MODULE

Week 9 Workshop Walk-throughs, Tute Discussions and Lab Tasks

Stephen Huxford, Marc Cheong
Updated 15 September 2021

Week 9 Tutorial Discussions

T1 (50 mins)

In your tutorial you will again be walked through typical section D code but using a PhonePlan class this time.

PhonePlan

Instant variables	name (of the plan)	50 chars max
	period	12 .. 48 months inclusive
	totalCost	>= 0.0
	dataRollover	true/false
Constructors	0-parameter	period = 12, rollover = true
	3-parameter name, period, rollover	totalCost to be manually initialised (do not rely on default)
Accessors		At least one
Mutators		At least one with guardian code and Boolean return value indicating success of mutation
Others	toString	Report the complete state
	discount001	If period >= 24 and data rollover false then data rollover = true and reduce total cost by supplied amount or set it to 1000.00 whichever is higher

Week 9 Lab Tasks

Download this application shell. It contains a partially developed application.

In this week, imagine you are a programmer who has been asked to develop an application for a university that has the following attributes.

INSTANCE VARIABLE	DESCRIPTION
universityName	at least 3 characters
worldRank	a positive non-zero integer
faculties	An ArrayList of strings that represents the list of faculties. The minimum length of eacl string is 3 characters.
students	An ArrayList of strings, where each string represents a student name. The minimum length of each string is 2 characters.

INSTANCE VARIABLE	DESCRIPTION
enrollment	An ArrayList of strings, where each string must have two segments separate by '|'. The segment represents the student name and the second segment represents the faculty name

Now, your tasks are:

L1 (0.2 marks)

Code the five instance variables that are described in the table above.

L2 (0.3 marks)

Code setters and getters (mutators and accessors) for:

- universityName
- worldRank

Code getters only for:

- faculties
- students
- enrollment

The setters must return true if the provided value is valid and set to the instance variable; false otherwise.

L3 (0.2 marks)

- Code a 2-parameter constructors for the (universityName,worldRank) instance variables. The instance variables faculties, students, and enrollment must be initiated with empty ArrayLists.

L4 (0.1 marks)

Code a toString method that displays the value of all the instance variables.

L5 (0.2 mark)

Code a boolean method called 'addStudent' that accepts a string as an input represents a student name. The method should return true if the input is valid (length of name>=2) and add it to the ArrayList "students". If the input is invalid, the method must return false.

L6 (0.2 marks)

Code a boolean method called 'addFaculty' that takes a string as input represents a faculty name. The method should return true if the input is valid(length of faculty >=3) and add it to the ArrayList "faculties". If the input is invalid, the method must return false.

L7 (0.5 marks)

Code a boolean method called 'addEnrollment' that takes two strings as input to represent a student name a faculty name. If both input strings 'name' and 'faculty' exist in the 'students' and 'faculties' array lists respectively, the method should add them to the 'enrollment' ArrayList using the aforementioned format and return true; otherwise, the method must return false.

L8 (0.3 marks)

Code a method called **countStudentsPerFaculty** that accepts a string as input represent a faculty name and returns an integer value that represents the number of students in that faculty.

For example:

If the enrollment arraylist has [“John|FIT”,”Tim|Law”,”Emma|FIT”]

- Input “FIT”
  - return 2
- Input “Law”
  - return 1
- Input “Business”
  - return 0

Hints: methods might help you to achieve task L8

- Java String charAt() Method
- Java substring() Method

Expected Output

If you execute the following code:

```
University monash=new University("Monash",20);
monash.addFaculty("FIT");
monash.addFaculty("Law");
monash.addFaculty("Business");
monash.addFaculty("La");//rejected
System.out.println("Faculties="+monash.getFaculties()); //test faculties OK
monash.addStudent("John");
monash.addStudent("Tim");
monash.addStudent("Emma");
monash.addStudent("E"); //rejected
System.out.println("Students="+monash.getStudents());
monash.addEnrollment("John","FIT");
monash.addEnrollment("Emma","FIT");
monash.addEnrollment("Tim","Law");
System.out.println("Enrollments="+monash.getEnrollment()); // test enrollment OK

int studentsInFIT=monash.countStudentsPerFaculty("FIT");
int studentsInLaw=monash.countStudentsPerFaculty("Law");
int studentsInBusiness=monash.countStudentsPerFaculty("Business");
System.out.println("Number of students in FIT ="+studentsInFIT);
System.out.println("Number of students in Law ="+studentsInLaw);
System.out.println("Number of students in Business ="+studentsInBusiness);
```

and here is the expected output:

```
Faculties=[FIT, Law, Business]
Students=[John, Tim, Emma]
Enrollments=[John|FIT, Emma|FIT, Tim|Law]
Number of students in FIT =2
Number of students in Law =1
Number of students in Business =0
```
