8. file in&out, data structure
August 30, 2015

Minhaeng Lee
• **Stack**
  – First in, last out

• **Queue**
  – First in, First out (FIFO)

• **List**
  – Versatile list typed data structure

• **Tree**
  – Node connected to its children
  – Binary tree: tree has at most two children

• **Map**
  – Also called hash map
  – Data mapped via hash function
  – Can access elements via “Object” called key
Stack

- First in – last out
- `push()` and `pop()` method
- Useful to solve searching problem
  - Think, when you got deadend, then?

Let’s run example code!
Queue

- First in, First out
- Useful for time work scheduling
List

• Super class: *List*
  • *List a = new Vector();*

• Example class: *Vector*
  – Let’s run example code (Uploaded in our website)

• Useful for various purposes

Let’s run example code!
Tree

• Stems from Root
• Each element is called node
• Each node may have its own children
• Useful for data search, sorting

Figure: Tree data structure
Map

- Keys are connected to their data
- Hash function converts keys into index
- Well-formed hash function does not allow index duplication

Let’s run example code!
File read/write

• How to read and write file
• Useful to save data
• Useful to use pre-defined parameter

• PrintWriter
  – When we write file
• BufferedReader
  – When we read file
package fileinout;
import java.io.FileWriter;

public class PrintWriterDemo {
    public static void main(String[] args) throws Exception {
        String filename = "fileName.txt";
        String[] linesToWrite = new String[] { "abc", "bddd" }; 
        boolean appendToFile = true;

        PrintWriter pw = null;
        if (appendToFile) {
            pw = new PrintWriter(new FileWriter(filename, true));
        } else {
            pw = new PrintWriter(new FileWriter(filename));
        }
        for (int i = 0; i < linesToWrite.length; i++) {
            //Use like System.out.println
            pw.println(linesToWrite[i]);
        }
        pw.flush();
        pw.close();
    }
}
Buffered Reader – type it!

```java
package fileinout;

import java.io.BufferedReader;
import java.io.FileReader;

public class BufferedReaderDemo {
    public static void main(String[] args) throws Exception {
        String thisLine = null;
        try{
            // open input stream test.txt for reading purpose.
            FileReader fr = new FileReader("fileName.txt");
            BufferedReader br = new BufferedReader(fr);
            while ((thisLine = br.readLine()) != null) {
                System.out.println(thisLine);
            }
            br.close();
        }catch(Exception e){
            e.printStackTrace();
        }
    }
}
```
Where is the written file?

If you can’t see, do Refresh (F5) list
Practice: Object Read and Write

- Read Student.java
  - Set/get functions
  - toString
- Read ScoreGeneratorDemo.java and run it
- Check generated file

- Complete
  - Score Reader Demo
  - Read from txt file and then save them to Vector as Student Object
Sorting

• Bubble sort, merge sort, quick sort...
  – You can learn them in college if you choose CS

• We just need to how to use it.

• Use Collections.sort function
  – Use Comparator object
Practice

• Read
  – Sorting code
  – `compareTo` function in `Student.java`
    • Negative values : false
    • Positive values : true
  – Comparator
  – What is current sorting criteria?

• Convert
  – Sort based on mean score.
Draw Examples

• Open codes in drawExample
  – Run BouncingBall.java
Activity Diagram

{Client is registered with the brokerage system}

Client enters login name and password

Correct Login and password?

[Yes] Client has successfully logged in

[No] Invalid login/password

Client's settings are displayed

{Client logs into the system.}
Graphical User Interface (GUI) design
Homework

• Next week (September 6\textsuperscript{th})
  – Use MS powerpoint
  – Bring your idea to program
    • Name and what it is
  – Draw GUI
    • and Activity Diagram – optional
    • As detail as possible
  – At least 2 slides.
Questions
References

• IBM