**Universitat de Lleida. Escola Universitària Politècnica**
**Examen d'Estructures de dades i algorismes. 11 de febrer de 1997**

**Problema 1** (2)

En cadascuna de les situacions que descrivim tot seguit cal implementar una taula per a obtenir accés directe per una determinada clau. Digues en cada cas quina és la clau per la qual s'ha de fer accés directe i quina seria una bona implementació de les taules per aquella situació (a voltes més d'una solució és possible, però només en demano una):

1. El sistema informàtic de gestió d'un hospital necessita saber amb tota rapidesa les dades del pacient que es troba en una determinada habitació.
2. El sistema d'ajut d'un famós processador de textos està estructurat de forma que quan l'usuari entri les primeres lletres d'alguna comanda del processador, el sistema retorni ràpidament una llista de totes les comandes que comencen per aquelles lletres.
3. El sistema de cerques informatitzades de la biblioteca d'una important universitat està estructurat de forma que quan l'usuari entri la (les) primera (es) paraula (es) del títol d'un llibre, el sistema retorni una llista amb tots els llibres que comencin per aquella (es) paraula (es).
4. Una cadena de supermercats ha implantat un sistema informàtic que permet la lectura del codi de barres de cada producte que es troba en el supermercat. En el mateix moment de la lectura, cal registrar a la nota del client el nom i preu del producte i, també cal restar una unitat de les existències d'aquell producte a la base de dades del supermercat.
5. Els mossos d'esquadra, conscients de que la bona salut del pacte els permetrà assumir les competències de tràfic en els propers mesos, han demanat a una empresa de software que els dissenyi un sistema que els permeti saber immediatament, a la vista d'una matrícula, totes les dades del cotxe i del seu propietari.

**Problema 2** (2.5)

Ens plantegem el problema d'escriure un arbre en un fitxer seqüencial de forma que posteriorment pugui ser recuperat.

Com ho podem fer

1. Si l'arbre és un arbre binari.
2. Si l'arbre és un arbre binari de cerca i el volem recuperar exactament amb la mateixa forma que té abans de ser escrit al fitxer.
3. Si l'arbre és un arbre binari de cerca i el volem recuperar de forma que quedi el més equilibrat possible (independentment de si abans d'escriure'l al fitxer era o no equilibrat).

**Problema 3** (2.5)

Considerem el següent codi $C++$:

```
#include <iostream.h>

class o_c{

  char nom[20];
  long  preu;

 public:

  o_c(..){..}

  void llista_caract(){
   cout<<"nom: "<<nom<<"\n";
   cout<<"preu: "<<preu<<"\n";
  }
 //...
};

class moble :public o_c{

  char material[20];

 public:

  moble(..){..}

  void llista_caract(){
   ...
  }
 //...
};
```

1. Escriu el codi de les operacions constructores de `o_c` i de `moble`.
2. Escriu el codi de l'operació `llista_caract` de `moble` de forma que llisti el valor de tots els atributs d'un objecte de classe *moble*.
3. Quina és la sortida d'aquest programa?

```
void f(o_c& x)
{
  x.llista_caract();
```

```
}


void f2(o_c x)
{
  x.llista_caract();
}

void main()
{
  o_c x1("detergent", 200);
  moble x2("taula",10000,"fusta");

 x1.llista_caract();
 x2.llista_caract();
 f(x1);
 f(x2);
 f2(x1);
 f2(x2);
}
```

4. Quina seria la sortida del mateix programa si l'operació `llista_caract` de la classe `o_c` hagués estat declarada:

   `virtual void llista_caract(){...}` ?

5. Com sobrecarregaries l'operador de sortida estàndar `<<` per a que si canviem al programa de l'apartat 3 totes les crides a l'operació `llista_caract` per crides a l'operador `<<` continui obtenint el mateix resultat?

**Problema 4** (3)

Volem implementar un diccionari mitjançant una *taula de hash* per tal d'aconseguir la major velocitat en la cerca d'una determinada paraula.

Proposem la funció de dispersió següent:

Sigui $m$ un mot i $m_1, m_2, .., m_n$ els caràcters de què està composat.

Sigui ordre($\emptyset$)=0, ordre('a')=1, ordre('b')=2,..., ordre('z')=26

$h(m)$=ordre($m_1$)*1000 + ordre($m_2$)*100 + ordre($m_3$)*10 + ordre($m_4$)

Si un mot tingués una longitud més petita de 4, el completaríem per l'esquerra amb $\emptyset$ fins arribar a la longitud 4 (ex: $'a' \longrightarrow' \emptyset\emptyset\emptyset a'$

1. Comenta breument però raonada, el rendiment de cadascuna de les situacions que descrivim tot seguit suposant que utilitzem com a funció de dispersió la que hem presentat.

   (a) Volem col.locar al diccionari 20000 cadenes aleatòries de caràcters, utilitzant una estratègia d'adreçament obert amb redispersió lineal.

   (b) Volem col.locar al diccionari 26000 cadenes aleatòries de caràcters, utilitzant una estratègia d'adreçament obert amb redispersió lineal.

   (c) Volem col.locar al diccionari 20000 paraules catalanes utilitzant una estratègia d'adreçament obert amb redispersió quadràtica.

   (d) Volem col.locar al diccionari 20000 paraules provinents d'un llibre escrit en català utilitzant una estratègia d'encadenaments en memòria dinàmica.

2. Finalment, suposem que volem inserir al diccionari 30000 paraules provinents d'un llibre escrit en català i que optem per un adreçament obert:

   (a) Dimensiona el vector que necessitarem.

   (b) Proposa una funció de dispersió i una família de funcions de redispersió de tal manera que s'evitin el major nombre de problemes possibles. Comenta les característiques de les funcions que proposes (ventatges i inconvenients).

**Full de resultats**

**Problema 1**

1. **Clau:** **Implementació:**
2. **Clau:** **Implementació:**
3. **Clau:** **Implementació:**
4. **Clau:** **Implementació:**
5. **Clau:** **Implementació:**

**Problema 2**

1. Arbre binari:

2. Arbre binari de cerca recuperat idènticament:

3. Arbre binari de cerca recuperat equilibrat:

**Problema 4**

**1.a**

**1.b**

**1.c**

**1.d**

**2.a**

**2.b**