



Transparències de suport de la Pràctica 3

Estructura de Dades

Grau en Enginyeria Informàtica
Facultat de Matemàtiques i Informàtica,
Universitat de Barcelona



Laboratori 8. Pràctica 3

(inclou les transparències del Laboratori 7)

Estructura de Dades

Grau en Enginyeria Informàtica
Facultat de Matemàtiques i Informàtica,
Universitat de Barcelona

Descripció del problema

Fer una aplicació per a la gestió de transaccions d'un corredor de borsa (broker)

El corredor (o broker) ens ha facilitat un fitxer on hi té emmagatzemades totes les transaccions (una per línia) dels seus clients l'any 2020.

Per cada transacció, hi consten:

1. L'instant de temps, data i hora, de la transacció (*time* en format "YYYY-MM-DD HH:mm") guardat com string,
2. L'identificador intern dels seu client (*client_id*) guardat com integer, i
3. La quantia de la transacció (*amount*) en euros guardada com a float, que serà negativa si es tracta d'una compra d'accions per part del seu client o positiva en cas de tractar-se d'una venda.

Descripció del problema

El fitxer de transaccions inclou una capçalera:

```
time,client_id,amount
2020-01-01 09:23,11712,107.78
2020-01-01 09:29,13708,5121.25
2020-01-01 12:48,24190,1575.12
2020-01-01 13:47,25088,-149.52
2020-01-01 13:56,5395,2268.61
2020-01-01 14:23,6094,-1068.34
2020-01-01 14:45,17893,262.73
2020-01-01 14:52,23209,-5823.66
2020-01-01 15:16,24757,-1489.3
2020-01-01 15:24,6049,-1506.71
2020-01-02 09:27,2744,-11161.46
2020-01-02 09:36,27769,-227.6
2020-01-02 09:42,14245,2881.39
2020-01-02 10:07,3736,-487.7
2020-01-02 10:38,25209,138.51
```

Contingut de la Pràctica 3

Exercicis:

- 4 exercicis a desenvolupar per resoldre el problema plantejat
- 1 exercici d'avaluació de les estructures

Tindreu lliuraments parciaus al campus virtual on deixar la feina feta fins aquell moment.

- La darrera setmana s'han de lliurar en el ZIP TOTS els exercicis

Lliurament: dia 16 de maig de 2021

Exercici 1

- **Objectiu:**
 - Definir i implementar el **TAD BinarySearchTree**
- **Passos:**
 - Definir i implementar el TAD **BinaryTreeNode**
 - Definir i implementar el TAD **BinarySearchTree**
 - Fer un **main.cpp** que usi el TAD BSTree i faci el cas de prova definit a l'enunciat.

És opcional implementar el TAD amb templates. Qui ho faci tindrà puntuació extra a la nota final de la pràctica.

Exercici 1: Especificació BinaryTreeNode

Operació	Definició
constructor	Construeix un nou node de l'arbre binari, passant com a paràmetre una clau
constructor còpia	Construeix una còpia del node partir del node original rebut per paràmetre
destructor	Destruïx els nodes fills. Atenció a la gestió de memòria dinàmica
getKey	Retorna la clau del node (atribut key)
getValues	Retorna el vector de valors (atribut values)
isRoot	Retorna cert si el node és el root de l'arbre binari de cerca, fals altrament
hasLeft	Retorna cert si el node té un fill esquerre, fals altrament
hasRight	Retorna cert si el node té un fill dret, fals altrament
isLeaf	Retorna cert si el node és una fulla de l'arbre binari de cerca, fals altrament
addValue	Afegeix un valor a la llista de valors.
depth	Retorna la profunditat del node en l'arbre binari de cerca. Convindrem que el root té sempre profunditat 0. Aquesta funció s'ha d'implementar de forma RECURSIVA
height	Retorna l'alçada del node en l'arbre de cerca binari. Convindrem que les fulles sempre tenen alçada 1. Aquesta funció s'ha d'implementar de forma RECURSIVA
operator==	(Sobrecarrega de l'operador d'igualtat) Retorna cert si dos nodes són iguals: tenen la mateixa clau i els mateixos valors

Exercici 1: Especificació BinarySearchTree

Operació	Definició
constructor	Construeix l'arbre buit.
constructor còpia	Construeix una còpia de l'arbre partir del node original rebut per paràmetre
destructor	Destruïx els nodes de l'arbre binari, començant pel root.
isEmpty	Atenció a la gestió de memòria dinàmica
size	Retorna cert si l'arbre binari està buit, fals en cas contrari
height	Retorna el nombre de nodes que hi ha l'arbre binari. Aquesta funció recorre els nodes per calcular quants nodes té l'arbre binari.
add	Retorna un enter amb l'alçada de l'arbre binari de cerca, és a dir, l'alçada del root. Aquesta funció s'ha d'implementar de forma RECURSIVA
has	Afegeix un nou node a l'arbre binari de cerca Retorna cert si l'arbre binari de cerca té l'element indexat amb una certa clau, fals en cas contrari. Aquesta funció s'ha d'implementar de forma RECURSIVA
valuesOf	Retorna el vector de valors (atribut values) d'un node de l'arbre amb una certa key passada per paràmetre
showPreOrder	Mostra per consola les claus i el contingut de les claus dels nodes seguint un recorregut en preordre
showInOrder	Mostra per consola les claus i el contingut les claus dels nodes seguint un recorregut en inordre
showPostOrder	Mostra per consola les claus i el contingut les claus dels nodes seguint un recorregut en postordre
equals	Retorna cert si l'arbre binari intern és igual a l'arbre binari que es passa per paràmetre, fals en cas contrari. Dos arbres són iguals si tots els seus nodes ho són
getLeafs	Retorna un vector amb tots els nodes fulla de l'arbre.
find (protected)	Troba un element (indexat per la clau) de l'arbre binari de cerca i retorna el node si el troba, en cas contrari retorna nullptr. Aquesta funció s'ha d'implementar de forma ITERATIVA.

Exercici 1. Pasos

1. Implementeu el TAD `BinaryTreeNode`
2. Implementeu el TAD `BinarySearchTree`
3. Feu el cas de prova per testejar l'arbre binari de cerca

Pas 1. TAD BinaryTreeNode

```
template <class K, class V>
class BinaryTreeNode {
public:
    BinaryTreeNode(const K& key);
    BinaryTreeNode(const BinaryTreeNode<K,V>& orig);
    virtual ~BinaryTreeNode();

    // ELS METODES DEFINITS A L'ENUNCIAT

private:
    K key;
    vector<V> values;
    // Afegiu-hi aquí els atributs que manquen
};
```

ÉS OPCIONAL FER _HO AMB TEMPLATES!

Pas 2. TAD BinarySearchTree

```
template <class K, class V>
class BinarySearchTree {
public:
    BinarySearchTree();
    BinarySearchTree(const BinarySearchTree <K, V>& orig);
    virtual ~BinarySearchTree();
    // ELS METODES DEFINITS A L'ENUNCIAT
protected:
    BinaryTreeNode<K, V>* p_root;
    BinaryTreeNode<K, V>* find(const K& key) const;
};
```

ÉS OPCIONAL FER HO AMB TEMPLATES!

Pas 3. Cas de prova

Crear dos arbres (anomenats tree1) que emmagatzemin la clau entera i valors enters.

Inserir a l'arbre (primer) les claus d'un array anomenat testKeys que contindrà :

```
int testKeys [] = {2, 0, 8, 45, 76, 5, 3, 40 };  
int testValues[] = {5, 5, 1, 88, 99, 12, 9, 11};
```

Mostrar en preordre l'arbre (tree1) per pantalla

Mostrar en postordre l'arbre (tree1) per pantalla

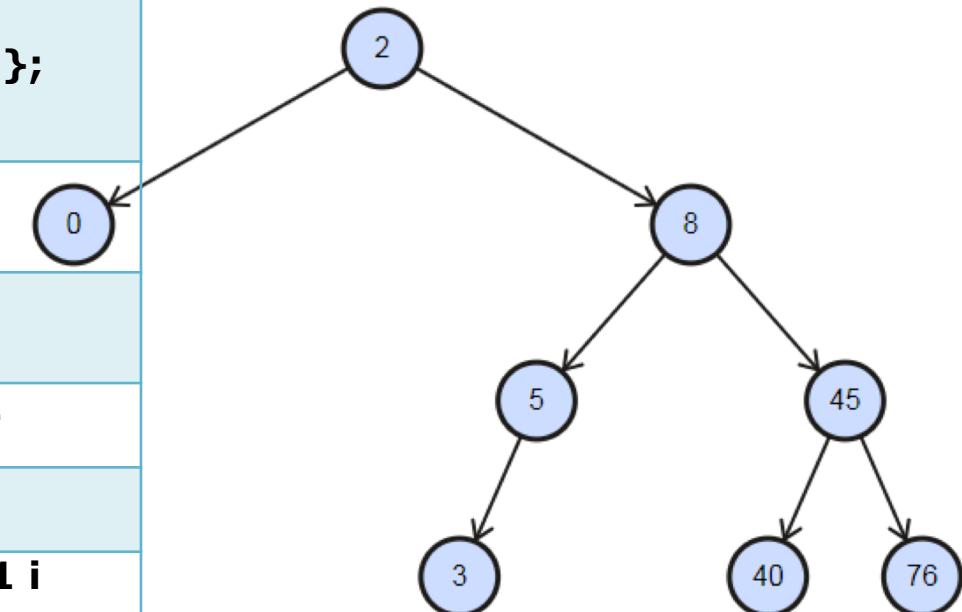
Fer una còpia de l'arbre tree1 sobre tree2

Cridar a la funció tree1.equals(tree2)

Afegir a l'arbre tree1 l'element amb clau 1 i valor 10

Cridar a la funció tree2.equals(tree1)

Eliminar l'arbre



Exercici 2

- **Objectiu:**
 - Fer el gestor de transaccions pel corredor de borsa
- **Passos:**
 - Definir i implementar el TAD **Transactions**
 - Implementar el TAD **TransactionManager**
 - Fer un **main.cpp** que usi el TAD TransactionManager i que resolgui el problema plantejat

S'han de controlar les excepcions

Exercici 2

- **Fitxers al campus virtual:**
 - *transactions-cas_de_prova.shuffled.txt*: fitxer cas de prova per testejar l'aplicació amb transaccions desordenades
 - *transacctions-cas_de_prova.txt*: fitxer cas de prova per testejar transaccions
 - *transactions-large.shuffled.txt*: fitxer gran desordenat
 - *transactions-large.txt*: fitxer gran ordenat
 - *transactions-small.shuffled.txt*: fitxer petit desordenat
 - *transactions-small.txt*: fitxer petit
 - *queries.txt*: fitxer que conté transaccions per cercar a l'arbre binari de cerca
- Els fitxers s'inclouran al projecte a la part de **Resources**

Exercici 2: Especificació TAD Transaction

Operació Definició

Constructor	i Construeix una transacció amb totes les seves dades: instant de temps de la transacció (string en format YYYY-MM-DD HH:mm), ID del client (int), i quantitat de la transacció (float).
funcions consultores	Una o més funcions per consultar cada una de les dades associades a la transacció
funcions modificadores	Una o més funcions per modificar cada una de les dades associades a la transacció
print	Imprimirà la transacció en format: (atribut_1, ..., atribut_N)

Herència

TransactionManager ha d'heretar de BinarySearchTree

TransactionManager.h

(BinarySearchTree implementat amb templates)

```
class TransactionManager : protected BinarySearchTree<string, Transaction> {  
    ...  
}
```

TransactionManager.h

(BinarySearchTree implementat sense templates)

```
class TransactionManager : protected BinarySearchTree {  
    ...  
}
```

Exercici 2: Main

1. Oferirà a l'usuari especificar la ruta del fitxer que es vulgui carregar en el Gestor de Transaccions. Us proporcionarem, mitjançant el Campus Virtual, diversos fitxers trading-* .txt que haureu d'afegir al directori arrel del vostre projecte i com a "Resource Files" del vostre projecte NetBeans.
2. Mostrarà totes les transaccions ordenades temporalment, demanant primer quantes se'n desitgen mostrar de cop i, després d'haver-les mostrat, demanarà quantes més se'n volen mostrar. Si l'usuari no desitja seguir, entrarà per teclat un 0.
3. Mostrarà totes les transaccions en ordre temporal invers, de la mateixa manera que es fa a l'opció 2.
4. Mostrarà les transaccions del primer instant de temps del fitxer de transaccions carregat.
5. Mostrarà les transaccions del darrer instant de temps del fitxer de transaccions carregat.
6. Mostrarà la comissió recaptada amb totes les transaccions.
7. Mostrarà la comissió recaptada a partir d'una data específica.
8. Mostrarà la comissió recaptada entre dues dates. La interacció amb l'usuari serà semblant a la de l'opció 5, amb la diferència que se li demanaran les dues dates que delimiten l'interval temporal.
9. Mostrarà el balanç de totes les transaccions efectuades en les dates llistades en un fitxer queries.txt. Aquest fitxer no contindrà transaccions, sinó només dates que serviran de clau per anar a buscar les transaccions ja carregades al **TransactionManager**. Igual que en l'opció 1, us demanarem mesurar el temps que trigui a processar-se la petició.
10. Sortir

Exercici 3

- **Objectiu:**
 - Definir i implementar un arbre binari balancejat
- **Definir i implementar el TAD AVLTree**
 - AVLTree hereta del TAD BinarySearchTree
 - És opcional implementar el TAD amb templates. Qui ho faci tindrà puntuació extra a la nota final de la pràctica.
- Fer un **main.cpp** que usi el TAD AVLTree i faci el cas de prova definit a l'exercici 1

Exercici 4

- **Objectiu:**
 - Fer el gestor de transaccions pel corredor de borsa
- **Passos:**
 - Implementar el TAD **TransactionManagerAVL**
 - Fer un **main.cpp** que usi el TAD TransactionManager i que resolgui el problema plantejat

S'han de controlar les excepcions

Exercici 5

- **Objectiu:**
 - Fer una avaluació comparativa entre els dos tipus d'arbres implementats a la pràctica
- Calculeu el temps de cerca i inserció als dos arbres (BSTree i AVLTree)
 - Feu les proves en la mateixa màquina i en condicions similars
 - Disposeu de dos fitxers per fer les proves *shortText.txt* i *longText.txt*,
 - *Dictionary.txt* per fer les cerques
- Poseu la taula comparativa i comenteu els resultats al main de l'exercici 4

Exercici 5

- **Objectiu:**
 - Fer una avaluació comparativa entre els dos tipus d'arbres implementats a la pràctica
- Calculeu el temps de cerca i inserció als dos arbres (BinarySearchTree i AVLTree)
 - Feu les proves en la mateixa màquina i en condicions similars
 - Disposeu de diferents fitxers per fer les proves
 - *queries.txt* per fer les cerques
- Poseu la taula comparativa i comenteu els resultats al main de l'exercici 4

Planificació (del 8 d'abril al 16 de Maig)

- **Setmana 1** (*Classe de Laboratori 7*)
 - Implementació de les classes **BinaryTreeNode**, **Transactions** i comentar el codi
 - Fer una funció que llegeixi el fitxer i el guardi cada línia a un objecte Transactions
- **Setmana 2** (*Classe de Laboratori 8*)
 - Implementació del **BinarySearchTree**, el main, i comentar el codi
- **Setmana 3** (*Classe de Laboratori 9*)
 - Implementació de la classe **TransactionsManager** i comentar el codi
 - Implementació del main de l'exercici 2, la part de l'exercici 5 sobre arbres binaris i comentar el codi
- **Setmana 4** (*Classe de Laboratori 10*)
 - Implementació de l'exercici 3 i comentar el codi
- **Setmana 5** (*Classe de Laboratori 11*)
 - Implementació de l'exercici 4, exercici 5 i comentar el codi
- **Setmana 6** (*Classe de Laboratori 12*)
 - Dubtes

Transparències de suport de la Pràctica 3

Estructura de Dades

Grau en Enginyeria Informàtica
Facultat de Matemàtiques i Informàtica,
Universitat de Barcelona