Lecture 9-16-16

Example of Program Heading:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Compilation: javac Fibonacci.java

\* Execution: java Fibonacci N

\* Programmer: James Papademas

\* Lab # 1

\* Date: 9.2.15

\*

\* Description: Computes and prints the first N Fibonacci numbers.

\*

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**Interfaces 🡪 compare abstraction vs. interfaces**

**Array/ArrayList**

**Interfaces**

* Helpful due to lack of multiple inheritance constraint in java
* Multiple interfaces allowed for a class
  + class Bank implements A, B, C
* all fields are final and static
* java 8 🡪 “new item” default method can be defined
* interface objects can be created *but* they must reference same class type

interface A IA = new Client();

**Array – fixed**

Do class

Methods method (primitive or reference based)

int grades[];

Integer grades[] = new Integer [10];

School grades[] = new School [100];

**ArrayList – dynamic**

Class methods available (built in reference based)

ArrayList <Integer> grades = new ArrayList <Integer> (10); 🡪 10 is the constructor

***Student.java***

public abstract class Students

{

abstract void read();

abstract void process();

abstract void print();

}

***StudentRecords.java***

Public class StudentRecords extends Students

{

// Create objects for processing.

StudentRecords rObjs[];

ArrayList List <List<String>> array = new ArrayList <List<String>>();

}

@Override

void read()

{

// TODO Auto-generated method stub

String line = “ “;

// Use Try with resources statement for file IO

try((BufferedReader br = new BufferedReader(new FileReader(“student-Detail.csv”)))

{

int index = 0;

while(line = br.readLine() != null)

{

// Read from file

array.add(Arrays.asList(line.split(“,”)));

System.out.println(array.get(index++));

}

}

catch (FileNotFoundException e)

{

//TODO Auto-generated catch block

e.printStackTrace();

}

catch (IOException e)

{

//TODO Auto-generated catch block

e.printStackTrace();

}

}