# 1.00 Tutorial 4

## Access, Static, Arrays, ArrayLists

# Things to be discussed….

- Quiz 1 Logistics
- Concepts/key ideas you don’t understand !
- Access
- Static data and Static methods
- Arrays and ArrayLists
- Exercise using Eclipse
  - Please download today's Exercise Example from MIT server if you haven’t.

# Quiz 1 Logistics

- Friday March 4 class time (3 to 4.30pm)
- Topics Included
  - Lectures 1 to 11
  - PS 1 to PS 3
- Open Book, Open Notes, **NO** Laptops
- Optional Quiz 1 Review on March 2 (7-9pm).

# Stuff

- How to import external files into an eclipse project
- How to use javadoc (either on the computer or on the website)

# Access

- private
  - only visible to methods which belong to the same class
- package/default (no access modifier)
  - only visible to methods which belong to the same package
- public
  - visible to all methods

# Static

- Static members:
  - are not associated with any particular instance of the class—one copy shared by all instances
  - are accessible to both static and non-static methods
- Static Methods:
  - may only access static members, **not** instance members
  - -may be called using Classname.methodName() or objectReference.methodName()

# When to Use Static Methods

- When no access to any instance field is required. Usually one of two scenarios:
  - The method takes in all the information it needs as parameters:

    ```
    Math.pow(double base, double exp)
    ```

  - Or, the method needs access to only static variables.

# Arrays vs. ArrayLists

- Arrays are fixed in size;
- Arrays can hold both Objects and primitive types;
- Arrays can only hold elements of the same type.

- ArrayLists can grow as needed
- ArrayLists can hold only Objects (no primitives types!)
- ArrayLists can hold Objects of different types

# Using Arrays

Three things to do:

- Declare an array

```
Integer[] myIntObject; // Array of Object
int[] myIntPrimitive ; //Array of primitive data
```

- Create an array

```
myIntObject   = new Integer[2];
myIntPrimitive   = new int[2];
```

- Create/initialize each object in the array

```
myIntObject[0]   = new Integer(1);
myIntObject[1]   = new Integer(2);
myIntPrimitive[0]= 1;
myIntPrimitive[1]= 2;
```

# Shortcuts

- Declaring and creating in one step:

```
Integer[] myInts=new Integer[2];
```

- Sometimes can declare, create, and initialize all in one step!

```
/* Creates an object w/o new keyword! */
int[] powers={0,1,10,100};
int[] powers = {0,1,10,100};
String[] tas = {“Sakda”, “Felicia”};
Integer[] ints = {new Integer(1), new Integer
  (10)};
```

- Use arrayName.length to get # elements

# Using ArrayLists

- javadoc!
- Must import java.util.*;
- Common constructors: (is an example of constructor overloading)
  - `public ArrayList()`
  - `public ArrayList(int initialCapacity)`
- Adding to a ArrayList
  - `public boolean add(Object o)`
  - `public void add(int index,Object o)`
- Getting things out
  - `public Object get(int index)`
  - Must cast object back to its real type!

    ```
    String someObj=(String)someArrayList.get(1);
    ```
- Other methods:
  - `int size()`
  - `Object set(int index, Object obj)`
  - `Object remove(int index)`
  - `boolean isEmpty()`
  - ………

# Exercise Overview

- Description
  - We want to calculate the total cost on tuition and books for MIT students, where

    Total cost = tuition + (sum of cost of textbooks of all the courses taken)
- For a particular student, the output should have the following format

```
Courses for Student with ID : 123
Course No 1.00 , Book price = $80.0
Course No 2.00 , Book price = $120.0
Course No 6.00 , Book price = $208.0
MIT tuition : $15300.0
TOTAL COST = $15708.0
================================
```

- We will model the problem using the following java classes
  - Student, Course & StudentTest (main() method)

# Let’s get started !

## Create Student Class

- Write the Student class with  the following private data members
  - Student Name,
  - Student ID.
- Add  to the class a constant data member to represent MIT tuition.
  - What type of data member should be used ?? Why ??
  - Assign it a constant value of 15300.0
- Write the class constructor
- Provide any getXXX and setXXX methods.

You can generate get and set methods in eclipse using Source->Generate Getters and Setters.

# Student Class

```
public class Student {

   private String name;
   private int id;

   public static final double TUITION = 15300.0;

   public Student(String n, int i) {
        name = n;
        id =i; }

   public int getId() { return id; }
   public String getName() { return name; }
   public void setId(int i) { id = i; }
   public void setName(String string) {
   name = string; }
}
```

# Create CourseTaken Class

- Write the CourseTaken class with private data members as
  - Course number,
  - Text book name
  - Text book cost
- Write the class constructor
- Provide any getXXX and setXXX methods.

# CourseTaken class

```
public class CourseTaken {
  double bookPrice;
  String bookName;
  String courseNo;

  public CourseTaken(String no, double bp, String bN) {
    bookPrice = bp;
    courseNo = no;
    bookName = bN;}

  public double getBookPrice() { return bookPrice;}
  public String getCourseNo() {return courseNo;}
  public String getBookName() {return bookName;}
  public void setBookPrice(double d) {bookPrice = d;}
  public void setCourseNo(String string) {courseNo = string;}
  public void setBookName(String string) {bookName = string;}

}
```

# StudentTest Class (1 of 3)

- Write a StudentTest class with the main() method
- In the main method,
  - Create 3 Student objects and store them in a ArrayList.
  - Create an array of 3 CourseTaken  objects.
- Now, compile before continuing!
  - Use debugger to step through the code to read your code, even if you think it’s correct, to check it.

# StudentTest Class Code

```
public class StudentTest {

   public static void main(String[] args) {
     //Create a List of Student Objects and stores in a
ArrayList
     Student s1 = new Student(“Student1", 123);
     Student s2 = new Student(" Student2 ", 234);
     Student s3 = new Student(" Student3 ", 345);
     ArrayList students = new ArrayList();
     students.add(s1);
     students.add(s2);
     students.add(s3);

     //MIT Courses
     CourseTaken[] course = new CourseTaken[3];
     course[0] = new CourseTaken("1.00", 80.0, "Big Java");
     course[1] = new CourseTaken("2.00", 120.0, “Mechanical");
     course[2] = new CourseTaken(“6.00", 90.0, “Computer Sc");
     //We will adding more code here
    ..............
    }
}
```

# StudentTest Class (2 of 3)

- **You are given the code skeleton of StudentTest class for this task.**

  **main method continued….**
  - Prompt user to enter student ID using JOptionPane and store the ID using an int variable. Entry of 0 for ID indicates end of user input
  - For each student with non zero ID,
    - Create a variable for total cost with initial value = value of MIT tuition
      - **How will you access the data member TUITION defined in Student Class ?**
    - Initialize a ArrayList `v` . (This will maintain a list of all coursesTaken objects for the student).
    - Prompt for courses taken.
      - End of course taken entry is indicated by 0.
    - For each course entered,
      - Retrieve the CourseTaken object corresponding to course number entered.
      - Add the book cost for the course to total cost variable
      - Store the retrieved courseTaken objects in the ArrayList v
    - Print all the necessary information for the students using the ArrayList `v` and the “total cost” variable.

**Please complete the main method.**

## StudentTest Class Code Skeleton……

```
public class StudentTest {
  public static void main(String[] args) {
  .........
  int id = 0;
  do {
       String sid = JOptionPane.showInputDialog("Enter the Student ID. Enter 0 when done");
       id = Integer.parseInt(sid);
       if( id != 0) {  //For Students with non zero IDs
       A. Initialse a ArrayList v

          double totalCost =  ??? //SET THIS EQUAL TO THE TUITON DEFINED IN STUDENT CLASS
          String s = "0";
          do {
              s = JOptionPane.showInputDialog("Enter the course No. Enter 0 when done");
        for (int j = 0; j < course.length; j++) {
            B. For each of the CourseTaken Object
                 -  If the course no equals user input course No s:
                . add the book cost for that course to total cost.
                            . Store the retrieved courseTaken objects in the ArrayList v
         } } while (!s.equals("0"));
            System.out.println("Courses for Student with ID : " + id );
      C. - Retrieve "CourseTaken" objects stored in v and
               - Print its course no and bookprice

            System.out.println("MIT tuition : $"+ Student.TUITION);
            System.out.println("TOTAL COST = $"+totalCost);
       } //end of check for non Zero IDS
    } while (id != 0);
  } } //end of class
```

# StudentTest Class (3 of 3)

- Compile and run
  - The Output should have the following format

```
Courses for Student with ID : 123
Course No 1.00 , Book price = $80.0
Course No 2.00 , Book price = $120.0
Course No 5.00 , Book price = $208.0
MIT tuition : $15300.0
TOTAL COST = $15708.0
================================
```

Step through your code using the debugger