



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: Alejandro Pimentel Alarcón

Asignatura: Fundamentos de programación

Grupo: Bloque 135

No de Práctica(s): Práctica 10

Integrante(s): Partida Arias Emily Rachel

No. de Lista o Brigada: 41

Semestre: 2020-1

Fecha de entrega: Lunes 28 de Octubre

Observaciones: Muy bien

CALIFICACIÓN: 10

Práctica 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.

Introducción:

¿Qué es GDB?

GDB o GNU Debugger es el depurador estándar para el compilador GNU.

Es un depurador portable que se puede utilizar en varias plataformas Unix y funciona para varios lenguajes de programación como C, C++ y Fortran. GDB fue escrito por Richard Stallman en 1986. GDB es software libre distribuido bajo la licencia GPL.

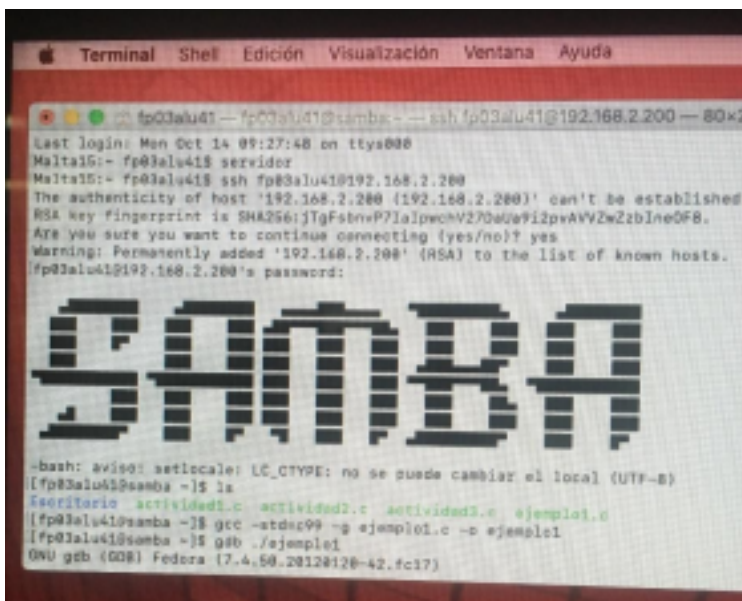
GDB ofrece la posibilidad de trazar y modificar la ejecución de un programa. El usuario puede controlar y alterar los valores de las variables internas del programa.

GDB no contiene su propia interfaz gráfica de usuario y por defecto se controla mediante una interfaz de línea de comandos. Existen diversos front-ends que han sido diseñados para GDB, como Data Display Debugger, GDBtk/Insight y el «modo GUD» en Emacs.

Desarrollo:

Ejemplo 1.

Para el ejemplo 1 primero descargamos el programa y lo compilamos en la terminal, en las Macs de la escuela no se pudo, pero nos conectamos todos a una máquina con Linux y pudimos correr y compilar con gdb, lo que hicimos fue poner cada uno de los comandos que venían en la práctica y GDB nos pone los errores que tenemos en el programa.



```
Terminal Shell Edición Visualización Ventana Ayuda
fp03alu41 ~ - ssh fp03alu41@samba - ssh fp03alu41@192.168.2.200 - 80x24
Last login: Mon Oct 14 09:27:48 on ttys000
Malta15:~ fp03alu41$ ssh server
Malta15:~ fp03alu41$ ssh fp03alu41@192.168.2.200
The authenticity of host '192.168.2.200 (192.168.2.200)' can't be established.
RSA key fingerprint is SHA256:jTgFsbvP7IalpwchV270dave9i2pvAVV2wZzb1ne0F8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.2.200' (RSA) to the list of known hosts.
fp03alu41@192.168.2.200's password:
Samba
-bash: avis00: actiocale: LC_CTYPE: no se puede cambiar el local (UTF-8)
fp03alu41@samba ~$ ls
Escritorio  actividad1.c  actividad2.c  actividad3.c  ejemplo1.c
fp03alu41@samba ~$ gcc -std=c99 -g ejemplo1.c -o ejemplo1
fp03alu41@samba ~$ gdb ./ejemplo1
GNU gdb (GDB) Fedora (7.4.50.20120120-42.fc17)
```

```
Terminal Shell Edición Visualización Ventana Ayuda
fp03alu41 — fp03alu41@samba:~ — ssh fp03alu41@192.168.2.200 — 80x24

También podemos poner un caracter: B
Un numero real: 89.88

Program received signal SIGSEGV, Segmentation fault.
0x00000000400660c in main (argc=19, argv=0x1100000010) at ejemplo1.c:21
21      lista[i] = i;
Missing separate debuginfos, use: debuginfo-install glibc-2.15-37.fc17.x86_64
(gdb) list
16      printf("También podemos poner un caracter: %c\n", caracter);
17      printf("Un numero real: %.2f\n", numeroReal);
18
19      // Podemos llenar la lista con valores
20      for(int i = numero ; i >= numero ; i++){
21          lista[i] = i;
22      }
23
24      // Y ahora podemos hacer calculos con la lista
25      for(int i = numero ; i >= numero ; i++){
(gdb) q
A debugging session is active.

    Inferior 1 [process 21556] will be killed.

Quit anyway? (y or n) 
```

```
Terminal Shell Edición Visualización Ventana Ayuda
fp03alu41 — fp03alu41@samba:~ — ssh fp03alu41@192.168.2.200 — 80x24

[ No Source Available ]

exec No process in:
(gdb) Line: ?? PC: ??
```



```

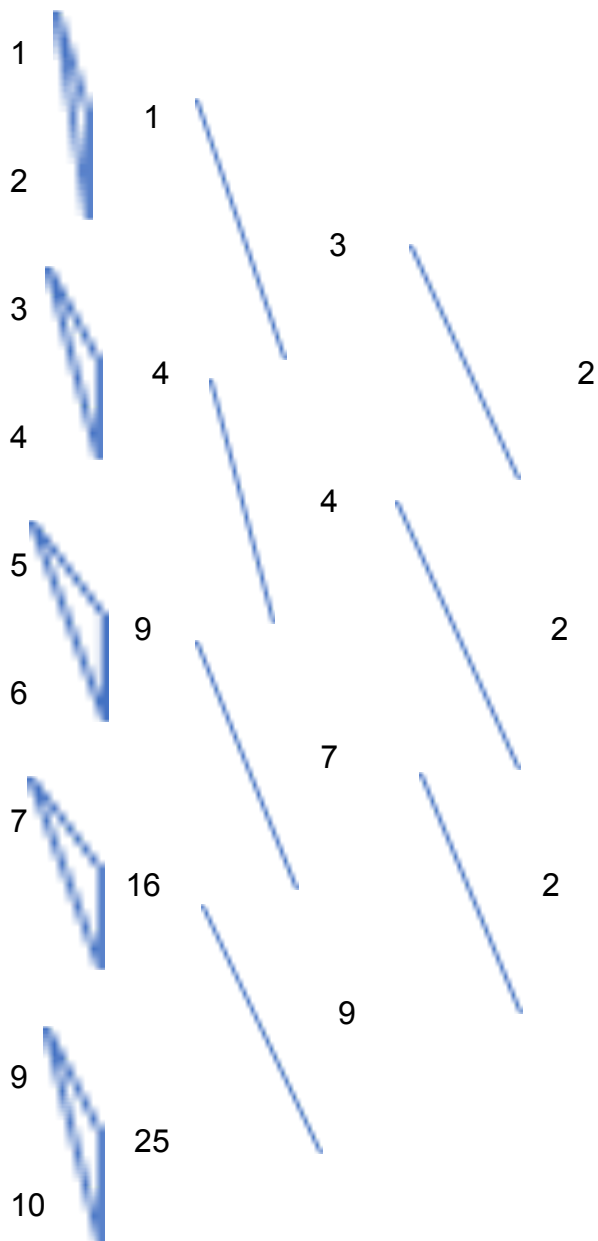
[fp@kali:~/samba]$ gdb ejemplo1
GNU gdb (GDB) Fedora (7.4.50.20120120-42.fc17)
Copyright (C) 2012 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /users/fp@fp@kali:~/samba/...done.
[fp@kali:~/samba]$ gdb ejemplo1
GNU gdb (GDB) Fedora (7.4.50.20120120-42.fc17)
Copyright (C) 2012 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 "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /users/fp@fp@kali:~/samba/...done.
[fp@kali:~/samba]$ gdb ejemplo1
bash: gdb: command not found...
Similar command is: 'gdb'
[fp@kali:~/samba]$ break 20
--bash: break: sólo tiene significado en un ciclo 'for', 'while' o 'until'
[fp@kali:~/samba]$ gdb ejemplo1
GNU gdb (GDB) Fedora (7.4.50.20120120-42.fc17)
Copyright (C) 2012 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 "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /users/fp@fp@kali:~/samba/...done.
(gdb) break 20
Breakpoint 1 at 0x4005f7: file ejemplo1.c, line 20.
(gdb) r
No symbol "i" in current context.
(gdb) gdb ejemplo1
Undefined command: "gdb". Try "help".
[fp@kali:~/samba]$ gdb ejemplo1
GNU gdb (GDB) Fedora (7.4.50.20120120-42.fc17)
Copyright (C) 2012 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 "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /users/fp@fp@kali:~/samba/...done.
(gdb) break 20
Breakpoint 1 at 0x4005f7: file ejemplo1.c, line 20.
(gdb) r
Starting program: /users/fp@fp@kali:~/samba/...
Primero texto solo
Luego podemos poner un enter: 10
Después podemos poner un carácter: D
Un número final: 99.99
Breakpoint 1, main (argc=1, argv=0x7fffffe390) at ejemplo1.c:20
20      printf("i = numero 1 i es numero : %i\n", i);
Missing separate debuginfos, use: debuginfo-install glibc-2.15-37.fc17.x86_64
[fp@kali:~/samba]$

```

Actividad 1.

Utilizar GDB para encontrar la utilidad del programa y describir su funcionalidad.

El programa de la actividad 1 era una serie ya que al poner el numero 2 por ejemplo te sale el numero 1 , si pones el 3 te da el 4 y si pones el 5 te da 9 pero porque de 1 a 4 hay tres números de diferencia y del 4 al 9 hay 4 números de diferencia y del 3 al 4 hay dos números de diferencia , así que la serie es de esta manera:



```
actividad1.c
4  {
5      int N, CONT, AS;
6      AS=0;
7      CONT=1;
8      printf("Ingresa un número: ");
9      scanf("%i",&N);
10     while(CONT<=N)
11     {
12         AS=(AS+CONT);
13         CONT=(CONT+2);
14     }
15     printf("\nEl resultado es: %i\n", AS);
16 }
```

native Thread 17908.0x172c In: main L6 PC: 0x10040108d
(gdb) start
Temporary breakpoint 1 at 0x10040108d: file actividad1.c, line 6.
Starting program: /home/hugo morones/actividad1
[New Thread 17908.0x172c]
[New Thread 17908.0x50d5]
Thread 1 "actividad1" hit Temporary breakpoint 1, main () at actividad1.c:6
(gdb) |

```
4  {
5      int N, CONT, AS;
6      AS=0;
7      CONT=1;
8      printf("Ingresa un número: ");
9      scanf("%i",&N);
10     scanf("%i",&N);
11     while(CONT<=N)
12     {
13         AS=(AS+CONT);
14         AS=(AS+CONT);
15         CONT=(CONT+2);
16     }
17     printf("\nEl resultado es: %i\n", AS);
18 }
```

native Thread 17908.0x172c In: main L9 PC: 0x1004010a7
[New Thread 17908.0x172c] L3 c2
Thread 1 "actividad1" hit Temporary breakpoint 1, main () at actividad1.c:6
(gdb) n
[New Thread 17908.0x3bb4]
[New Thread 17908.0x2798]
Ingresa un número: 9
(gdb) next
(gdb) n
(gdb) |

Compilación en sublime text de actividad 1:

```
actividad1.c  actividad3.c  actividad01.c
1  #include <stdio.h>
2
3  void main()
4  {
5      int N, CONT, AS;
6      AS=0;
7      CONT=1;
8      printf("Ingresa un número: ");
9      scanf("%i",&N);
10     while(CONT<=N)
11     {
12         AS=(AS+CONT);
13         CONT=(CONT+2);
14     }
15     printf("\nEl resultado es: %i\n", AS);
16 }
```

```
Documentos — -bash — 80x24
actividad01.c
[Libano06:documents fp03alu41$ gcc actividad01.c -o act
actividad01.c:3:1: warning: return type of 'main' is not 'int'
    [-Wmain-return-type]
void main()
^
actividad01.c:3:1: note: change return type to 'int'
void main()
^~~~~~
int
1 warning generated.
[Libano06:documents fp03alu41$ ./act
Ingresa un número: 3

El resultado es: 4
[Libano06:documents fp03alu41$ ./act
Ingresa un número: 6

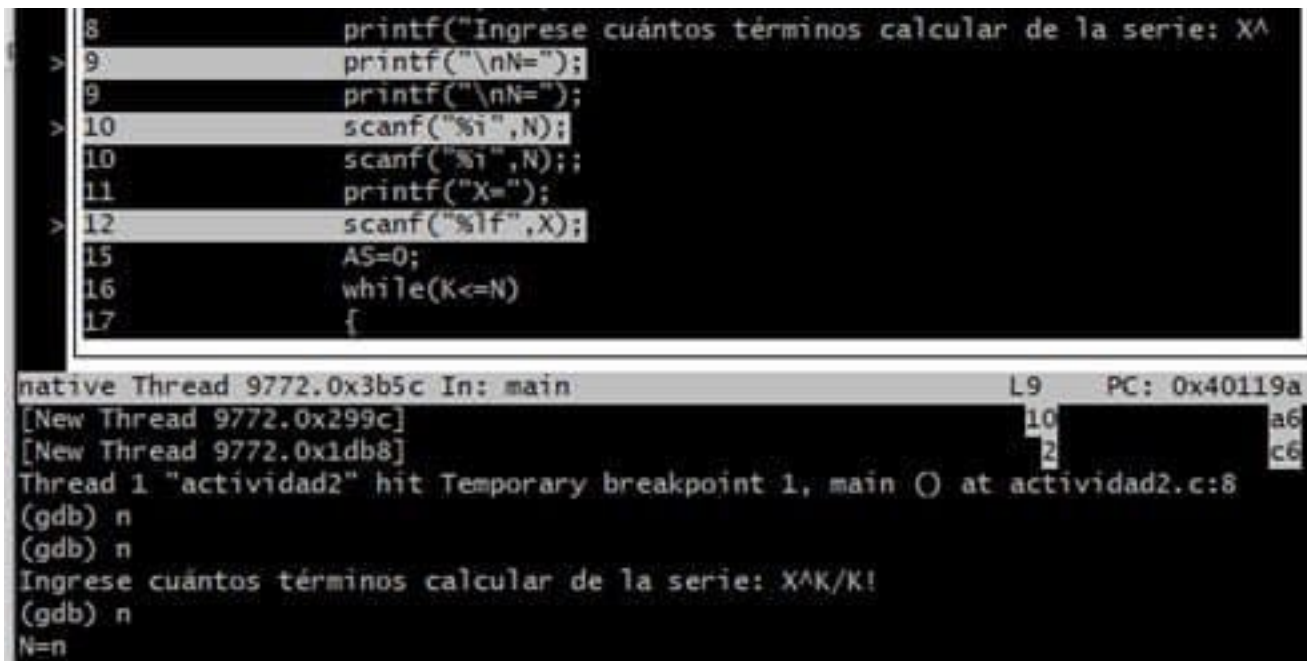
El resultado es: 9
[Libano06:documents fp03alu41$ ./act
Ingresa un número: 10

El resultado es: 25
[Libano06:documents fp03alu41$ ]
```


Actividad 2. Utilizar GDB para corregir el programa.

NOTA: para compilar el código de la actividad, ejecutar:

```
$ gcc -w actividad2.c -o actividad2 -lm
```



```
8      printf("Ingrese cuántos términos calcular de la serie: X^
> 9      printf("\nN=");
9      printf("\nN=");
> 10     scanf("%i",N);
10     scanf("%i",N);
11     printf("X=");
> 12     scanf("%lf",X);
15     AS=0;
16     while(K<=N)
17     {

native Thread 9772.0x3b5c In: main L9 PC: 0x40119a
[New Thread 9772.0x299c] 10 a6
[New Thread 9772.0x1db8] 2 c6
Thread 1 "actividad2" hit Temporary breakpoint 1, main () at actividad2.c:8
(gdb) n
(gdb) n
Ingrese cuántos términos calcular de la serie: X^K/K!
(gdb) n
N=n
```

Al poner el programa en GDB nos damos cuenta que hay un error en el espacio de `printf("\n N=");`

Otro error en los `scanf("%i",&N);` , `scanf("%lf", &X);` que sirven para obtener el valor de la variable que estás llamando.

Vemos que en el programa hay errores de sintaxis y no de lógica ya que al correr y compilar el programa siempre te da el mismo valor.

```

1  #include <stdio.h>
2  #include <math.h>
3
4  int main()
5  {
6      int K, AP, N;
7      double X, AS;
8      printf("Ingrese cuántos términos calcular de la serie: X^K/K!");
9      printf("\n N=");
10     scanf("%i",&N);
11     printf("X=");
12     scanf("%lf",&X);
13     K=0;
14     AP=1;
15     AS=0;
16     while(K<=N)
17     {
18         AS=AS+pow(X,K)/AP;
19         K=K+1;
20         AP=AP*K;
21     }
22     printf("Resultado=%le",AS);
23 }
24

```

```

Documents — -bash — 80x24
actividad2.c:12:14: warning: format specifies type 'double *' but the argument
has type 'double' [-Wformat]
scanf("%lf",X);
      ~~~~ ^
1 warning generated.
Libano06:documents fp03alu41$ ./act
Ingrese cuántos términos calcular de la serie: X^K/K!
N=1 2 3
Segmentation fault: 11
Libano06:documents fp03alu41$ ./act
Ingrese cuántos términos calcular de la serie: X^K/K!
N=1 1
Segmentation fault: 11
Libano06:documents fp03alu41$ gcc actividad2.c -o act
actividad2.c:12:14: warning: format specifies type 'double *' but the argument
has type 'double' [-Wformat]
scanf("%lf",X);
      ~~~~ ^
1 warning generated.
Libano06:documents fp03alu41$ ./act
Ingrese cuántos términos calcular de la serie: X^K/K!
N=1 2
Segmentation fault: 11
Libano06:documents fp03alu41$

```

Actividad 3. Utilizar GDB para corregir el programa.

```
actividad3.c
1
2 {
3     int numero;
4
5     printf("Ingrese un número:\n");
6     scanf("%i",&numero);
7
8     long int resultado = 1;
9     while(numero>=0){
10         numero--;
11         resultado *= numero;
12     }
13
14     printf("El factorial de %i es %li.\n", numero, resultado);
15
16     return 0;
17 }
18
19
20
21
22
```

```

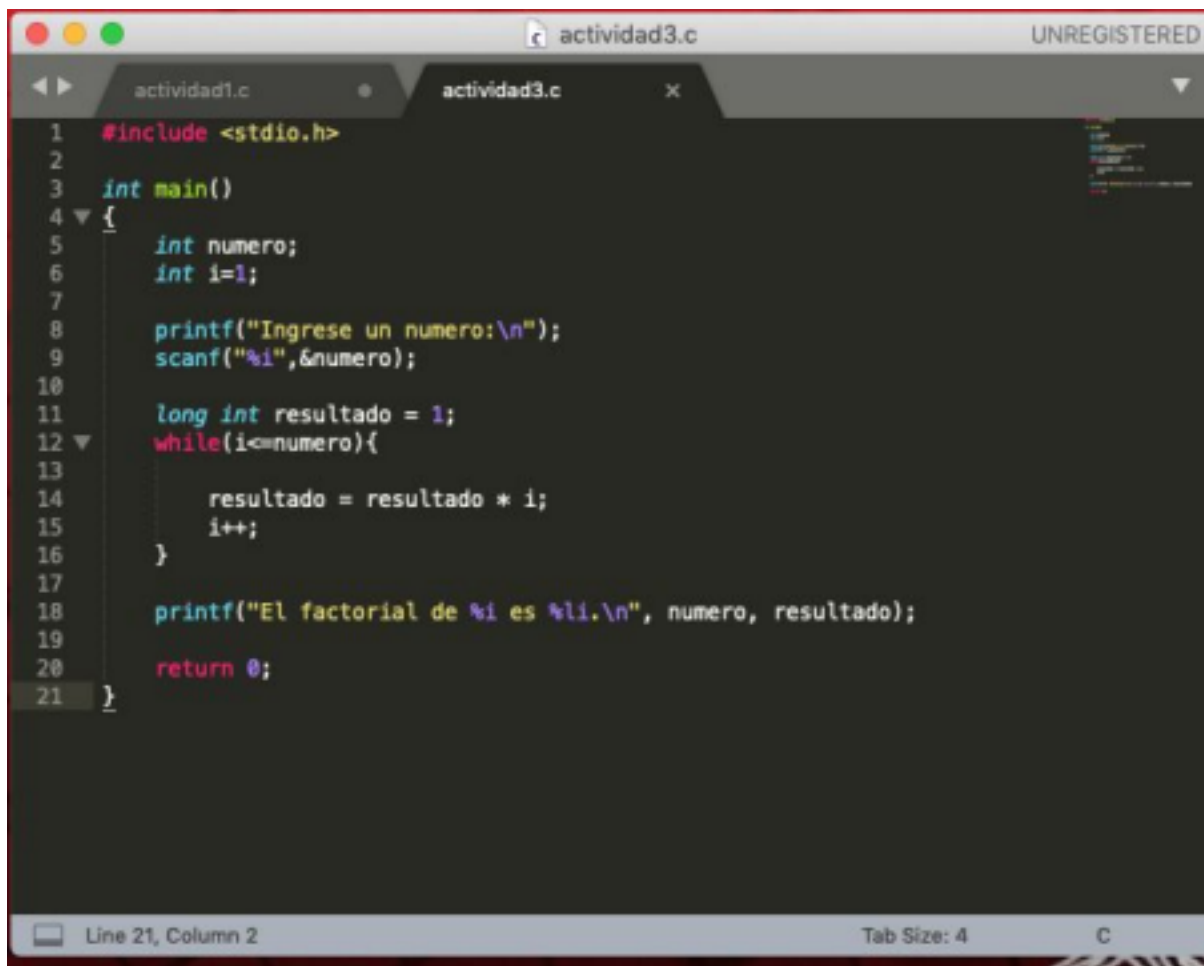
1     printf("Ingrese un número:\n");
2     int numero; scanf("%i",&numero);
3
4     printf("Ingrese un número:\n");
5     scanf("%i",&numero);
6     numero--;
7     long int resultado = 1;
8     while(numero>=0){
9         numero--;
10        resultado *= numero;
11    }
12    printf("El factorial de %i es %li.\n", numero, resultado);
13
14    return 0;
15
16    printf("El factorial de %i es %li.\n", numero, resultado);
17
18    return 0;
19 }
20
21
22
23
24
25
26
27
28
29
30
31
32
nat11 Thread 16572.0x48b8 In: main
(gdb) start
Temporary breakpoint 1 at 0x10040108d: file actividad3.c, line 7.
Starting program:
[New Thread 16572.0x4008]
[New Thread 16572.0x4b04]

(gdb) n
0x000000018004a816 in _cygwin_exit_return () from /usr/bin/cygwin1.dll
Single stepping until exit from function _cygwin_exit_return,
which has no line number information.
[Thread 16572.0x4008 exited with code 0]
[Thread 16572.0x4b04 exited with code 0]
[Inferior 1 (process 16572) exited normally]
(gdb) n
The program is not being run.
The program is not being run.
(gdb)
```

```
actividad3.c
1  #include <stdio.h>
2
3  int main()
4  {
5      int numero;
6
7      printf("Ingrese un número:\n");
8      scanf("%i",&numero);
9
10     long int resultado = 1;
11     while(numero>=0){
12         numero--;
13         resultado *= numero;
14     }
15
16     printf("El factorial de %i es %li.\n", numero, resultado);
17
18     return 0;
19 }
20
21
22
23
24
25
26
27
28
29
30
31
Native Thread 20640.0x518c In: main
(gdb) n
[New Thread 20640.0x2d0c]
[New Thread 20640.0x3518]
(gdb) n
(gdb) p i
No symbol "i" in current context.
(gdb) print lista
No symbol "lista" in current context.
(gdb) display lista
No symbol "lista" in current context.
(gdb) |
```

Uno de los errores que nos marca en GDB es en While(numero>=0){

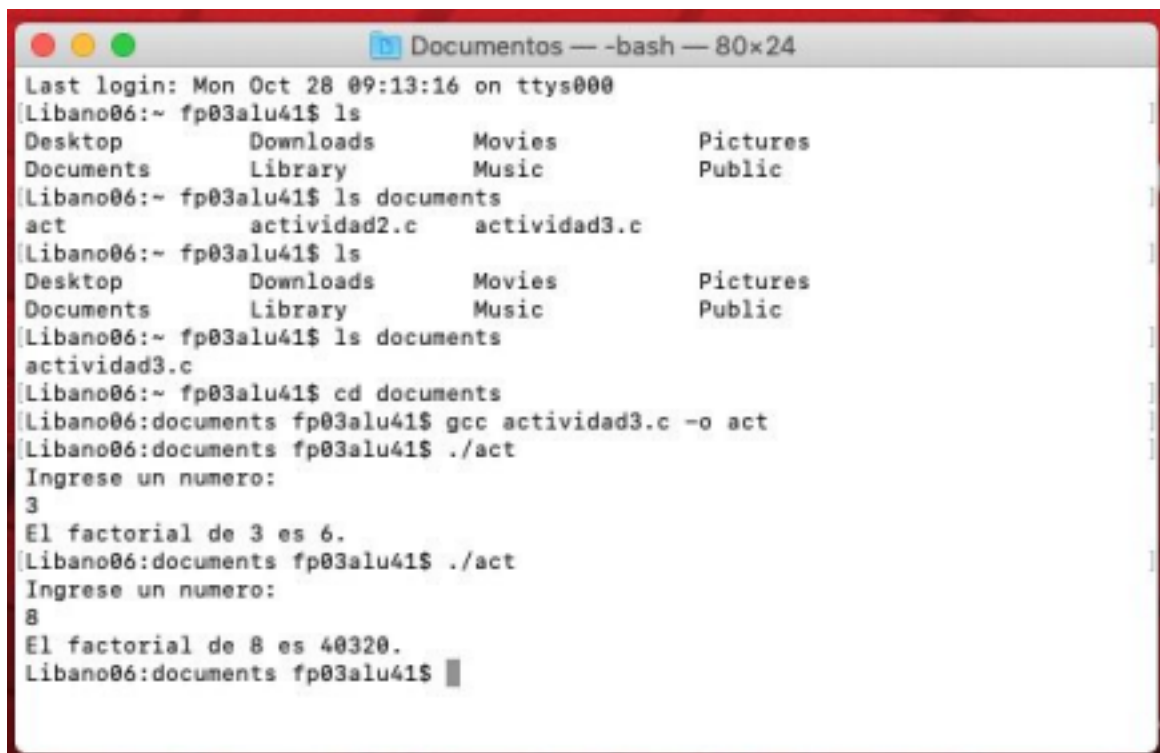
Al compilar y correr el programa corregido desde GDB en sublime text ahora si, si damos un número de entrada te da de resultado su factorial.



The screenshot shows a code editor window titled 'actividad3.c' with a status bar indicating 'UNREGISTERED'. The code is written in C and calculates the factorial of a user-input number. The code is as follows:

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int numero;
6      int i=1;
7
8      printf("Ingrese un numero:\n");
9      scanf("%i",&numero);
10
11     long int resultado = 1;
12     while(i<=numero){
13
14         resultado = resultado * i;
15         i++;
16     }
17
18     printf("El factorial de %i es %li.\n", numero, resultado);
19
20     return 0;
21 }
```

The status bar at the bottom shows 'Line 21, Column 2', 'Tab Size: 4', and 'C'.



The screenshot shows a terminal window titled 'Documentos — -bash — 80x24'. The terminal output shows the user navigating to the 'documents' directory, compiling the program, and running it twice with inputs 3 and 8.

```
Last login: Mon Oct 28 09:13:16 on ttys000
[Libano06:~ fp03alu41$ ls
Desktop      Downloads    Movies       Pictures
Documents    Library      Music        Public
[Libano06:~ fp03alu41$ ls documents
act          actividad2.c  actividad3.c
[Libano06:~ fp03alu41$ ls
Desktop      Downloads    Movies       Pictures
Documents    Library      Music        Public
[Libano06:~ fp03alu41$ ls documents
actividad3.c
[Libano06:~ fp03alu41$ cd documents
[Libano06:documents fp03alu41$ gcc actividad3.c -o act
[Libano06:documents fp03alu41$ ./act
Ingrese un numero:
3
El factorial de 3 es 6.
[Libano06:documents fp03alu41$ ./act
Ingrese un numero:
8
El factorial de 8 es 40320.
[Libano06:documents fp03alu41$
```

Conclusión:

Para concluir esta práctica es importante saber utilizar bien GDB y todos sus comandos ya que este nos permite saber nuestros errores de los programas y así es más fácil saber porque no está corriendo el programa, pues a veces desde los editores de texto no te marcan bien los errores al programar.