Universidad de Costa Rica
Escuela de Ingeniería Eléctrica
Programación Bajo Plataformas Abiertas
MSc. Andrés Mora Zúñiga
I Ciclo 2020
IE-0117

Práctica # 3: Programas C

1. Numero Par

Para esta sección se realizaron 3 archivos. El primero corresponde al archivo odden.c el cual corresponde a:

```
#include <stdio.h>
#include "include/functions.h"
#include<stdlib.h>

//Nota el programa se diseno para ingresar solo 1 numero

int main (int argc, char const *argv[])

{

    int numero = atoi(argv[1]);

    numeropar( numero);


    return 0;
}
```

El código empieza incluyendo la biblioteca stdio.h como es frecuente. Posteriormente es necesario incluir el archivo functions.h y especificarlo con su ruta relativa ya que este define a función que se utiliza. Además se importa stdib.h para importar los datos de la terminal. Básicamente dentro del main lo único que se hace es convertir el numero que entra de la terminal de una hilera de caracteres a un numero entero. Posteriormente se llama a la función numeropar y se retorna un cero.

Luego se tiene el código del archivo functions.h:

```
#if !defined(FUNCTIONS)
#define FUNCTIONS

int numeropar(int numero);

#endif // FUNCTIONS
```

Este archivo lo que hace es definir la función numeropar. Por último se muestra el archivo functions.c:

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

```
2  #include "functions.h"
3
4  int numeropar(int numero)
5  {
6      int modulo;
7      modulo= numero %;
8
9      switch (modulo)
10     {
11         case 0:
12             printf("El numero es par\n");
13             break;
14
15         default:
16             printf("El numero es impar\n");
17             break;
18     }
19
20     return 0;
```

En este archivo llamado functions.c se muestra la función que se realizó. Primeramente se incluye la biblioteca stdio.h y además para definirla el archivo functions.h(nota que la ruta relativa cambia porque estos 2 archivos están en directorio include). Posteriormente consiste en hacer la función. La función consiste en que se reciba un parámetro enviado desde la terminal(./salida.x #), este correspondería a la variable numero. A este numero se le saca el modulo de 2 y por medio de un switch se condiciona a que si el modulo corresponde a cero, imprime que es un número par. Utilizando default en caso de que no sea par, imprime que es impar.En la figura 1 se muestra en la terminal el programa en ejecución con varios ejemplos.

Figura 1: Numero par

2. Numero menor

Para esta sección se realizaron 3 archivos igualmente. El primero corresponde al archivo less.c el cual corresponde a:

```
1
2  #include <stdio.h>
```

```
#include "include/functions.h"
#include<stdlib.h>
int main (int argc, char const *argv[])

{

    int numero1 = atoi(argv[1]);
    int numero2 = atoi(argv[2]);
    int numero3 = atoi(argv[3]);


    numeromenor( numero1,numero2,numero3);


    return 0;
}
```

El código empieza incluyendo la biblioteca stdio.h como es frecuente. Posteriormente es necesario incluir el archivo functions.h y especificarlo con su ruta relativa ya que este define a función que se utiliza. Además se importa stdib.h para importar los datos de la terminal. Básicamente dentro del main lo único que se hace es convertir los 3 numero que entran de la terminal de una hilera de caracteres a números enteros. Posteriormente se llama a la función numeromenor y se retorna un cero.

Luego se tiene el código del archivo functions.h:

```
#if !defined(FUNCTIONS)
#define FUNCTIONS

int numeromenor( int numero1, int numero2, int numero3);

#endif // FUNCTIONS
```

Este archivo lo que hace es definir la función numeromenor. Por último se muestra el archivo functions.c:

```

#include <stdio.h>
#include "functions.h"

int numeromenor( int numero1, int numero2, int numero3)
{
    int x=0;

    if(numero1<numero2 && numero1<numero3){
        x++;
        printf("El numero menor es %d\n",numero1);}
    if(numero2<numero1 && numero2<numero3){
        x++;
        printf("El numero menor es %d\n",numero2);}
    if(numero3<numero2 && numero3<numero1){
        x++;
        printf("El numero menor es %d\n",numero3);}
    if(x==0){

```

```
20          printf("No hay numero menor ya que 2 o mas numeros son iguales\n");}
21
22
23      return 0;
24  }
```

En este archivo llamado functions.c se muestra la función que se realizó. Primeramente se incluye la biblioteca stdio.h y además para definirla el archivo functions.h(nota que la ruta relativa cambia porque estos 2 archivos están en directorio include). Posteriormente consiste en hacer la función. La función consiste en que se reciba tres parámetro enviado desde la terminal(./salida.x x y z), estas corresponderían a las variables numero1, numero2 y numero3. Por medio de if se comparan los números diciendo que en caso de que algún numero sea menor que los otros 2 se imprima que ese es el numero menor. En caso de que 2 números se repitan y sean los menores el programa imprime que se repiten 2 o mas números y por ende no hay uno menor a los otros 2.En la figura 2 se muestra en la terminal el programa en ejecución con varios ejemplos.

```
alberto@debian:~/laboratorio3/numeromenor$ gcc -o salida.x less.c include/functions.c
alberto@debian:~/laboratorio3/numeromenor$ salida.x 1 2 3
bash: salida.x: orden no encontrada
alberto@debian:~/laboratorio3/numeromenor$ ./salida.x 1 2 3
El numero menor es 1
alberto@debian:~/laboratorio3/numeromenor$ ./salida.x -2000 4 600
El numero menor es -2000
alberto@debian:~/laboratorio3/numeromenor$ ./salida.x 22 22 7
El numero menor es 7
alberto@debian:~/laboratorio3/numeromenor$ ./salida.x 22 22 22
No hay numero menor ya que 2 o mas numeros son iguales
alberto@debian:~/laboratorio3/numeromenor$ ./salida.x 2 2 70
No hay numero menor ya que 2 o mas numeros son iguales
alberto@debian:~/laboratorio3/numeromenor$
```

Figura 2: Numero menor

3. Numero mayor

Básicamente para este punto se realizó lo mismo que para el pasado, con la diferencia que ahora el método es numeromayor y se compara cual es mayor en vez de menor.Se realizaron 3 archivos igualmente. El primero corresponde al archivo great.c el cual corresponde a:

```
1
2   #include <stdio.h>
3   #include "include/functions.h"
4   #include<stdlib.h>
5   int main (int argc, char const *argv[])
6
7   {
8
9       int numero1 = atoi(argv[1]);
10      int numero2 = atoi(argv[2]);
11      int numero3 = atoi(argv[3]);
12
13
```

```
14      numeromayor( numero1,numero2,numero3);
15
16
17      return 0;
18  }
```

El código empieza incluyendo la biblioteca stdio.h como es frecuente. Posteriormente es necesario incluir el archivo functions.h y especificarlo con su ruta relativa ya que este define a función que se utiliza. Además se importa stdib.h para importar los datos de la terminal. Básicamente dentro del main lo único que se hace es convertir los 3 numero que entran de la terminal de una hilera de caracteres a números enteros. Posteriormente se llama a la función numeromayor y se retorna un cero.

Luego se tiene el código del archivo functions.h:

```
1  #if !defined(FUNCTIONS)
2  #define FUNCTIONS
3
4  int numeromayor( int numero1, int numero2, int numero3);
5
6  #endif // FUNCTIONS
```

Este archivo lo que hace es definir la función numeromayor. Por último se muestra el archivo functions.c:

```
1
2  #include <stdio.h>
3  #include "functions.h"
4
5  int numeromayor( int numero1, int numero2, int numero3)
6  {
7      int x=0;
8
9      if(numero1>numero2 && numero1>numero3){
10         x++;
11         printf("El numero mayor es %d\n",numero1);}
12     if(numero2>numero1 && numero2>numero3){
13         x++;
14         printf("El numero mayor es %d\n",numero2);}
15     if(numero3>numero2 && numero3>numero1){
16         x++;
17         printf("El numero mayor es %d\n",numero3);}
18     if(x==0){
19
20         printf("No hay numero mayor ya que 2 o mas n meros son iguales\n");}
21
22
23     return 0;
24 }
```

En este archivo llamado functions.c se muestra la función que se realizó. Primeramente se incluye la biblioteca stdio.h y además para definirla el archivo functions.h(nota que la ruta relativa cambia porque estos 2 archivos están en directorio include). Posteriormente consiste en hacer la función. La función consiste en que se reciba tres parámetro enviado desde la

terminal(./salida.x x y z), estas corresponderían a las variables numero1, numero2 y numero3. Por medio de if se comparan los números diciendo que en caso de que algún numero sea mayor que los otros 2 se imprima que ese es el numero mayor. En caso de que 2 números se repitan y sean los mayores el programa imprime que se repiten 2 o mas números y por ende no hay uno mayor a los otros 2.En la figura 3 se muestra en la terminal el programa en ejecución con varios ejemplos.

```
alberto@debian:~/laboratorio3/numeromayor$ gcc -o salida.x great.c include/functions.c
alberto@debian:~/laboratorio3/numeromayor$ ./salida.x 1 2 3
El numero mayor es 3
alberto@debian:~/laboratorio3/numeromayor$ ./salida.x -100 1 2
El numero mayor es 2
alberto@debian:~/laboratorio3/numeromayor$ ./salida.x 9 9 1
No hay numero mayor ya que 2 o 3 numeros son iguales
alberto@debian:~/laboratorio3/numeromayor$ ./salida.x 9 1 1
El numero mayor es 9
alberto@debian:~/laboratorio3/numeromayor$ ./salida.x 9 9 9
No hay numero mayor ya que 2 o 3 numeros son iguales
alberto@debian:~/laboratorio3/numeromayor$
```

Figura 3: Numero mayor

4. Ecuación Cuadrática

Para esta sección se realizaron 3 archivos igualmente. El primero corresponde al archivo eq.c el cual corresponde a:

```
1  #include <stdio.h>
2  #include "include/functions.h"
3  #include<stdlib.h>
4  #include<math.h>
5  int main (int argc, char const *argv[])
6
7  {
8
9      int a = atoi(argv[1]);
10     int b = atoi(argv[2]);
11     int c = atoi(argv[3]);
12
13     ecuacioncuadratica(a,b,c);
14
15
16     return 0;
17 }
```

El código empieza incluyendo la biblioteca stdio.h como es frecuente. Posteriormente es necesario incluir el archivo functions.h y especificarlo con su ruta relativa ya que este define a función que se utiliza. Además se importa stdib.h para importar los datos de la terminal.A diferencia de los otros casos en este también se importa la biblioteca math.h para realizar la raíz cuadrada. Básicamente dentro del main lo único que se hace es convertir los 3 numero que entran de la terminal de una hilera de caracteres a números enteros. Posteriormente se llama a la función ecuacioncuadratica y se retorna un cero.

Luego se tiene el código del archivo functions.h:

```
#if !defined(FUNCTIONS)
#define FUNCTIONS

int ecuacioncuadratica(int a, int b, int c);

#endif // FUNCTIONS
```

Este archivo lo que hace es definir la función ecuacioncuadratica. Por último se muestra el archivo functions.c:

```

#include <stdio.h>
#include "functions.h"
#include<math.h>

int ecuacioncuadratica(int a, int b, int c)
{
    float discriminante=(b*b)-(4*a*c);

    if(discriminante<0){
        printf("No hay soluciones reales\n");
    }
    else{

       float sol1=(-b+sqrtf(discriminante))/(2*a);
       float sol2=(-b-sqrtf(discriminante))/(2*a);
        if(sol1==sol2){
            printf("Poseen la misma solucion, la cual es:%f\n",sol1);
        }else
        {
          printf("Poseen diferente solucion.\n La solucion 1 es:%f\n",sol1);
          printf("La solucion 2 es :%f\n",sol2);
        };

    }



    return 0;
}
```

En este archivo llamado functions.c se muestra la función que se realizó. Primeramente se incluye la biblioteca stdio.h y además para definirla el archivo functions.h(nota que la ruta relativa cambia porque estos 2 archivos están en directorio include).Además se agrega también la biblioteca math.h. Posteriormente consiste en hacer la función. La función consiste en que se reciba tres parámetro enviado desde la terminal(./salida.x x y z), estas corresponderían a las variables a, b y c. Primeramente se calcula el discriminante. Por medio de un if se tiene que si el discriminante es menor a cero que imprima que no hay solución real.Dentro del else ligado al if pasado se prosigue con el calculo. Aplicando la formula general donde la solución 1 equivale cuando la raíz del discriminante se suma y la solución 2 cuando se resta. Se abre otro if con condiciones que si las dos soluciones son iguales se se imprima que poseen

la misma solución y se imprima cual es. Por último por medio del else ligado a ese último if se tiene que se imprime primero la solución 1 y posteriormente la solución 2.En la figura 4 se muestra en la terminal el programa en ejecución con varios ejemplos. Recordando que se necesita compilar con -lm.

```
alberto@debian:~/laboratorio3/eqcuadratica$ gcc -o salida.x eq.c include/functions.c
/usr/bin/ld: /tmp/ccPxip0r.o: en la función `ecuacioncuadratica':
functions.c:(.text+0x61): referencia a `sqrtf' sin definir
/usr/bin/ld: functions.c:(.text+0x92): referencia a `sqrtf' sin definir
collect2: error: ld returned 1 exit status
alberto@debian:~/laboratorio3/eqcuadratica$ gcc -o salida.x eq.c include/functions.c
lm
alberto@debian:~/laboratorio3/eqcuadratica$ ./salida.x 1 2 1
Poseen la misma solucion, la cual es:-1.000000
alberto@debian:~/laboratorio3/eqcuadratica$ ./salida.x 8 2 1
No hay soluciones reales
alberto@debian:~/laboratorio3/eqcuadratica$ 1 8 4
bash: 1: orden no encontrada
alberto@debian:~/laboratorio3/eqcuadratica$ ./salida.x 1 8 4
Poseen diferente solucion.
 La solucion 1 es:-0.535898
La solucion 2 es :-7.464102
```

Figura 4: Soluciones ecuacion cuadratica