



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Alejandro Pimentel

Profesor:

Asignatura:

Fundamentos de la Programación

Grupo:

3

No de Práctica(s):

10

Integrante(s):

Valeria Patricia Padilla Arellano (2438)

*No. de Equipo de
cómputo empleado:*

50

No. de Lista o Brigada:

34

Semestre:

2020-1

Fecha de entrega:

28 de Octubre del 2019

Observaciones:

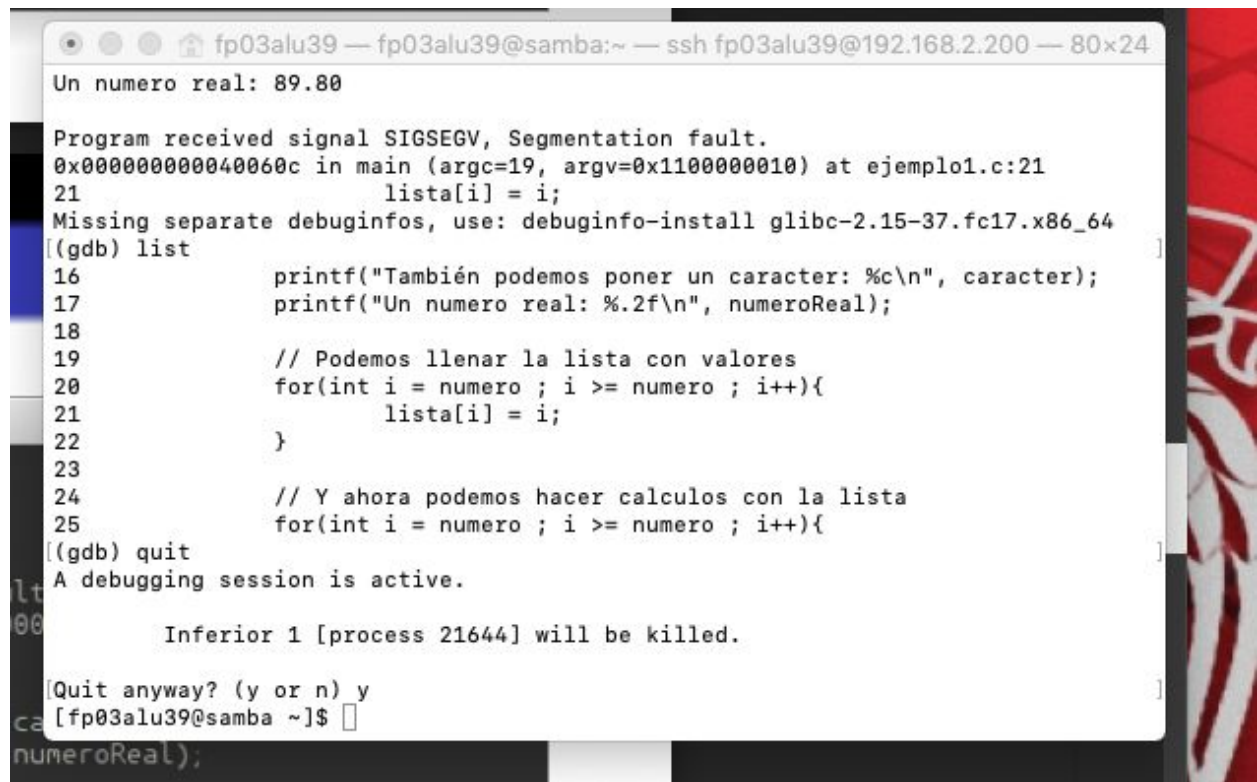
CALIFICACIÓN: _____

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.

INTRODUCCION

Depurar programas permite visualizar con mayor detalle el proceso del programa. Con esto se busca encontrar cualquier error en las líneas de código u optimizar el mismo.



```
fp03alu39 — fp03alu39@samba:~ — ssh fp03alu39@192.168.2.200 — 80x24
Un numero real: 89.80

Program received signal SIGSEGV, Segmentation fault.
0x000000000040060c 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) quit
A debugging session is active.

        Inferior 1 [process 21644] will be killed.

[Quit anyway? (y or n) y
[fp03alu39@samba ~]$
```

```
fp03alu39 — fp03alu39@samba:~ — ssh fp03alu39@192.168.2.200 — 80x24
ejemplo1.c
15     printf("Luego podemos poner un entero: %i\n", numero);
16     printf("También podemos poner un caracter: %c\n", caracter);
17     printf("Un numero real: %.2f\n", numeroReal);
18
9 11     // Podemos llenar la lista con valores
b+> 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++){
26         suma += lista[i];
27     }
Un numero real: 89.80 un caracter: B
child process 22512 In: main Line: 20 PC: 0x40060f
Temporary breakpoint 1 at 0x400542: file ejemplo1.c, line 6.
Starting program: /users/fp03/fp03alu39/ejemplo1

Temporary breakpoint 1, main (argc=1, argv=0x7ffffffe398) at ejemplo1.c:6
Missing separate debuginfos, use: debuginfo-install glibc-2.15-37.fc17.x86_64
(gdb) break 20
Breakpoint 2 at 0x4005f7: file ejemplo1.c, line 20.
(gdb)
```

ACTIVIDAD 1

```
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 }
17
```

Input

Ingresa un número: 4

El resultado es: 4

...Program finished with exit code 20
Press ENTER to exit console.

ACTIVIDAD 2

```
1  #include <stdio.h>
2  #include <math.h>
3
4  void 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("\nN=");
10     //falta la & en los scanf
11     scanf("%i",&N);
12     printf("X=");
13     scanf("%lf",&X);
14     K=0;
15     AP=1;
16     AS=0;
17     //se cambio el k<=N por K<N para que el proceso se haga
18     //N veces
19     while(K<N)
20     {
21         AS=AS+pow(X,K)/AP;
22         K=K+1;
23         AP=AP*K;
24     }
25     printf("Resultado=%le",AS);
26 }
```

ACTIVIDAD 3

```
1  #include <stdio.h>
2
3  int main()
4  { //se agrego una variable porque al final se imprime numero pero
5    //el resultado final de numero es 0 así que se agrego numero2
6    int numero,numero2;
7
8    printf("Ingrese un número:\n");
9    scanf("%i",&numero);
10    //numero2 es igual a numero para mantener su funcion
11    numero2=numero;
12    long int resultado = 1;
13    //se cambio el "numero2>=0" a "numero2>0" para evitar la multiplicacion por 0
14    while(numero2>0){
15        resultado *= numero2;
16        //se cambio la posicion de numero2-- para que se restara hasta el final
17        //y no al principio del proceso iterativo
18        numero2--;
19    }
20
21    printf("El factorial de %i es %li.\n", numero, resultado);
22
23    return 0;
24 }
```

CONCLUSION

El depurar un programa es de gran ayuda para el usuario ya que nos permite crear un programa sin errores y si se encuentra uno, ayuda a minimizar el proceso de búsqueda para encontrarlo.