PRÁCTICA 2
FUNCIONES

F1) Crear un programa que permita pasar una temperatura en Celsius y la convierta en Farenheit mediante una función. Crear además la función que realice la acción inversa.

F2) Implementar una función que dado un entero determine si es capicúa, es decir que el valor del mismo sea el mismo si de lee del derecho o el revés.

F3) Realizar un programa que permita obtener los enteros primos entre 1 y 500.
Implementar una función para determinar si un número es o no primo.

F4) Crear un programa que permita pasar una cantidad de días y horas (especificados con enteros positivos, validar esto) a minutos.

F5)
a) Ejecutar el siguiente programa y observar la salida

```
#include <stdio.h>

int func();

int main() {
    int i=0;
    for(i=0; i<10; i++) {
         printf("%d ", func());
    }
    return 0;
}

int func() {
   static b=0;
   return b++;
}
```

Buscar en bibliografía información acerca de "variables estáticas".

b) En base al punto anterior, crear un programa que utilice una función que en invocaciones sucesivas genere retorne los valores enteros 0, 1, 2, 3, 0, 1, 2, 3, …., 0, 1, 2, 3, etc.

F6) Crear un set de funciones que permitan calcular lo siguiente de un triángulo.
En primer lugar el usuario debe elegir si especifica el triángulo por su base y altura o por sus tres lados.
Si lo hace por su base y altura debe ser posible conocer:

- el perímetro
- la superficie

si el usuario elige especificar el triángulo con sus tres lados, debe ser posible conocer:
(validar si dichos lados forman un triángulo primero dado su base y altura:
http://es.wikihow.com/saber-si-hay-un-tri%C3%A1ngulo-teniendo-las-medidas-de-tres-lados)

- el perímetro
- categoría: equilátero, isósceles o escaleno

ARRAYS

A1) Crear un programa que cargue un array con los primeros 50 números enteros positivos, luego mostrarlo por pantalla.

A2) Crear un programa que permita mediante un **for** cargar con números enteros aleatorios entre 1 y 100 un array de 20 componentes, mostrarlo por pantalla y calcular la suma de todas las componentes.

A3) Escriba un programa que permita cargar un array con 30 enteros aleatorios entre -10 y 10 y que reciba un entero "e" ingresado por el usuario:
a) determinar si dicho entero "e" está o no en el array
b) en otra función indicar las ocurrencias de "e" en el mismo array

A4) Crear un programa que dado un array de 15 componentes enteras y dos números enteros ingresado por el usuario, por ejemplo **n** y **m**, haga lo siguiente:
- la primer componente del array es el número **n**
- las segunda componente del array es el número **m**
- cada componente restante de establece como la suma de las dos anteriores

A5) Escriba un programa que permita ordenar un array de 10 enteros siguendo algún algoritmo de ordenamiento (se dará documentación al respecto).

A6) Juego con un dado
Crear un progama que permita emular el lanzamiento de un dado.
El programa debe permitir al usuario "apostar" a un número del 1 al 6 (validar rango) e indicar cuantas veces se lanzará el dado (mínimo 10 veces, máximo 50 veces, validarlo).
Si el número que indicó es el que más veces apareció recibirá el mensaje de "excelente", si su número elegido fue el segundo en orden de apariciones el mensaje será "muy bueno" y luego los mensajes serán "bueno", "regular", "malo" y "muy malo", dependiendo de la cantidad de ocurrencias del número que arriesgó respecto a los otros. Mostrar también la cantidad de apariciones de cada número.

Notas: emplear generación de números aleatorios para representar los lanzamientos del dado.
considerar el uso de paso de argumentos por referencia.
considerar ordenamiento de arrays para facilitar el ejercicio.

A7) Crear un programa que declare dos arrays de 10 enteros.
Mediante una función llamada **cargaAleatArray** el primero debe ser instanciado con valores aleatorios (0 a 100) y lo muestre por pantalla.
Mediante otra función **cargaManualArray** el segundo array debe ser instanciado con valores que el usuario suministre y lo muestre luego por pantalla.
Debe existir una función **mostrarArray**.
Con una última función que recibe ambos arrays como argumento, mostrar por pantalla la suma de las componentes homólogas de cada array, función **sumaArrays**.

A8) Implementar una función que al recibir un array de N enteros lo muestre invertido.
N debe ser un valor definido mediante macro, elija N>=2.
El array debe ser ingresado por el usuario.

RECURSIVIDAD

R1) Crear las siguientes funciones recursivas
suma(a, b) = a + b
prod(a, b) = a * b
exp(a, b) = a^b
factorial(n) = n!
fib(n) = n-ésimo término de la sucesión de Fibonacci (1, 2, 3, 5, 8, 13, 21, 35, 55,…)

R2) Repetir el ejercicio anterior pero empleando recursividad de cola.

R3) Crear una función recursiva para sumar los elementos de un array.

R4) Crear una función recursiva para invertir un array.

R5) Implementar el algoritmo de ordenación quick-sort.