TPN°5: Funciones y Biblioteca Estándar

Si se invoca alguna función de la biblioteca matemática se debe linkeditar con la opción –lm.

Ejercicio 1

Suponiendo que cada código fuente es el único módulo que forma un programa ejecutable, indicar para cada variable su ***persistencia***, ***alcance*** y ***visibilidad***, especificando si se la está declarando o definiendo. Si hay errores indicarlos y corregirlos. Una vez corregido realizar el seguimiento de cada código e indicar qué valores se imprimen en cada caso:

a)

```
#include <stdio.h>

int a,b ;
void local ( void );

int
main ( void )
{
      a = 2; b = 3;

      local();
      printf(" a vale : %d\tb vale %d\n",  a, b );
      return 0;
}

void
local ( void )
{
      int a;

      a =  -5;
      b = 10;

      return ;
}
```

b)

```
#include <stdio.h>
int a, b, c;

void primero ( void );
void segundo ( void );

int
main ( void )
{
      a = 1;
      printf("El valor de c es %d", c );
      segundo();
      printf ("El valor de a es %d, el de b es %d",a , b );
      primero();
      printf("El valor de a es %d, el de c es %d", a, c );
      return 0;
}

void
```

```
primero ( void )
{
      int a ;
      a = 3;
      c = 0;
}

void
segundo ( void )
{
      int a ;
      c = 0;
      a = 2;
      b = -a;
}
```

c)

```
#include <stdio.h>

char letra;
void segundo ( void );

int
main( void )
{
      printf( "Ingrese un carácter : ");
      letra = getchar();
      segundo();
      printf( " letra es : %c\n", letra );
      return 0;
}

void
segundo( void )
{
      char letra;
      letra = 'X';
      return;
}
```

d)

```
#include <stdio.h>

static int  m
int b;
static void este(void);

int
main(void)
{
      m = 2;
      b = 3;

      este();
      printf("m vale : %d", m );
      printf("b vale : %d", b );

      return 0;
}
```

```
static void
este(void)
{
      static int m;

      m++;
      b = - 3;
      printf( "Dentro de este() m vale %d y b vale %d", m, b);

      return;
}
```

e)

```
#include <stdio.h>

static int cubo(int num);

int
main( void )
{
      int x;

      for ( x=1; x<=5; x++)
            printf("El cubo de %d es %4d\n", x, cubo(x));

      return 0;
}

static int
cubo(int num)
{
      return num * num * num;
}
```

Ejercicio 2

Decir, sin necesidad de ejecutar el código, qué imprime cada uno de los siguientes fragmentos de código, asumiendo que la variable **i** es de tipo **int**, la variable **x** es de tipo **double** y que se incluyeron los archivos de encabezado que corresponden.

a)

```
for ( i = 46; i <= 'H'; i++)
{
      printf("%d %c %c\t", i, i, tolower(i));
      printf("%s ", isalnum(i)? "si" : "no");
      printf("%s ", isalpha(i)? "si" : "no");
      printf("%s ", isdigit(i)? "si" : "no");
      printf("%s ", islower(i)? "si" : "no");
      printf("%s ", isupper(i)? "si" : "no");
      printf("%s ", isxdigit(i)? "si" : "no");
      printf("%s ", iscntrl(i)? "si" : "no");
      printf("%s \n", ispunct(i)? "si" : "no");
}
```

b)

```
for ( x = -1; x < 2; x += 0.5 )
{
      printf("%f  %f   %f   %f\n", x, fabs(x), ceil(x), floor(x));
      printf("%f  %f\n", pow(x,2), pow(x,.5));
}
```

Ejercicio 3

El siguiente programa detecta si un boleto de 5 cifras es capicúa, para ello utiliza una función llamada **lugar** que dado dos números ***n*** y ***k***, retorna el k-ésimo dígito contando desde la derecha del número ***n***. Los ceros a izquierda se descartan.

```
#include <stdio.h>
#include "getnum.h"
#include <math.h>

int lugar (int num, int pos);

int
main( void )
{
      int boleto, esCapicua=1;
      int  j;

      boleto = getint("Ingrese un número de 5 dígitos:”);

      for (j=1; j<=2; j++)
            if ( lugar(boleto, j) != lugar(boleto,5-j+1))
            {
                  esCapicua=0;
                  break;
            }
      printf("El boleto %ses capicúa",(esCapicua)? "":"no ");
      return 0;
}

int
lugar( int num, int pos )
{
      return (num / (int)pow(10, pos-1)) % 10;
}
```

a) Agregar una decisión de forma tal que si ingresan un número que no tenga exactamente 5 dígitos, el programa finalice sin hacer nada.

b) Modificar el programa original de forma tal que si ingresa un número que no sea de 5 dígitos imprima el cartel "Ingreso incorrecto de datos".

c) Modificar el programa del punto b) de forma tal que siga leyendo números hasta que se ingrese un número de exactamente 5 dígitos.

d) Modificar el programa original (sin las correcciones a, b y c) de forma tal que sirva para números de cualquier cantidad de cifras.

e) Modificar el programa d) de forma tal que, además de imprimir el mensaje, devuelva el valor 1 en caso de ser capicúa y 0 en caso de no serlo.

Ejercicio 4

Escribir una función que reciba dos parámetros de entrada de tipo entero y devuelva en su nombre un numero aleatorio entre ambos números. (En una línea)

Ejercicio 5

Escribir una función **potencia**, que reciba en dos parámetros de entrada la **base (de tipo double)** y el **exponente (entero),** y devuelva el valor de dicha potencia (En no más de 10 líneas).

Por ejemplo, **potencia(2,-4)** debe devolver 0.0625. En caso de no poder calcularse el resultado debe devolver –1.

Ejercicio 6

Escribir un programa que ofrezca las siguientes opciones para accionar sobre un caracter leído desde la entrada estándar. Utilizar distintas funciones (con no más de 4 líneas) para cada una de las opciones y una función para el menú de opciones:

- **Convertir de mayúscula a minúscula.**
- **Convertir de minúscula a mayúscula.**
- **Imprimir el carácter siguiente.**
- **Imprimir la siguiente letra en forma circular (si la letra es 'z', debe imprimir 'a')**

Ejercicio 7

Escribir una función real para calcular aproximadamente el valor de e^x por medio de la serie $1 + x + x^2 / 2! + x^3 / 3! + ...$ Dicha función recibe como parámetros de entrada el valor de x. (En no más de 10 líneas)

Hacer un programa que invoque a la función y escriba el resultado de la misma y el valor de la función exp(x) de C.

Ejercicio 8

La función **floor** puede ser utilizada para redondear un numero a una cantidad especifica de lugares decimales. Por ejemplo, **floor(x * 100 + .5) / 100** redondea ***x*** a la posición de los centésimos.

a) Escribir una función redondeo que reciba dos parámetros de entrada correspondientes al número a redondear y la cantidad de cifras decimales que se desean, y que devuelva en su nombre el número redondeado (en no más de 5 líneas).

b) Escribir un programa que ofrezca el siguiente menú para redondear un numero x de varias formas e utilice la función anterior para cada una de ellas:
   - **Redondeo al entero más cercano**
   - **Redondeo a la décima mas cercana.**
   - **Redondeo a la centésima mas cercana.**
   - **Redondeo a la milésima mas cercana.**

Ejercicio 9

Escribir un programa (en no más 15 líneas) para encontrar raíces de funciones en un intervalo dado. Se deberá recorrer el intervalo a incrementos de 0.001, evaluando la función en cada paso y escribiendo por salida estándar los puntos que son raíces. Los extremos del intervalo serán de tipo real y su valor estará dado por constantes del programa.

Tener en cuenta no sólo el caso en el que el resultado de evaluar la función sea cero, sino también aquellos puntos en los cuales la función cambia de signo.

Ejecutar el programa para las siguientes funciones:

- **Seno(x)**
- **Coseno(x)**
- **$2^x * x^3$**
- **Seno(x) + Log(x)**
- $\sqrt[2]{X} + e^x$

Ejercicio 10

Escribir un programa que ofrezca un menú con las siguientes opciones:

- **Simplificar una fracción**
- **Sumar dos fracciones**
- **Terminar la ejecución**

Si se elige la primera opción se le solicitará numerador y denominador de la fracción y se imprimirá la fracción simplificada (algunos de ellos podrían ser negativos). Si se opta por la segunda, se le solicitará dos fracciones y se imprimirá la suma de las mismas, también simplificada. Obviamente con la tercera opción finaliza la ejecución del mismo. Escribir una función para cada item del menú. (No más de 20 líneas para **main** y no más de 10 líneas para las funciones del menú).

Pista: Escribir y usar la función DivisorComunMaximo.

Ejercicio 11

Reescribir el ejercicio 5 del TP 3 utilizando funciones de la biblioteca estandar.

Ejercicio 12

Reescribir el ejercicio 24 del TP 3 utilizando funciones de la biblioteca estandar.

Ejercicio 13

Dados tres archivos func.h, func.c y main.c (que obviamente contiene la función main y la invocación a funciones definidas en el archivo func.c), explicar:

a) Que sentido tiene agregar la directiva #include “func.h” en el archivo main.c

b) Que sentido tiene agregar la directiva #include “func.c” en el archivo main.c

c) Cual sería la secuencia de comandos a ejecutar para generar a partir de dichos archivos un programa ejecutable.

Ejercicio 14

Un número IP identifica en una red TCP/IP a un *host* particular (puede ser una computadora, impresora, router, cámara de seguridad, etc.). Todo número IP está compuesto de 32 bits, donde parte de ese número (una cantidad arbitraria de bits a la izquierda del número) identifican la red a la que pertenece y el resto de los bits identifican al *host*. Si bien el número se almacena como un entero sin signo de 4 bytes, se lo presenta al usuario en 4 grupos de números decimales

Escribir una función que reciba:

- **ip**: unsigned long con el número IP de un host
- **bitsNet**: unsigned char que indica cuántos bits se utilizan para identificar a la red

La función debe imprimir en salida estándar el número de red y el número de host correspondientes al parámetro **ip**.

Ejemplos

ip	bitsNet	Red	Host
0xC0A80064 (192.168.0.100)	16	192.168.0.0	0.0.0.100
0x10FF1112 (16.255.17.18)	24	16.255.17.0	0.0.0.18
0x10FF1112 (16.255.17.18)	23	16.255.16.0	0.0.1.18
0x10FF1112 (16.255.17.18)	16	16.255.0.0	0.0.16.18