



## Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

# Laboratorios de computación salas A y B

*Profesor:* Marco Antonio Martínez Quintana

*Asignatura:* Laboratorio de Fundamentos de Programación

*Grupo:* 3

*No de Práctica(s):* 10

*Integrante(s):* Cecilia Torres Bravo

*No. de Equipo de  
cómputo empleado:* N/A

*No. de Lista o Brigada:* 54

*Semestre:* 1º

*Fecha de entrega:* 13 de diciembre del 2020

*Observaciones:*

**CALIFICACIÓN:** \_\_\_\_\_

# Depuración de programas

## Objetivo

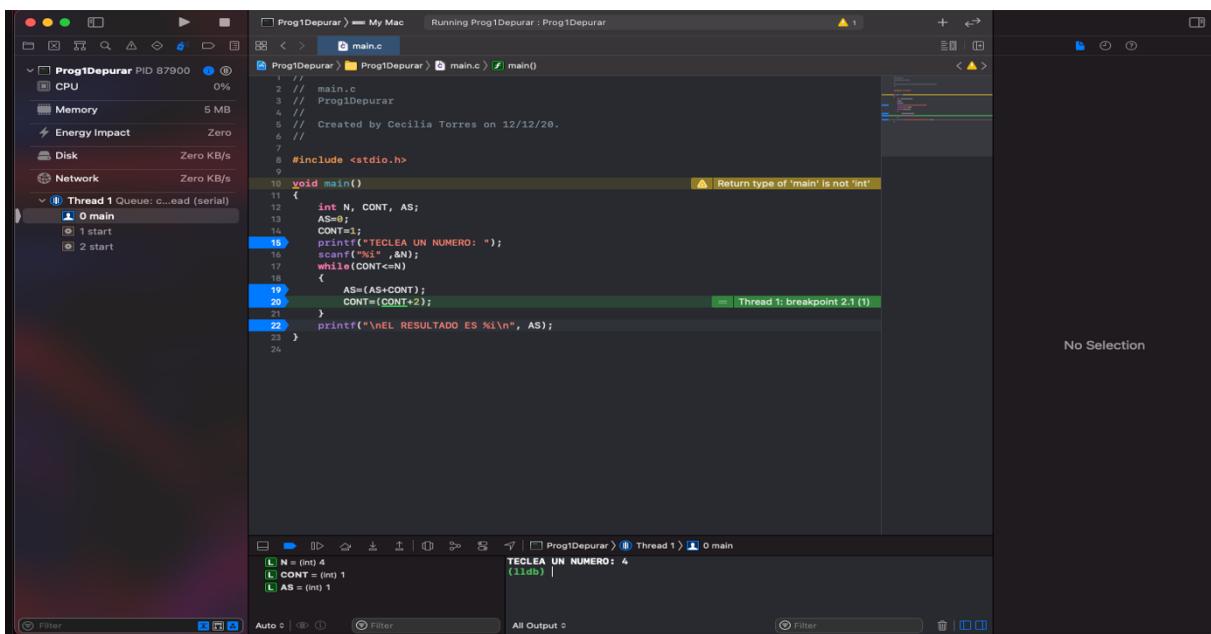
Aprender las técnicas básicas de depuración de programas en C para revisar de manera precisa el flujo de ejecución de un programa y el valor de las variables; en su caso, corregir posibles errores.

## Introducción

La depuración se refiere a someter un programa a un ambiente controlado para conocer su funcionamiento paso a paso. Nos sirve para poder optimizar el programa, para revisar algún fallo y/o corregir algún defecto o error de ejecución. Asimismo, con un depurador podemos realizar algunas de las siguientes funciones básicas; ejecutar el programa, ver el código fuente, establecer un punto de ruptura, continuar con la ejecución, continuar, ejecutar la siguiente función, ejecutar la línea anterior y visualizar el valor de las variables.

## Actividades

**A-1** 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.



The screenshot shows the Xcode debugger interface. The top left pane displays system resource usage (CPU, Memory, Energy Impact, Disk, Network) and a list of threads. Thread 1 is selected, showing a queue named 'c...ead (serial)' with tasks '0 main', '1 start', and '2 start'. The bottom left pane shows the source code of 'main.c' with a breakpoint set at line 19. The bottom right pane shows the output window where the program has printed 'TECLEA UN NUMERO: 4'.

```
1 // main.c
2 // Prog1Depurar
3 // Created by Cecilia Torres on 12/12/20.
4 //
5 // TECLEA UN NUMERO: 4
6 //
7 #include <stdio.h>
8
9 void main()
10 {
11     int N, CONT, AS;
12     AS=0;
13     CONT=1;
14     printf("TECLEA UN NUMERO: ");
15     scanf("%i", &N);
16     while(CONT<=N)
17     {
18         AS=(AS+CONT);
19         CONT=(CONT+2);
20     }
21     printf("\nEL RESULTADO ES %i\n", AS);
22 }
```

No Selection

```
TECLEA UN NUMERO: 4
(11db)
```

Primero, incluí 4 breakpoints en el programa para poder ver su funcionamiento paso por paso.

Luego, fui observando el valor de cada variable después de cada breakpoint.

This screenshot is identical to the one above, showing the Xcode debugger interface with a breakpoint at line 19. The source code, output window, and status bar are all the same.

```
1 // main.c
2 // Prog1Depurar
3 // Created by Cecilia Torres on 12/12/20.
4 //
5 // TECLEA UN NUMERO: 4
6 //
7 #include <stdio.h>
8
9 void main()
10 {
11     int N, CONT, AS;
12     AS=0;
13     CONT=1;
14     printf("TECLEA UN NUMERO: ");
15     scanf("%i", &N);
16     while(CONT<=N)
17     {
18         AS=(AS+CONT);
19         CONT=(CONT+2);
20     }
21     printf("\nEL RESULTADO ES %i\n", AS);
22 }
```

No Selection

```
TECLEA UN NUMERO: 4
(11db)
```

The screenshot shows the Xcode debugger interface. On the left, the sidebar displays system resource usage (CPU, Memory, Energy Impact, Disk, Network) and a list of threads, with Thread 1 selected. The main pane shows the source code for `main.c`. A blue arrow points to line 20, which contains the assignment `CONT=(CONT+2);`. A red warning icon is present in the gutter next to this line. The bottom pane shows the output window with the command `(lldb)` and the printed value `TECLEA UN NUMERO: 4`.

```
1 // 
2 // main.c
3 // Prog1Depurar
4 //
5 // Created by Cecilia Torres on 12/12/20.
6 //
7 #include <stdio.h>
8
9 void main()
10 {
11     int N, CONT, AS;
12     AS=0;
13     CONT=1;
14     printf("TECLEA UN NUMERO: ");
15     scanf("%i", &N);
16     while(CONT<=N)
17     {
18         AS=(AS+CONT);
19         CONT=(CONT+2);
20     }
21     printf("\nEL RESULTADO ES %i\n", AS);
22 }
23 
```

Conforme iba viendo cómo cambiaban los valores de CONT y AS, pude ir entendiendo mejor el funcionamiento del programa.

This screenshot is similar to the previous one but shows a different state. The blue arrow now points to line 22, where the program is printing the result. The bottom output window shows the command `(lldb)` and the printed value `TECLEA UN NUMERO: 4`.

```
1 // 
2 // main.c
3 // Prog1Depurar
4 //
5 // Created by Cecilia Torres on 12/12/20.
6 //
7 #include <stdio.h>
8
9 void main()
10 {
11     int N, CONT, AS;
12     AS=0;
13     CONT=1;
14     printf("TECLEA UN NUMERO: ");
15     scanf("%i", &N);
16     while(CONT<=N)
17     {
18         AS=(AS+CONT);
19         CONT=(CONT+2);
20     }
21     printf("\nEL RESULTADO ES %i\n", AS);
22 }
23 
```

```

1 //
2 // main.c
3 // Prog1Depurar
4 //
5 // Created by Cecilia Torres on 12/12/20.
6 //
7
8 #include <stdio.h>
9
10 void main()
11 {
12     int N, CONT, AS;
13     AS=0;
14     CONT=1;
15     printf("TECLEA UN NUMERO: ");
16     scanf("%i", &N);
17     while(CONT<=N)
18     {
19         AS=(AS+CONT);
20         CONT=(CONT+2);
21     }
22     printf("\nEL RESULTADO ES %i\n", AS);
23 }
24

```

No Selection

TECLEA UN NUMERO: 4  
EL RESULTADO ES 4  
Program ended with exit code: 19

Luego de correr el código con diferentes valores del 1 al 10 en “N”. obtuve la siguiente tabla:

1	2	3	4	5	6	7	8	9	10
1	1	4	4	9	9	16	16	25	25

Tomando en consideración dichos valores, pienso que la funcionalidad del programa es la de calcular las potencias de los números, sin embargo, existe una falla en las sentencias a seguir en *while*.

**A-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 corregirlo 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.

The screenshot shows two instances of the Xcode debugger interface. The top instance has a debug session named "Prog2Depurar" with PID 1680. It displays the "main.c" file with a breakpoint set at line 18. The output window below shows the results of the program execution, which is printing multiplication tables from 1 to 9. The bottom instance shows the same setup but with a message "No Debug Session" in the top-left corner.

```

1 //
2 //  main.c
3 //  Prog2Depurar
4 //
5 //  Created by Cecilia Torres on 12/12/20.
6 //
7
8 #include <stdio.h>
9
10 void main()
11 {
12     int i, j;
13
14     for(i=1; i<=10; i++)
15     {
16         printf("\nTabla del %i\n", i);
17
18         for(j=1; j<=10; j++)
19         {
20             printf("%i X %i = %i\n", i, j, i*j);
21         }
22     }
23 }

```

No Selection

i = (int) 9  
j = (int) 1

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  
(lldb)

All Output

No Selection

No Debug Session

Tabla del 2  
Tabla del 3  
Tabla del 4  
Tabla del 5  
Tabla del 6  
Tabla del 7  
Tabla del 8  
Tabla del 9  
Tabla del 10  
Program ended with exit code: 11

All Output

Primero corrí el código con algunos breakpoints para encontrar el error y fue cuando me di cuenta de que en la línea 14 la expresión lógica no se encontraba bien definida, pues decía que solamente se mostraran valores menores a 10 cuando en realidad tendría que decir valores menores o iguales a 10.

```

R. > gcc - Build and ... C prog2depurar.c > launch.json
VARIABLES
prog2depurar.c > main()
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 }

WATCH
CALL STACK
TERMINAL PROBLEMS 1 OUTPUT DEBUG CONSOLE
Filter (e.g. text, | exclude)
@> 9 X 9 = 81\r\n"
@> 9 X 10 = 90\r\n"
@>\r\n"
@>"Tabla del 10\r\n"
@>"10 X 1 = 10\r\n"
@>"10 X 2 = 20\r\n"
@>"10 X 3 = 30\r\n"
@>"10 X 4 = 40\r\n"
@>"10 X 5 = 50\r\n"
@>"10 X 6 = 60\r\n"
@>"10 X 7 = 70\r\n"
@>"10 X 8 = 80\r\n"
@>"10 X 9 = 90\r\n"
@>"10 X 10 = 100\r\n"
The program '/Users/Cecilia.Torres/C/prog2depurar' has exited with code 0 (0x00000000).

```

TERMINAL PROBLEMS 1 OUTPUT DEBUG CONSOLE

Ln 5, Col 8 Spaces: 4 UTF-8 LF C Mac ⌘ ⌘

```

TERMINAL PROBLEMS 1 OUTPUT DEBUG
@>\r\n"
@>"Tabla del 1\r\n"
@>"1 X 1 = 1\r\n"
@>"1 X 2 = 2\r\n"
@>"1 X 3 = 3\r\n"
@>"1 X 4 = 4\r\n"
@>"1 X 5 = 5\r\n"
@>"1 X 6 = 6\r\n"
@>"1 X 7 = 7\r\n"
@>"1 X 8 = 8\r\n"
@>"1 X 9 = 9\r\n"
@>"1 X 10 = 10\r\n"
@>\r\n"
@>"Tabla del 2\r\n"
@>"2 X 1 = 2\r\n"
@>"2 X 2 = 4\r\n"
@>"2 X 3 = 6\r\n"
@>"2 X 4 = 8\r\n"
@>"2 X 5 = 10\r\n"
@>"2 X 6 = 12\r\n"
@>"2 X 7 = 14\r\n"
@>"2 X 8 = 16\r\n"
@>"2 X 9 = 18\r\n"
@>"2 X 10 = 20\r\n"
@>\r\n"
@>"Tabla del 3\r\n"
@>"3 X 1 = 3\r\n"
@>"3 X 2 = 6\r\n"
@>"3 X 3 = 9\r\n"
@>"3 X 4 = 12\r\n"
@>"3 X 5 = 15\r\n"
@>"3 X 6 = 18\r\n"
@>"3 X 7 = 21\r\n"
@>"3 X 8 = 24\r\n"
@>"3 X 9 = 27\r\n"
@>"3 X 10 = 30\r\n"
@>\r\n"
@>"Tabla del 4\r\n"
@>"4 X 1 = 4\r\n"
@>"4 X 2 = 8\r\n"
@>"4 X 3 = 12\r\n"
@>"4 X 4 = 16\r\n"
@>"4 X 5 = 20\r\n"
@>"4 X 6 = 24\r\n"
@>"4 X 7 = 28\r\n"
@>"4 X 8 = 32\r\n"
@>"4 X 9 = 36\r\n"
@>"4 X 10 = 40\r\n"
@>\r\n"

```

Luego de corregir esa parte y volver a correr el código pude ver que el siguiente *for* tenía un error muy similar, indicaba que solo se imprimieran las tablas cuando *j* tuviera el valor de 10, por lo que también lo cambié a un menor o igual a 10, logrando que el programa cumpliera su función.

```

@>"Tabla del 5\r\n"
@>"5 X 1 = 5\r\n"
@>"5 X 2 = 10\r\n"
@>"5 X 3 = 15\r\n"
@>"5 X 4 = 20\r\n"
@>"5 X 5 = 25\r\n"
@>"5 X 6 = 30\r\n"
@>"5 X 7 = 35\r\n"
@>"5 X 8 = 40\r\n"
@>"5 X 9 = 45\r\n"
@>"5 X 10 = 50\r\n"
@>\r\n"
@>"Tabla del 6\r\n"
@>"6 X 1 = 6\r\n"
@>"6 X 2 = 12\r\n"
@>"6 X 3 = 18\r\n"
@>"6 X 4 = 24\r\n"
@>"6 X 5 = 30\r\n"
@>"6 X 6 = 36\r\n"
@>"6 X 7 = 42\r\n"
@>"6 X 8 = 48\r\n"
@>"6 X 9 = 54\r\n"
@>"6 X 10 = 60\r\n"
@>\r\n"
@>"Tabla del 7\r\n"
@>"7 X 1 = 7\r\n"
@>"7 X 2 = 14\r\n"
@>"7 X 3 = 21\r\n"
@>"7 X 4 = 28\r\n"
@>"7 X 5 = 35\r\n"
@>"7 X 6 = 42\r\n"
@>"7 X 7 = 49\r\n"
@>"7 X 8 = 56\r\n"
@>"7 X 9 = 63\r\n"
@>"7 X 10 = 70\r\n"
@>\r\n"
@>"Tabla del 8\r\n"
@>"8 X 1 = 8\r\n"
@>"8 X 2 = 16\r\n"
@>"8 X 3 = 24\r\n"
@>"8 X 4 = 32\r\n"
@>"8 X 5 = 40\r\n"
@>"8 X 6 = 48\r\n"
@>"8 X 7 = 56\r\n"
@>"8 X 8 = 64\r\n"
@>"8 X 9 = 72\r\n"
@>"8 X 10 = 80\r\n"
@>\r\n"
@>"Tabla del 10\r\n"
@>"10 X 1 = 10\r\n"
@>"10 X 2 = 20\r\n"
@>"10 X 3 = 30\r\n"
@>"10 X 4 = 40\r\n"
@>"10 X 5 = 50\r\n"
@>"10 X 6 = 60\r\n"
@>"10 X 7 = 70\r\n"
@>"10 X 8 = 80\r\n"
@>"10 X 9 = 90\r\n"
@>"10 X 10 = 100\r\n"

```

**A-3** El siguiente programa muestra una *violación de segmento* durante su ejecución y se interrumpe; usar un depurador para detectar y corregir la falla.

The screenshot shows the Xcode interface with the following details:

- Top Left:** Activity Monitor showing "Prog3Depurar" PID 3577 with CPU usage at 0%.
- Top Center:** Project Navigator showing "main.c".
- Code Editor:** The code for "main.c" is displayed. Lines 17 and 19 have yellow warning markers indicating a format specification mismatch: "Format specifies type 'int' but the argument has type 'int'".
- Output Window:** Shows the program's output: "EL TERMINO GENERICO DE LA SERIE ES: X^K/K!" followed by "N=". The status bar indicates "No Selection".
- Bottom:** Xcode navigation and search bars.

En primera instancia, *Xcode*, me marca un error en las líneas 17 y 19 “Format specifies type 'int'\* but the argument has type 'int'”, por lo que al revisarlo me di cuenta de que faltaban las “&” en el *scanf*.

The screenshot shows the Xcode interface with the following details:

- Top Left:** Activity Monitor showing "Prog3Depurar" PID 3669 with CPU usage at 0%.
- Top Center:** Project Navigator showing "main.c".
- Code Editor:** The code for "main.c" is displayed. Line 26 has a green breakpoint marker with the annotation "Thread 1: breakpoint 5.1 (3)".
- Output Window:** Shows the program's output: "EL TERMINO GENERICO DE LA SERIE ES: X^K/K!". Below it, variable values are listed: X = (int) 2, K = (int) 2, AP = (int) 2, AS = (float) 5, N = (int) 3. The status bar indicates "No Selection".
- Bottom:** Xcode navigation and search bars.

```
1 // main.c
2 // Prog3Depurar
3 // Created by Cecilia Torres on 12/12/20.
4 //
5
6 //
7
8 #include <stdio.h>
9 #include <math.h>
10
11 void main()
12 {
13     int K, X, AP, N;
14     float AS;
15     printf("EL TERMINO GENERICO DE LA SERIE ES: X^K/K!");
16     printf("\n");
17     scanf("%d", &N);
18     printf("X=%d", X);
19     scanf("%d", &X);
20     K=0;
21     AP=1;
22     AS=0;
23     while(K<=N)
24     {
25         AS=AS+pow(X, K)/AP;
26         K=K+1;
27         AP=AP*K;
28     }
29     printf("SUM=%le", AS);
30 }
31
32
```

No Debug Session      No Selection

EL TERMINO GENERICO DE LA SERIE ES: X^K/K!  
N=3  
X=2  
SUM=6.33333e+00Program ended with exit code: 16

Luego de corregir la falla, volví a correr el programa con algunos breakpoints para así verificar que estuviera funcionando correctamente y no hubiera alguna otra falla.

## Conclusión

Para terminar, el realizar esta práctica me ayudó bastante a entender la importancia de depurar un programa, aunque todavía me cuesta un poco de trabajo utilizar correctamente todas las funciones, específicamente *ejecutar la siguiente instrucción* y *ejecutar la siguiente línea*. Esta práctica la realicé con *Xcode* y aparte hice un ejercicio en *Visual Studio Code*, sin embargo aún así quiero manejar *GDB*, por lo que más adelante practicaré con dicho depurador.

## Fuentes de Consulta

Facultad de Ingeniería, (2018). *Guía práctica de estudio 10: Depuración de un programa*. Recuperado el 13 de diciembre del 2020.