



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: Marco Antonio Martínez Quintana

Asignatura: Fundamentos de Programación

Grupo: 03

No de Práctica(s): 8

Integrante(s): López Martínez Diana

*No. de Equipo de
cómputo empleado:*

No. de Lista o Brigada: 28

Semestre: 1

Fecha de entrega: 13/12/2020

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.



practica: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
#include <stdio.h>
```

```
void main()
{
    int N, CONT, AS;
    AS=0;
    CONT=1;
    printf("TECLEA UN NUMERO: ");
    scanf("%i",&N);
    while(CONT<=N)
    {
        AS=(AS+CONT);
        CONT=(CONT+2);
    }
    printf("\nEL RESULTADO ES %i\n", AS);
}
```

```
C:\Users\paust\Downloads\Lenguaje c\Ejemplos>gcc practica.c -o practica.exe
```

```
C:\Users\paust\Downloads\Lenguaje c\Ejemplos>practica.exe
```

```
TECLEA UN NUMERO: 6
```

```
EL RESULTADO ES 9
```

```
C:\Users\paust\Downloads\Lenguaje c\Ejemplos>gcc practica.c -o practica.exe
```

```
C:\Users\paust\Downloads\Lenguaje c\Ejemplos>practica.exe
```

```
TECLEA UN NUMERO: 15
```

```
EL RESULTADO ES 64
```

```
C:\Users\paust\Downloads\Lenguaje c\Ejemplos>gcc -g -o practica practica.c

C:\Users\paust\Downloads\Lenguaje c\Ejemplos>gdb ./practica
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\paust\Downloads\Lenguaje c\Ejemplos\practica.exe...done.
(gdb) break practica.c:5
```

Puntos de ruptura

```
(gdb) break practica.c:5
Breakpoint 1 at 0x40141e: file practica.c, line 5.
(gdb) break practica.c:7
Breakpoint 2 at 0x401426: file practica.c, line 7.
```

Ejecución del programa

```
(gdb) run
Starting program: C:\Users\paust\Downloads\Lenguaje c\Ejemplos\./practica.exe
[New Thread 6364.0x21d4]

Breakpoint 1, main () at practica.c:6
6          AS=0;
```

c: Continuar con la ejecución del programa después de un punto de ruptura.

```
(gdb) c
Continuing.

Breakpoint 2, main () at practica.c:7
7          CONT=1;
```

list o l: Permite listar diez líneas del código fuente del programa

```
(gdb) list
3      void main()
4      {
5          int N, CONT, AS;
6          AS=0;
7          CONT=1;
8          printf("TECLEA UN NUMERO: ");
9          scanf("%i",&N);
10         while(CONT<=N)
11         {
12             AS=(AS+CONT);
```

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.

```
#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);
        }
    }
}
```

Punto de ruptura

```
(gdb) break practicaMal.c:5
Breakpoint 1 at 0x40141e: file practicaMal.c, line 5.
(gdb) break practicaMal.c:7
Breakpoint 2 at 0x401428: file practicaMal.c, line 7.
```

Ejecución del programa

```
(gdb) run
Starting program: C:\Users\paust\Downloads\Lenguaje c\Ejemplos/./practicaMal.exe
[New Thread 9356.0x1348]

Breakpoint 1, main () at practicaMal.c:6
6          for(i=1; i<10; i++)
```

El error se encuentra en la línea 6 del código, ya que solo nos indica que al ejecutar el programa va a mostrar las tablas menores al número 10 por ello debemos de agregarle un signo igual para que también nos muestre la tabla del número 10.

Punto de ruptura

```
(gdb) break practicaMal.c:8
Note: breakpoint 2 also set at pc 0x401428.
Breakpoint 3 at 0x401428: file practicaMal.c, line 8.
(gdb) break practicaMal.c:10
Breakpoint 4 at 0x401446: file practicaMal.c, line 10.
```


Ejecución del programa

```
(gdb) s

Tabla del 1
9          for(j=1; j==10; j++)
```

El error se encuentra al realizar la estructura de repetición for ya que no estamos pidiendo que nos enumere las tablas del uno al 10 cada una.

Código corregido

```
 practicaA: Bloc de notas
Archivo Edición Formato Ver Ayuda
#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);
        }
    }
}
```

```
C:\Users\paust\Downloads\Lenguaje c\Ejemplos>gcc practicaA.c -o practicaA.exe
```

```
C:\Users\paust\Downloads\Lenguaje c\Ejemplos>practicaA.exe
```

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

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

```
include <stdio.h>
include <math.h>
void main()

    int K, X, AP, N;
    float AS;
    printf("EL TERMINO GENERICO DE LA SERIE ES: X^K/K!");
    printf("\nN=");
    scanf("%d",N);
    printf("X=");
    scanf("%d",X);
    K=0;
    AP=1;
    AS=0;
    while(K<=N)
    {
        AS=AS+pow(X,K)/AP;
        K=K+1;
        AP=AP*K;
    }
    printf("SUM=%le",AS);
```

Puntos de ruptura

```
(gdb) break practicaB.c:17
Breakpoint 1 at 0x401482: file practicaB.c, line 17.
(gdb) break practicaB.c:19
Breakpoint 2 at 0x4014b3: file practicaB.c, line 19.
```

listar diez líneas del código fuente del programa

```
(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=");
```

Conclusión

Al realizar cualquier código pueden existir errores los cuales son muy importantes encontrarlos y corregirlos, pueden presentarse diferentes tipo de errores como: errores de digitación, errores de sintaxis, de diseño etc.

El depurador nos ayudara a entender donde fue el error para solucionarlo, también gracias a el ahorramos tiempo en encontrar alguno de ellos, es una herramienta crucial para desarrollar un código impecable.

Bibliografía

Solano Gálvez, García Cano, Sandoval Montaña, Nakayama Cervantes, Arteaga Ricci, Castañeda Perdomo, I., 2020. Laboratorio Salas A Y B. [online] Lcp02.fi-b.unam.mx. Available at: <<http://lcp02.fi-b.unam.mx/>> [Accessed 6 April 2020].