Objetivo de la práctica:
- Practicar el uso de vectores y matrices. Aplicar los conceptos vistos con anterioridad principalmente referentes a las correcta modularización del programa

Conceptos básicos: vectores, matrices y array multidimensionales.

Un *array* es una secuencia de objetos del mismo tipo. Estos objetos se llaman elementos del *array* y se numeran consecutivamente 0, 1, 2, 3, ... El acceso a cada uno de los elementos de un vector se realiza mediante un índice.
El tipo de elementos almacenados en el array puede ser cualquier tipo de dato C++.
Adoptaremos el convenio de emplear **`typedef`** para crear este tipo de dato estructurado:
Cuando el array necesita un solo índice para ser definido se llama ***vector*** y lo declararemos de la siguiente forma:

```
const int MAX = 20;

typedef int Vector20[MAX]; //define un vector de 20 enteros

Vector20 mi_vector; //declaro un vector de 20 enteros.
```

```
Ejemplo de acceso a un elemento del vector: mi_vector[3] = 45;
    //asigna el valor 45 a la cuarta componente
```

Cuando el array necesita 2 índices para ser definido se llama ***matriz*** y lo declararemos de la siguiente forma:

```
const int FIL = 10;
const int COL = 4;

typedef int Matriz10x4[FIL][COL];

Matriz10x4 mi_matriz;
```

```
Ejemplo de acceso a un elemento de la matriz: mi_matriz[1][3] = 32;
```

Cuando el array necesita más de 2 índices para ser definido se llama ***array multidimensional:***

```
const int FILA1 = 10;
...
const int FILAN =12;

typedef int Array_n_columnas[FILA1][FILA2]. . . .[FILAN];

Array_n_columnas mi_array_n_columnas;
```

```
Ejemplo de acceso a un elemento de la matriz:
    mi_array_n_columnas [1][2] . . .[10] = 20;
```

Los arrays **se pasan por defecto por referencia** como argumentos de una función. La única diferencia con los tipos simples es que no se usa él '&' ya que basta simplemente el nombre del array. Para pasar un array únicamente como parámetro de entrada se pone la palabra **const** delante del tipo del array.

SOBRE LA PRÁCTICA ANTERIOR:

Prueba el programa en el fichero **`ejErrores_p6_FP[05].cpp`** (accesible en la página web de la asignatura); si te parece que no funciona correctamente modifícalo hasta que funcione como debería.

PROBLEMAS

NOTA: Se valorará la correcta división en módulos de todos los ejercicios planteados.

1. Realiza un programa que calcule la frecuencia de aparición de los distintos caracteres que aparecen en una frase introducida por teclado. Es decir, se deben contar el número de 'a', 'b', 'c', etc..., sin distinguir mayúsculas y minúsculas.

2. Escribir un programa que realice la suma de dos números enteros positivos muy grandes (máximo 200 cifras) y del el resultado exacto.
   Nota: no se pueden utilizar valores reales (coma flotante). En ese caso, el resultado es una aproximación.
   Nota 2: Pensad claramente la manera de guardar el número en el vector de manera que se facilite la suma.

3. Una aplicación muy interesante de las matrices es el tratamiento digital de imágenes. Una imagen digital (en blanco y negro) se puede representar por una matriz de tamaño `nxm`, donde cada posición es un entero en el intervalo [0, 255] que contiene la luminancia en ese punto (0 = negro, 255 = blanco). Realiza un programa que:
   a. Permita leer una imagen desde teclado de tamaño `nxm` variable.
   b. Muestre por pantalla la imagen (matriz).
   c. Dada la imagen, calcule una imagen binaria (exclusivamente con valores 0 o 1) de forma que todos los puntos de la imagen (elementos de la matriz) con valor de luminancia superior a un umbral dado tomarán el valor 1 y todos los que sean inferiores tomarán el valor 0. El usuario introducirá el valor umbral por teclado.
   d. Una vez calculada la imagen binaria, y conociendo la imagen original, guardar en un vector los valores (no coordenadas) de la imagen original que valgan 1 en la imagen binaria.
   e. Calcular la media de los valores de la matriz que tienen un valor superior al umbral.

4. Dadas dos matrices de 4x4 caracteres simularemos el juego de '**hundir la flota'**.
   a. Ambas matrices se rellenarán con caracteres 'a' (que simularán agua).
   b. Por teclado se introducirán 5 posiciones por matriz que rellenaremos con 'b' (barco).
   c. Cada jugador realizará una jugada dando las coordenadas [i,j] de la matriz. Si la coordenada tiene 'b' pasará a 'a'. El primero que hunda toda la flota gana y termina el juego.
   d. Ampliar el programa para que se presente un mensaje de 'hundido', sólo si una posición marcada con 'b' como barco no tiene vecinos (en sentido vertical u horizontal) que tengan el valor 'b' (se ha hundido el barco completo).