Tema C

Hacer primero Ej 1 y Ej 2, el Ej 3 es solo para sacar B+, es decir se considerará si todo lo demás esta perfecto.
Para compilar un archivo .c escribir en la terminal en la carpeta donde esta el archivo: $> gcc -Wall -Wextra -std=c99 miarchivo.c -o miprograma
Para ejecutar escribir:
$> ./miprograma

Ejercicio 1

Considerar la siguiente asignación múltiple:

```
var x, y, z : Int;
{Pre: x = X, y = Y, z = Z, X > 0, Y > 0, Z > 0}
x, y, z := y + z, z * x, x / y
{Pos: x = Y + Z, y = Z * X, z = X / Y}
```

Escribir un programa en lenguaje C equivalente usando asignaciones simples teniendo en cuenta que:

- Se deben verificar las pre y post condiciones usando la función `assert()`.
- Los valores iniciales de `x`, `y`, `z` deben ser ingresados por el usuario ● Los valores finales de `x`, `y`, `z` deben mostrarse por pantalla.

Ejercicio 2

Programar las siguientes funciones:

a)

```
void llenar_con_temperaturas(float a[], int tam);
```

que dado un arreglo `a[]` de tamaño `tam`, pide al usuario ingresar los valores (float) del arreglo. Cada valor representa una temperatura en grados Celsius.

b)

```
bool hay_mas_de_2_altas(float a[], int tam);
```

que dado un arreglo `a[]` de tamaño `tam`, devuelve `true` sólo si en el arreglo hay por lo menos tres temperaturas mayores o iguales a 30.0 grados Celsius y menores o iguales a

45.0 grados Celsius. Por ejemplo:

a[]	tam	resultado Comentario
[28.5, 32.0, 29.5, 35.2, 40.0]	5	**true** Ya que hay más de 3 temperaturas mayores o iguales a 30.0 y menores o iguales a 45.0 en el arreglo.
[21.0, 30.5, 29.0, 31.0, 21.3]	5	**false** Ya que sólo hay 2 temperaturas mayores o iguales a 30.0 y menores o iguales a 45.0 en el arreglo.
[25.0, 47.0, 29.5, 15.0, 17.4]	5	**false** Ya que hay 0 temperaturas mayores o iguales a 30.0 y menores o iguales a 45.0 en el arreglo.
[30.0, 44.0, 31.5, 42.0, 29.5, 33]	5	**true** Ya que hay 3 temperaturas mayores o iguales a 30.0 y menores o iguales a 45.0 en el arreglo.

Cabe aclarar que la función `hay_mas_2_altas` no debe mostrar ningún mensaje por pantalla ni pedir valores al usuario.

c)

En la función `main` se debe:

- declarar un arreglo de longitud `N`. Definir a `N` como una constante, **el usuario no debe elegir el tamaño del arreglo**. Recordar que las constantes se definen al principio del archivo usando ***#define***
- Verificar con assert que N sea mayor estricto que 0.
- Llamar a la función *`llenar_con_temperaturas`*
- Llamar a la función *`hay_mas_de_2_altas`*
- Mostrar el resultado de *`hay_mas_de_2_altas`* por pantalla.
- Dejar un par de ejemplos de ejecución como comentario en el código

Ejercicio 3*

Hacer un programa que cuente la cantidad de:

- temperaturas bajas: menor a los 15 grados celsius
- temperaturas medias: entre los 15 y los 30 grados celsius (extremos incluidos)
- temperaturas altas: mayores a los 30 grados celsius

```
s_temperaturas totales(float a[], int tam);
```

donde la estructura **`struct`** `s_temperaturas` se define de la siguiente manera:

```
typedef struct {
    int cuantas_bajas;
    int cuantas_medias;
    int cuantas_altas;
} s_temperaturas;
```

La función toma un arreglo `a[]` y su tamaño `tam`, y devuelve una estructura con tres enteros que respectivamente indican: la cantidad de ‘temperaturas bajas’ (cuantas_bajas), la cantidad de ‘temperaturas medias’ (cuantas_medias) y la cantidad de ‘temperaturas altas’ (cuantas_altas) hay en el arreglo a[ ]. La función `totales` debe implementarse con un único ciclo y **no debe mostrar mensajes** por pantalla **ni pedir valores al usuario**.

En la función `main` declarar un arreglo de longitud `N` (definir a `N` como una constante, el usuario no debe elegir el tamaño del arreglo), llenarlo con la función *`llenar_con_temperaturas`*, llamar a la función totales y luego mostrar el resultado de la función por pantalla (los tres valores).