EXERCICIS RESOLTS A CLASSE DE TEORIA

1. Realitzeu un programa que demane 10 nombres enters i en calcule la mitjana. Feu una funció/procediment per a emmagatzemar els valors en un vector i una altra per a calcular la mitjana.

2. (P1) Realitzeu un programa que demane 10 nombres enters i els guarde en un vector. Després demanarà un altre nombre i dirà si es troba en el vector (especificant en quina posició es troba) o si no s'hi troba. Per poder comprovar-ho, mostreu el contingut del vector per pantalla. Recordeu descompondre el programa en mòduls.

3. Realitzeu un programa que sume dos vectors d'un espai vectorial real de dimensió n (R^n). Suposeu que l'espai mai serà major que 50. Caldrà demanar la dimensió de l'espai abans de demanar les dades i comprovar la condició anterior. Recordeu descompondre el programa en mòduls.

4. (P5) Realitzeu un programa que demane a l'usuari N nombres enters i els inserisca ordenadament en un vector. N serà un valor que demanarem a l'inici i que mai haurà de ser major que 100. Seguidament, el programa permetrà la cerca d'elements dins del vector mitjançant la "cerca dicotòmica". Mentre l'usuari no decidisca eixir, el programa demanarà el nombre que es vulga buscar i n'informarà de la posició o dirà que no existeix. Per poder comprovar-ho, mostreu el vector per pantalla. Recordeu descompondre el programa en mòduls.

EXERCICIS PER A RESOLDRE

5. (P2) **(Alumnes.cpp)** Realitzeu un programa que llija les notes dels 10 alumnes d'una classe, en calcule la mitjana i determine quants alumnes superen aquest valor i quants hi estan per sota. L'estructura del programa serà la següent:

```
#include <iostream>
using namespace std;
#define ALUMNES 10

void llegir_notes ( float [ALUMNES] );
float mitjana ( float [ALUMNES]);
void sup_inf_mitjana ( float [ALUMNES], float, int & , int & );
int main ( void );
void llegir_notes ( float notes [ALUMNES] ))
{
   ...
}
float mitjana ( float notes[ALUMNES] )
{
   ...
}
void sup_inf_mitjana (float notes[ALUMNES], float mit, int &sup_mit, int &inf_mit)
{
   ...
}
int main ( void )
{
   float notes[ALUMNES], mit;
   int sup_mit, inf_mit;

   llegir_notes ( notes );
   mit = mitjana ( notes);
   sup_inf_mitjana ( notes, mit, sup_mit, inf_mit );

   cout << "La mitjana és " << med << ". Hi ha " << sup_med;
   cout << " alumnes amb una nota superior a la mitjana i " << inf_med;
```

```
   cout << " alumnes amb una nota inferior" << endl;

   system("pause");
   return 0;
}
```

6. (Op. 1) **(Carrera.cpp)** Realitzeu un programa que llij els temps en què 10 corredors han fet una carrera. El programa decidirà quins corredors tenen el primer, segon i últim lloc, així com quin és el temps mitjà de carrera. L'estructura del programa serà la següent:

```
#include <iostream>
using namespace std;
#define CORREDORS 10

void llegir_temps ( float [CORREDORS] );
float calcula_mitjana ( float [CORREDORS]);
void calcula_guanyadors ( float [CORREDORS], int & , int & );
int calcula_perdedor ( float [CORREDORS]);

void llegir_temps ( float dades[CORREDORS] )
{
   ...
}
float calcula_mitjana ( float dades[CORREDORS] )
{
   ...
}
void calcula_guanyadors (float dades[CORREDORS],int &primer_lloc,int &segon_lloc)
{
   ...
}
int calcula_perdedor( float dades[CORREDORS])
{

}
int main ( void )
{
   float temps[CORREDORS], mit;
   int primer, segon, últim;

   cout << "Dona'm els temps dels 10 corredors" << endl;
   llegir_temps ( temps );

   mit = calcula_mitjana ( temps ;
   calcula_guanyadors ( temps, primer, segon );
   ultim = calcula_perdedor( temps );


   cout << "La mitjana de temps és" << med << endl;
   cout << "El primer lloc és per al corredor número" << primer<< endl;
   cout << "El segon lloc és per al corredor número" << segon << endl;
   cout << "L'últim lloc és per al corredor número" << últim << endl;

   system("pause");
   return 0;
}
```

7. (P3) **(Repetits.cpp)** Dissenyeu un programa que demane el valor de 10 nombres enters diferents i els emmagatzeme en un vector. Si l'usuari volguera introduir un nombre repetit, el programa avisarà de la situació i demanarà un altre nombre fins que aquest siga diferent dels anteriors. En acabant, mostrarà els 10 nombres per pantalla.

8. (Op. 3) **(Capicua.cpp)** Realitzeu un programa que llija una quantitat determinada (que es demanarà a l'inici del programa) de nombres enters d'uns sola xifra, els emmagatzeme en un vector i comprove si el nombre format per tots els elements del vector és capicua o no. Empreu com a grandària màxima del vector 100 elements. Feu una funció "LlegirVector" i un altra "EsCapicua".

9. **(Punts.cpp)** Realitzeu un programa que demane les coordenades de espacials de dos punts (punts tridimensionals) i mostre un menú amb les opcions següents:

- Mostrar els punts per pantalla.
- Calcular la distància entre els dos punts.
- Calcular el vector director de la recta que passa pels dos punts.

Caldrà fer una funció/procediment per llegir un punt tridimensional, una per a visualitzar-lo, una per a calcular la distància i una altra per a calcular el vector director. Totes seran cridades des del programa principal.

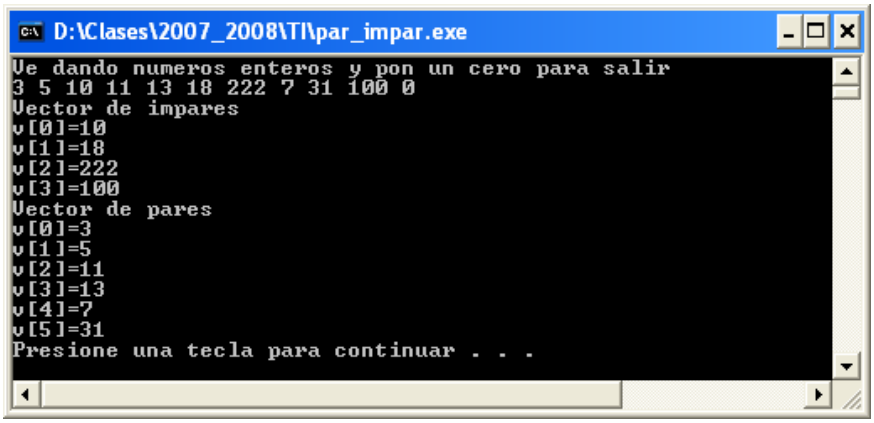
10. **(Vectors.cpp)** Realitzeu un programa que incloga una funció per a llegir vectors tridimensionals, una altra per mostrar-los per pantalla, una altra per a sumar, una altra per a fer la multiplicació escalar i una altra que faça la multiplicació vectorial. Feu una funció principal que mostre un menú i empre aquestes funcions per fer les diverses operacions.

Nota: Donats dos vectors {(a1, a2, a3) i (b1 ,b2, b3)}, el producte escalar i el producte vectorial es calcularan de la manera següent:
Producte escalar = Σ ai *bi i
Producte vectorial de dos vectors u(x,y,z) y v(x,y,z), on el resultat és: w(x,y,z)
w[x] = u[y] * v[z] - u[z] * v[y];
w[y] = u[z] * v[x] - u[x] * v[z];
w[z] = u[x] * v[y] - u[y] * v[x];

11. (P4) **(sumafila.cpp)** Dissenyeu un programa que llija els elements d'una matriu de 4 × 5 reals i genere un vector de grandària 4 en què cada element continga el sumatori dels elements de cada fila. El programa ha de mostrar la matriu original i el vector amb el format següent (evidentment, els valors correspondran amb aquells que haja introduït l'usuari):

```
        0      1      2      3      4       Suma
0 [ +27.33 +22.22 +10.00  +0.00 -22.22]  -> +37.33
1 [  +5.00  +0.00  -1.50  +2.50 +10.00]  -> +16.00
2 [  +3.45  +2.33  -4.56 +12.56 +12.01]  -> +25.79
3 [  +1.02  +2.22 +12.70 +34.00 +12.00]  -> +61.94
```

12. **(Parell_imparell.cpp)** Realitzeu un programa que demane nombres enters mentre no s'introduïsca el zero i emplene dos vectors, un amb els nombres parells i un altre, amb els imparells. En acabant, ha de mostrar per pantalla ambdós vectors amb el format que es mostra en la figura adjunta. Feu una funció/procediment per a "CarregarVectors" i una altra per a "MostrarVector".



13. **(VectorCaràcters.cpp)** Feu un programa que:

   a) Llija una seqüència d'un màxim de 1.000 caràcters i els emmagatzeme en un vector.

   b) Mostre per pantalla el vector llegit.

   c) Permeta eliminar totes les ocurrències d'un caràcter dins del vector.

   d) Mostre per pantalla el vector modificat.

Caldrà que definiu 5 funcions:

- Funció per a llegir vectors que rebrà com a paràmetre el nombre de caràcters que ha de llegir i tornarà per referència el vector de caràcters.
- Funció per a mostrar el vector que rebrà com a paràmetres el vector que cal mostrar i la dimensió d'aquest.
- Funció per a modificar un element del vector que rebrà com a paràmetres el vector que cal modificar, la posició i el caràcter nou.
- Funció per a buscar una lletra dins d'un vector que tornarà la posició en què es trobe la lletra o -1 si no hi és.
- Funció per a esborrar una lletra d'un vector que emprarà les funcions anteriors per a eliminar la primera ocurrència d'una lletra en el vector.

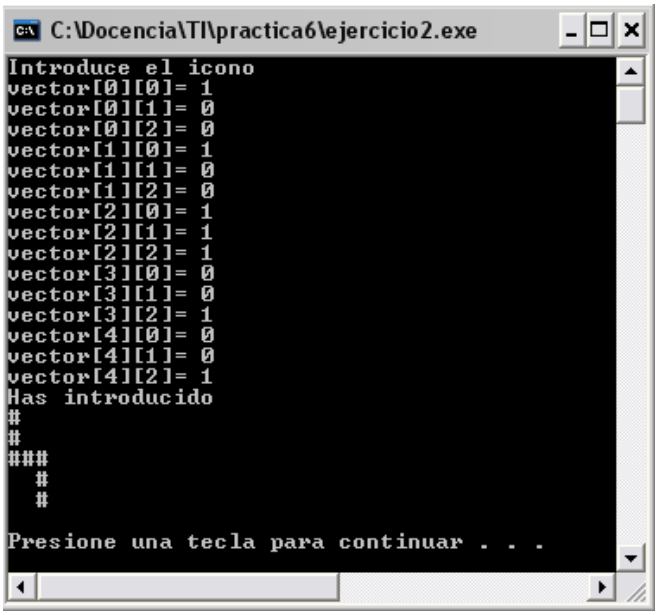
14. (Op. 2) **(icona.cpp)** Feu un programa que permeta llegir icones en blanc i negre. Les icones es representaran com a matrius (arrays bidimensionals). El programa permetrà introduir una icona i mostrar-la, substituint els 0 per espais en blanc i els 1, pel caràcter '#'. Cal que empreu una

funció/procediment que implemente cadascuna d'aquestes tasques: "LlegirMatriu", "VisualitzarMatriu" i "VeureIcona".

Les matrius seguiran el format:

```
#define HORITZ 5
#define VERTI 3

bool icono[HORITZ][VERTI];
```



15. (Op. 4) **(SumaMatrius.cpp)** Realitzeu un programa que demane dues matrius i les sume (sempre que siga possible) o indique si no es poden sumar. Feu el disseny descendent del problema (caldrà una funció/procediment per a llegir la matriu i una altra per a calcular la suma). El primer que farà el programa serà demanar la dimensió de la matriu i després se'n llegiran les dades.

Nota:

La suma de dues matrius A i B (sempre que A i B tinguen la mateixa dimensió) és una altra matriu C on es compleix que:

A←m x n (A és una matriu de m files i n columnes)

B←p x q (B és una matriu de p files i q columnes)

Si m=p i n=q, aleshores:

C←m x n (C serà la matriu suma que tindrà m files i n columnes) i s'obtindrà com: $C_{i,j}=A_{i,j}+B_{i,j}$

16. **(MultiplicaMatrius.cpp)** Realitzeu un programa que demane dues matrius (empreu una funció per a la lectura de matrius) i les multiplique (sempre que aquesta operació es puga realitzar).

Nota:

El producte de les matrius A i B és C:

A←m x p (A és una matriu de m files i p columnes)

B←p x n (B és una matriu de p files i n columnes)

C←m x n (C serà una matriu que tindrà m files i n columnes)

On $C_{i,j} = \sum_{k=1}^{p} A_{i,k} \cdot B_{k,j}$

El programa principal tindrà llavors l'estructura següent:

```
#include <iostream>

/* Prototips de funcions */
...

/* Definició de les funcions necessàries */
...

/* Funció principal */


int main ( void )
{
   int mat1[TAM][TAM], mat2[TAM][TAM], resultat[TAM][TAM];
….
/* es pot realitzar l'operació aleshores… */

   llegir_matriu ( mat1 , m , p);
   llegir_matriu ( mat2 , p , n);
   multiplicar ( mat1, mat2, resultat , m , p);
   mostrar_matriu ( resultat );
…
/* si no es pot, avisar del problema */
   return 0;
}
```

17. **(Transposada.cpp)** Calculeu la matriu transposada d'una altra matriu. Una matriu transposada s'obté intercanviant els valors de les files de la matriu original pels de les columnes.

Empreu un prototip similar a:

```
#define MAX_F  100
#define MAX_C  100
void transposada (double matriu[MAX_F][MAX_C], int f, int c);
```

La funció `transposada` rep la matriu original introduïda pel teclat, el nombre de files i de columnes (paràmetres f i c, respectivament) i calcula la transposada.

El programa principal mostrarà per pantalla, en format de matriu (files x columnes) la matriu original i la transposada.

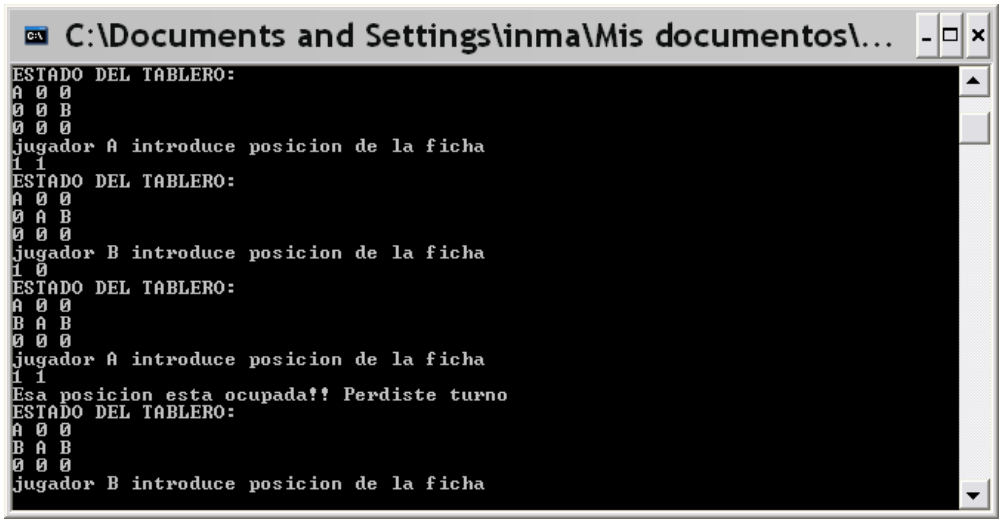
Nota: Cal fer també una funció per llegir matrius i una altra per a mostrar-les.

18. **(Tresenratlla.cpp)** Feu un programa que permeta jugar de manera senzilla al tres en ratlla de dos jugadors (A i B). El programa permetrà emmagatzemar l'estat del tauler (0 si ningú ha inserit cap fitxa en una posició, 1 pel jugador A i 2 pel jugador B). Realitzeu les funcions/procediments que:

a. Mostren l'estat actual del tauler.

b. Demanen la posició de la fitxa a introduir, alternativament, a cada jugador.

c. Comproven si el tauler està ple. Per senzillesa, aquesta versió del programa no comprovarà si s'han fet les tres en ratlla o no.



19. **(Tresenratlla2.cpp)** Completeu el programa del tres en ratlla perquè cada vegada que s'insereix una fitxa comprove si al tauler hi ha un tres en ratlla d'algun dels dos jugadors.

20. **(Palíndroma.cpp)** Realitzeu un programa en C/C++ que demane una frase i diga si és o no palíndroma (açò és, si es llig igual d'esquerra a dreta que de dreta a esquerra). Per exemple:

`Entrada → Refà l'afer Anna o Anna refà l'afer`

`Eixida → cert`

21. **(Aleatoris.cpp)** Implementeu un programa que declare un vector de 100 nombres enters. El programa mostrarà el menú següent:

1- Emplenar el vector amb nombres aleatoris.
2- Buscar un nombre amb cerca lineal.
3- Mostrar el vector.
4- Eixir.

Els nombres aleatoris es generaran entre 0 i 1.000. Cada nombre s'inserirà ordenat en el vector, que podrà contenir repetits. Per generar nombres aleatoris s'usarà la funció `rand()` de la llibreria `stdlib.h`, que genera un nombre aleatori entre 0 i RAND_MAX. Per a generar nombres entre 0 i el valor que vulguem, podem fer servir l'expressió següent:

`nom_aleatori = rand() * max/RAND_MAX` que genera nombres aleatoris entre 0 i max – 1.

Recordeu posar en el programa principal el valor inicial de la "llavor" a: **`srand(time(NULL))`**

22. **(Endevina.cpp)** Escriviu un programa en què un primer jugador introdueix una paraula, l'ordena aleatòriament i un segon jugador ha d'esbrinar quina era la paraula introduïda. El programa no tindrà en compte les majúscules i minúscules. Per tant, cal realitzar una funció per convertir totes les cadenes llegides a minúscules.