERMINO GENERICO DE LA SERIE ES: X^K/K!
N=6

Program received signal SIGSEGV, Segmentation fault.
0x767a040c in ungetc () from C:\windows\SysWOW64\msvcrt.dll
(gdb) c
Continuing.

Program received signal SIGSEGV, Segmentation fault.
0x767a040c in ungetc () from C:\windows\SysWOW64\msvcrt.dll
(gdb) c
Continuing.
[Inferior 1 (process 2140) exited with code 030000000005]
(gdb) clear
No source file specified.
(gdb) q

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>_

```

En este programa utilice 3 puntos de ruptura, en la línea 7 correspondiente a una impresión “printf”, en la línea 11 correspondiente a “scanf”, en la línea 15 correspondiente a otro “scanf”, al hacer la ejecución con esos 3 puntos de ruptura, y tomando en cuenta tiene una violación de segmento, me di cuenta en el los “scanf” falto agregar “&”, así que lo coregí

En la línea 9
scanf(“%d”, N);
lo cambie por:
scanf(“%d”, &N);

En la línea 11
scanf(“%d”, X);
lo cambie por:
scanf(“%d”, &X);

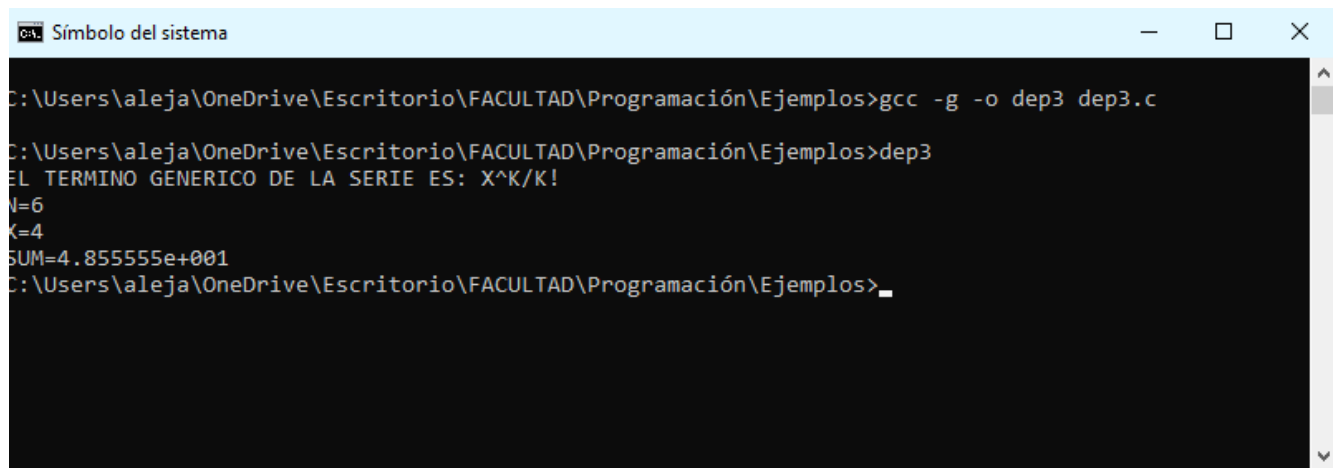
```

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos\dep3.c - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

dep1.c dep2.c dep3.c
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=");
9      scanf("%d",&N);
10     printf("X=");
11     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 }

```

Line 22, Column 2 Tab Size: 4 C



```
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>gcc -g -o dep3 dep3.c

C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>dep3
EL TERMINO GENERICO DE LA SERIE ES: X^K/K!
N=6
K=4
SUM=4.855555e+001
C:\Users\aleja\OneDrive\Escritorio\FACULTAD\Programación\Ejemplos>_
```

Al ejecutar el código nuevamente, ya te pedía el valor de K y N, después ya salía el resultado

Conclusiones

Esta práctica me costó trabajo de entenderla, pero al momento de entender para que eran los puntos de ruptura de un programa, fue sencillo el manejo de estos en gcc desde la terminal de la computadora. Me gustó mucho esta práctica, es de gran utilidad depurar los programas por que así, si tiene una falla, la puedes encontrar fácilmente.

Referencias

- Manual de prácticas del Laboratorio de Fundamentos de programación. 08 Diciembre 2020, de Facultad de Ingeniería Sitio web: <http://lcp02.fi-b.unam.mx/>