CSE214 Data Structures

File System Tree

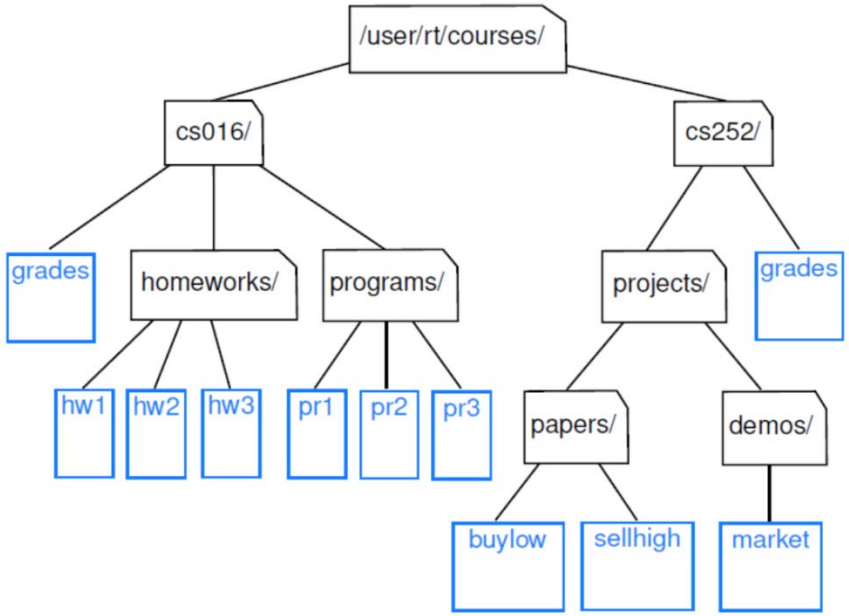
YoungMin Kwon

Binary Tree for File Systems

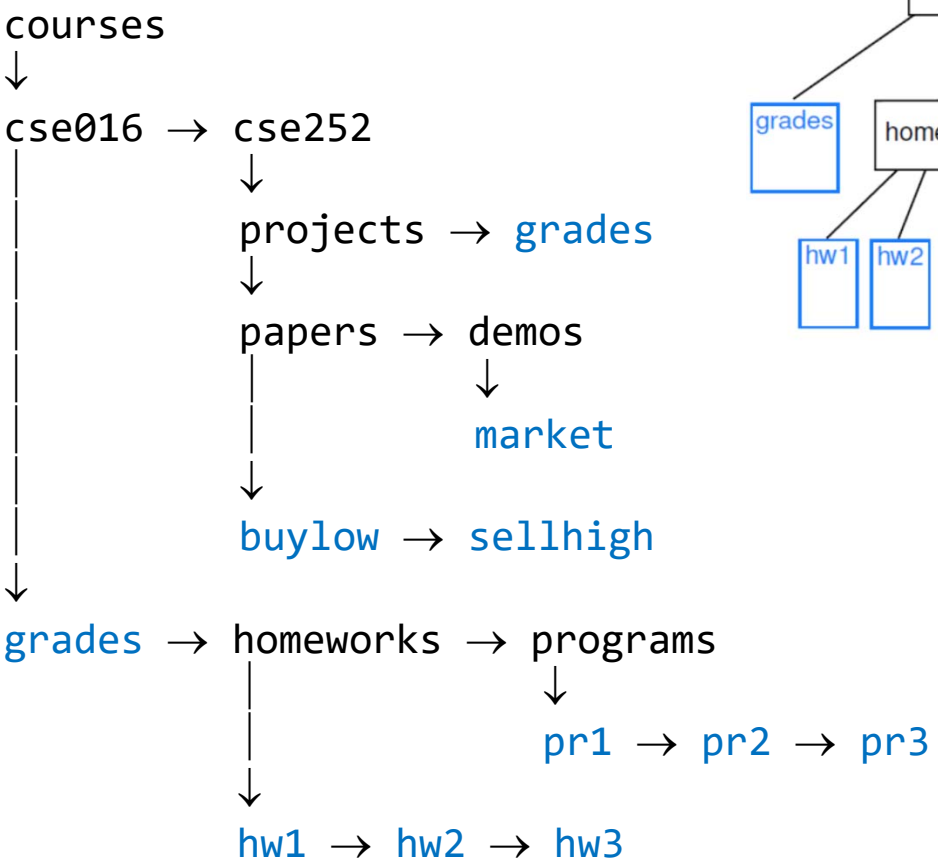
- Binary trees can represent directory structure
- Given a tree for a directory
  - Entries in the directory are added along the left links
  - If an entry is a directory, the tree for the entry is added to the right link of the entry

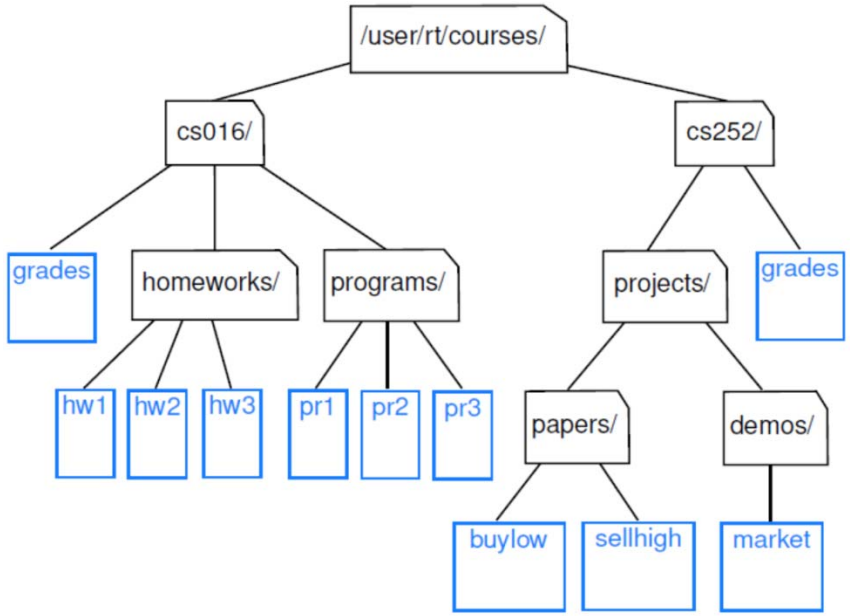
Binary Tree for File Systems

- Can we represent this file system structure by a binary tree?



Binary Tree for File Systems





```java
import java.io.File;

public class FileSystemTree {
    LinkedBinaryTree<String> fileSys;

    public static LinkedBinaryTree<String> build(String root) {
        return buildTree(new File(root));
    }

    public static void main(String[] args) {
        LinkedBinaryTree<String> tree = build(
                                            "C:\\Users\\youngmin.kwon\\tmp");
        //Initially check if the tree is built correctly
        for(String name : tree)
            System.out.println(name);

        //Print the structure with indentations
        printTree(tree, tree.root(), 0);
    }
```

```
protected static LinkedBinaryTree<String> buildTree(File root) {
    //TODO: - create an entry of LinkedBinaryTree<String> instance;
    //      - add root.getName() to the entry

    if(root.isDirectory()) {
        LinkedBinaryTree<String> folder = null;
        Position<String> pos = null;
        for(String childName: root.list()) {
            //TODO: - create child, a File instance, using root and childName;
            //      - create a childEntry, a LinkedBinaryTree instance, by
            //        calling buildTree;
            if(folder == null) {
                //TODO: - copy childEntry to folder
                //      - set pos to the root of folder
            }
            else {
                //TODO: - attach childEntry to the right link of pos
                //      - move pos to the right child of pos
            }
        }
        //TODO: - attach folder to the left link of entry
    }
    return entry;
}
```

```
public static void printTree(BinaryTree<String> tree,
                             Position<String> pos, int depth) {
    for(int i = 0; i < depth; i++)
        System.out.print("    ");

    //TODO: - print the name at pos;
    //      - if pos is a directory, visit all entries in the directory
    //        with an increased depth;
    //      - if more entries are in the current directory visit the
    //        remaining entries;
}
```

Sample Output

```
CSE114_LabTest_2_WoohyungLee.zip
CSE114_S17_LabTest2_SeungHwaLee.zip
CSE114_S17_Labtest_02_YeonjinOh.zip
CSE114_S17_Labtest_2_KyoungaWoo.zip
CSE114_S17_Labtest_2_SangUkLee.zip
CSE114_S17_Labtest_2_SunYoungPark.zip
CSE114_S17_Labtest_2_ZionLee.zip
Untitled.zip
test2
CSE114.S17_Labtest_2_HyungTaekKwon.zip
CSE114_S17_Labtest2_JiHyeWoo.zip
CSE114_S17_Labtest_2_DaYeEun.zip
CSE114_S17_Labtest_2_DoYoungEum.zip
CSE114_S17_Labtest_2_EunjiJang.zip
CSE114_S17_Labtest_2_KyoungaWoo.zip
CSE114_S17_Labtest_2_SangUkLee.zip
CSE114_S17_Labtest_2_SoYoonJeon.zip
CSE114_S17_Labtest_2_SuInCho.zip
CSE114_S17_Labtest_2_YoungWonChoi.zip
Untitled.zip
test3
tmp
```

```
tmp
    test2
        CSE114_LabTest_2_WoohyungLee.zip
        CSE114_S17_LabTest2_SeungHwaLee.zip
        CSE114_S17_Labtest_02_YeonjinOh.zip
        CSE114_S17_Labtest_2_HayoungSon.zip
        CSE114_S17_Labtest_2_KyoungaWoo.zip
        CSE114_S17_Labtest_2_SangUkLee.zip
        CSE114_S17_Labtest_2_SunYoungPark.zip
        CSE114_S17_Labtest_2_ZionLee.zip
        Untitled.zip
    test3
        CSE114.S17_Labtest_2_HyungTaekKwon.zip
        CSE114_S17_Labtest2_JiHyeWoo.zip
        CSE114_S17_Labtest_2_DaYeEun.zip
        CSE114_S17_Labtest_2_DoYoungEum.zip
        CSE114_S17_Labtest_2_KyoungaWoo.zip
        CSE114_S17_Labtest_2_SangUkLee.zip
        CSE114_S17_Labtest_2_SoYoonJeon.zip
        CSE114_S17_Labtest_2_SuInCho.zip
        CSE114_S17_Labtest_2_YoungWonChoi.zip
        Untitled.zip
```