EXERCICIS PER A RESOLDRE

STRINGS

1. (P1) (Ocurrències.cpp) Escriviu un programa en C/C++ que, donada una cadena de caràcters, permeta substituir les ocurrències d'una altra subcadena. Exemple:

```
Cadena: Aquest problema és més interessant que el problema anterior.
Vella Subcadena: problema
Nova Subcadena: programa
Resultat: Aquest programa és més interessant que el programa anterior.
```

Feu una funció que, donada una cadena de caràcters i una paraula, substituirà cada aparició de la darrera per una nova cadena que també rebrà com a paràmetre. La funció també tornarà el nombre d'aparicions de 'vella cadena' en la frase. Un prototip adient podria ser:

```
int BuscarSubstituir(string &frase, string vellacadena, string novacadena);
```

2. (P2) (Caràcters.cpp) Escriviu un programa en C/C++ que, donada una cadena de caràcters, permeta obtenir el nombre total de caràcters i paraules (nota: els espais també són caràcters). Exemple:

```
Cadena: Setze jutges d'un jutjat mengen fetge d'un penjat
Eixida:
  Paraules: 8
  Caràcters: 49
```

3. (Penjat.cpp) Escriviu un programa per a jugar al penjat. El programa triarà una paraula (codificada directament en el programa) i mostrarà el següent:

Endevina la paraula: XXXXXX

Cada X representarà una lletra. Si l'usuari endevina la paraula, el programa mostrarà:

¡L'enhorabona! Has endevinat la paraula. Vols tornar a jugar? Sí/No

Cal que l'usuari introduïsca la resposta desitjada. Si l'usuari no encerta, el programa mostrarà la part del cos que ha perdut. Després de 7 errades, l'usuari serà penjat i per la pantalla es mostrarà la imatge ASCII següent:

```
 o
/|\
 |
/ \
```

4. (Palíndroma.cpp) Realitzeu un programa en C/C++ que demane una frase i indique si és o no un palíndrom (açò és, si es llig igual d'esquerra a dreta que de dreta a esquerra) mitjançant les funcions dels *strings*. Per exemple:

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

Eixida  → cert
```

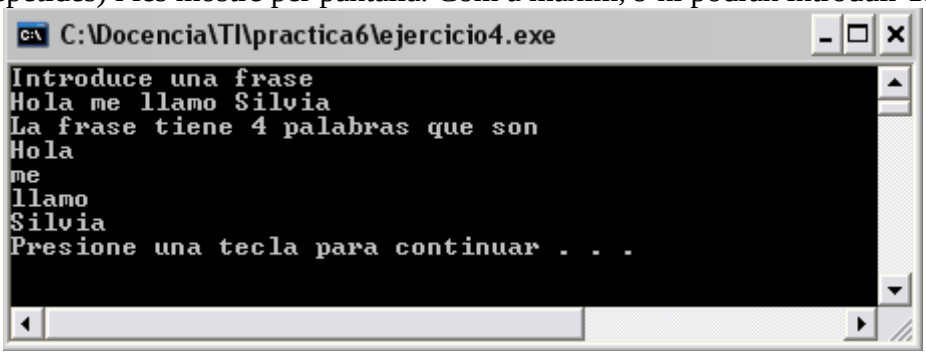
5. (Compta_par.cpp) Escriviu un programa que compte el nombre de paraules que hi ha en una sola línia d'entrada. Entenem per paraula una seqüència no buida de lletres majúscules o minúscules delimitada per l'inici de la línia, el final o qualsevol caràcter no alfabètic.

Per exemple, si l'usuari introdueix el darrer vers del poema "Els amants" del poeta burjassoter Vicent Andrés Estellés:

```
No hi havia a València dos amants com nosaltres, car d'amants com nosaltres en són parits ben pocs<retorno>
```

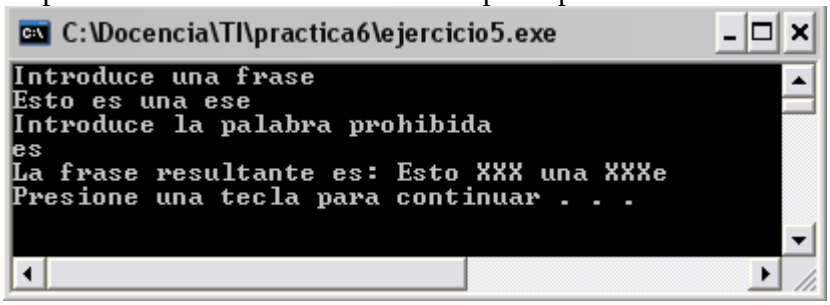
El programa comptarà 18 paraules.

6. (Separa.cpp) Escriviu un programa que emmagatzeme en un vector totes les paraules d'una frase (evitar repetides) i les mostre per pantalla. Com a màxim, s'hi podran introduir 1.000 paraules.

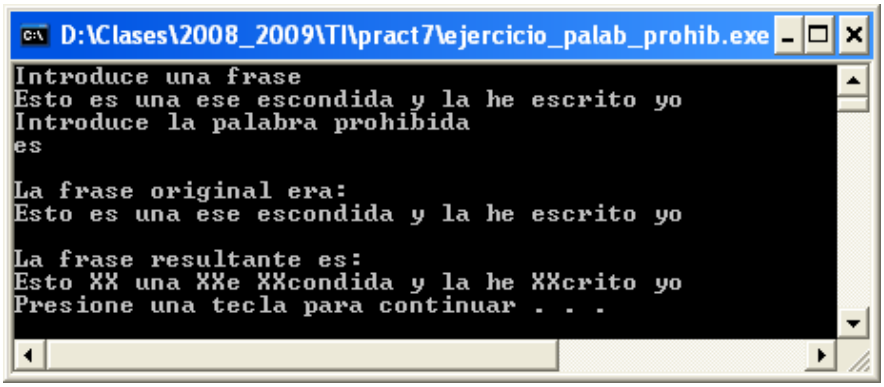


7. (Majúscules.cpp) Escriviu una funció que transforme una cadena de caràcters de minúscules a majúscules i useu-la des del programa principal.

8. (P3) (Par_prohib.cpp) Realitzeu un programa que cerque totes les ocurrències d'una paraula prohibida introduïda per l'usuari en una frase i les reemplace per la cadena "XXX".



9. (Op1) (Par_prohib2.cpp) Realitzeu un programa que cerque totes les ocurrències d'una paraula prohibida introduïda per l'usuari en una frase i la reemplace per la cadena "XXX" (on el nombre de "X" haurà de coincidir amb el nombre de caràcters que té la paraula prohibida).



10. **(Anagrama.cpp)** Diem que la paraula α és un anagrama de la paraula β si és possible obtenir α canviant d'ordre les lletres de β. Per exemple, ROMA és un anagrama d'AMOR. Realitzeu una funció que comprove si dues paraules són anagrames entre si. Les paraules poden contenir lletres en majúscules i minúscules. Feu una funció que convertisca majúscules ennúscules.

11. **(Vocals.cpp)** Escriviu un programa en C++ que calcule el nombre de vocals que hi ha en una frase introduïda pel teclat i informe de quantes vocals hi ha de cada tipus (quantitat de 'a', quantitat de 'e',...).

12. **(Quantes.cpp)** Escriviu un programa que, donada una frase, diga el nombre total de caràcters, de vocals, de consonants, d'espais en blanc així com el nombre de caràcters "d'un altre tipus".

13. **(Invertir.cpp)** Escriviu un programa que, donada una frase, torne la frase, però amb les paraules en ordre invers. Emprarem l'espai com el caràcter separador entre les paraules. Per tant, si la frase original és "Açò és un exemple, com ja us he dit.", la frase invertida serà "dit. he us ja com exemple, un és Açò".

14. **(Compte.cpp)** Feu un programa que llija una cadena des del teclat i obtinga el nombre de vocals i consonants que conté. Feu una funció *comptar_vocals* que reba una cadena i torne un enter amb la quantitat de vocals. Implementeu una funció similar que torne el nombre de consonants. El programa mostrarà un menú amb les opcions següents i s'executarà fins que l'usuari prema l'opció *Acabar*.

1. Llegir cadena.
2. Mostrar el nombre de vocals.
3. Mostrar el nombre de consonants.
4. Acabar.

15. **(Xifrat.cpp)** Volem desxifrar un missatge que està xifrat. Sabem que en el missatge s'ha reemplaçat la vocal 'a' per la 'e', la vocal 'e' per la 'i', la vocal 'i' per la 'o', la vocal 'o' per la 'u', i la vocal 'u' per la 'a'. Feu un programa que desfaça aquest tipus de xifrat, és a dir, que llija una cadena xifrada des del teclat i en mostre l'original. Per exemple, per a l'entrada *"hule, hu hes ecunsigaot"* mostrarà *"hola, ho has aconseguit"*. Feu una *funcio_desxifrar* que tinga com a entrada la cadena xifrada i torne la cadena desxifrada. El programa principal llegirà una cadena, cridarà a la funció i en mostrarà la cadena resultat.

16. (Op2) **(Soroll.cpp)** Escriviu un programa que lleve el soroll d'un senyal d'entrada. El senyal d'entrada serà una cadena amb lletres i nombres i l'eixida serà la mateixa cadena, però on els

nombres han estat eliminats. Per exemple, la cadena "Aç2ò 3és u9n se88nyal a0mb so1r2oll" quedarà com "Açò és un senyal amb soroll".

17. (Data.cpp) Escriviu un programa que convertisca una data en format "MMDDYYYY" al format "DD de mes de YYYY".

Per exemple, per a "12072006" tornarà "7 de desembre de 2006".

REGISTRES

18. **(P4)** **(complexos.cpp)** Realitzeu un programa en C/C++ que llegisca nombres complexos i faça operacions amb ells: suma, resta, multiplicació i divisió. Els nombres complexos es guardaràn en registres. Abans de mostrar el menú, el programa demanarà dos nombres complexos i després de fer l'operació mostrarà el resultat.

```
#include <iostream>
using namespace std;
struct complex
{
   int re, im;
};
/* Definicio de prototips */
void llegir     ( struct complex & );
void mostrar    ( struct complex );
void suma       ( struct complex, struct complex, struct complex & );
void multiplica ( struct complex, struct complex, struct complex & );
/* Implementacio de les funcions */
...
int main ( void )
{
   int opcio;
   struct complex a, b, res;

   cout << "Aquest programa fa operacions amb complexos." ;
   cout << "Dona'm el primer complex: " ;
   leer ( a );
   cout << "Dona'm el segon complex: " ;
   llegir( b );

   do
   {
      cout << " 1. Sumar \n" ;
      cout << " 2. Multiplicar \n" ;
      ...
      cout << " 0. Eixir \n" ;

      cin >> opcio ;
      switch ( opcio )
      {
         case 1: sumar ( a, b, res );
                 mostrar ( res );
                 break;
         ...
      }
   }
   while ( opcio != 0 );
   return 0;
}
```

19. **(fraccio.cpp)** Escriviu un programa que implemente l'estructura fracció amb dos camps, numerador i denominador. El programa inclourà una funció de suma amb el següent prototip:

```
suma_fraccions ( fraccio, fraccio );
```

20. **(punt.cpp)** Un *punt* és un element compost de tres coordenades *x*, *y* i *z*. Realitzeu un programa que demane dos punts, els guarde en dos variables de tipus *punt* i calcule i mostre, sense menús, la suma i la distància entre les dos variables.

21. (particula.cpp) Una *partícula* és un element compost de tres coordenades *x, y* i *z*, i una massa *m*. Realitzeu un programa que demane dos partícules, les guarde en dos variables de tipus *partícula* i calcule i mostre, sense menús, la distància entre les dos variables, la massa total i el centre de masses del sistema format per elles.

22. (triangles2D.cpp) Creeu un programa per a realitzar càlculs sobre triangles en un espai 2-D. El programa permetrà la introducció de les coordenades (x, y) pels tres vèrtex del triangle, els quals seran emmagatzemats en un registre. En acabant, calcularà l'àrea, el perímetre i el punt intermedi del triangle.

Àrea = base * altura / 2;
Àrea = ½ * ((x3.x – x1.x) * (x1.y- x2.y) + (x3.y - x1.y)*(x2.x - x1.x))
Perímetre = distancia(x1,x2) + distancia (x2,x3) + distancia(x3,x1)
Punt_intermedi = ((x1.x + x2.x + x3.x) / 3, (x1.y + x2.y + x3.y) / 3)

23. (Op3) **(mecanic.cpp)** Creeu una estructura de dades que allotge informació sobre components mecànics de l'automòbil. El programa serà capaç d'emmagatzemar fins a un màxim de 10 peces.

Cada fitxa d'un component contindrà la informació següent:

a. Nom de la peça
b. Unitats disponibles
c. Preu de la peça

Feu un programa que, mitjançant un menú, faça les següents accions:

- Introduir una nova peça, tot i comprovant que encara hi ha espai disponible.
- Eliminar una peça i moure una posició totes les posteriors. Caldrà actualitzar un comptador per a saber el nombre de peces emmagatzemades.
- Cercar per nom i llistar per pantalla les característiques d'una peça concreta.

Cada opció del menú serà realitzada com un subprograma diferent que rebrà els paràmetres que considereu escaients.

24. (Op4) **(imatges.cpp)** Feu un programa que dibuixe imatges binàries per pantalla. Una imatge binària és una matriu de n x m valors, els quals poden ser 1 o 0. Si el valor és un 1, el pixel de la imatge és de color negre, si és un 0, el pixel és de color blanc..

Creeu un tipus de registre per a treballar amb imatges binàries de, com a màxim, 80 x 24 pixels. A banda dels pixels, caldrà emmagatzemar també la grandària de la imatge, que podrà ser menor a 80 x 24 i també el nom de la imatge.

Feu un procediment que reba un registre d'imatge i guarde en ell una imatge d'una recta de –45 graus (açò és equivalent a posar a '1' la diagonal principal d'una imatge quadrada amb la mateixa dimensió de files que de columnes). A més a més, demanarà el nom i la resta de paràmetres. Proveu amb altres dibuixos si us ve de gust.

Feu un procediment que reba un registre d'imatge i dibuixe per consola de text la imatge. A l'hora d'imprimir, si hi ha un 1 en la matriu escriviu el caràcter '*' i si hi ha un 0, escriviu el caràcter ' '. El procediment anterior es podrà implementar amb un bucle "similar" al següent:

```
for(i=0; i<MAXFILES; i++)
{
```

```
        for(j=0; j<MAXCOLUMNES; j++)
             if(<el valor és 1>)
                 cout << '*';
             else
                cout << ' ';
        cout << endl;
    }
```

25. **(P5)** **(ciutats.cpp)** Construïu un vector amb informació relativa a un conjunt de ciutats. Cadascun dels elements del vector contindrà: nom de la ciutat, nombre d'habitants i codi de la província a què pertany la ciutat (valor enter).

Definiu les estructures de dades necessàries i les funcions per fer les tasques següents:

- Emplenar el vector de dades llegides des del teclat (el nombre màxim de ciutats és 20).
- Imprimir la suma total d'habitants de totes les ciutats que pertanyen a una província determinada (identificada pel seu codi).
- Imprimir la informació de la ciutat amb el major nombre d'habitants. Si hi ha més d'una ciutat que coincideixen en aquest valor màxim, el programa imprimirà la informació de la ciutat que tinga el menor índex del vector.

```
include <iostream.h>
using namespace std;

#define NUM 20

struct ciutat
{
//definir camps
 };

/* Definicio de prototips */
void llegir( ciutat &c );
void emplena_vector (ciutat v[NUM], int &tam );
int calcular_habit (ciutat v[NUM], int tam, int cod);
ciudad maxim_habitants ((ciutat v[NUM], int tam);
int main ( void );

/* Implementacio de les funcions */
…..
```

26. **(futbol.cpp)** Feu un programa per gestionar la informació de la taula de classificacions d'un campionat d'equips de futbol. Feu servir un registre que incloga el nom de l'equip, els punts obtinguts, els partits guanyats, empatats i perduts, els gols a favor i els gols en contra. Empreu un vector per emmagatzemar la informació dels 10 equips del campionat. Genereu dades aleatòries per a emplenar els valors de cada equip i fer proves. Després de generar la informació, el programa indicarà quins són el primer i l'últim equip de la classificació així com tota la seua informació. També podrà mostrar la informació completa sobre la classificació.

27. (calendari.cpp) Creeu un programa que implemente un calendari del 2011. Per a cada dia, el calendari guardarà el número de dia, el mes, el dia de la setmana, si és festiu o no i el nom de la festivitat, si escau. Empleneu el calendari amb la primera setmana de gener de 2011 i mostreu-la per pantalla. Implementeu una funció que torne el dia de la setmana que correspon amb una data determinada.

28. (alumnes.cpp) Realitzeu un programa que llegisca els noms i les notes dels 10 alumnes d'una classe, calcule la mitjana i determine quants superen i quants estan per sota de la nota mitjana. L'estructura del programa ja dividit en mòduls és la següent:

```
#include <iostream>
#include <string>
using namespace std;

#define ALUMNES 10

struct alumne
{
       string nom;
       float nota;
};

/* Definicio de prototips */

void llegir_notes( alumne [ALUMNES] );
float nota_mitjana (alumne [ALUMNES]);
void sup_inf_mitjana (alumne [ALUMNES], float, float &, float &, float & );
int main ( void );

/* Implementacio de les funcions */

void llegir_notes (alumne dades[ALUMNES] ))
{
   ...
}

float nota_mitjana (alumne dades[ALUMNES])
{
   ...
}
void sup_inf_mitjana (alumne dades[ALUMNES], float m, float &sup_m, float &inf_m )
{
   ...
}
int main ( void )
{
   alumne notes[ALUMNES];
   float mit, sup_mit, inf_mit;

   llegir_notes ( notes );
```

```
   mit = nota_mitjana ( notes);
   sup_inf_mitjana ( notes, mit, sup_mit, inf_mit );

   cout << "mitjana " << mit << " Sup " << sup_mit << " Inf " << inf_mit << endl;

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