Lab Assignment 6

1) Objective: To understand the usage and the implementation of constructors.

Problem Description: Create a class to store the employee information and calculate the average feedback and grade based on their feedback. Use the **default constructor** to initialize the member variables. Create the starter class to instantiate the employee and invoke the methods.

Solution

Step 1: Create the class EmployeeGrade under the working folder based on the below given class diagram.

```
EmployeeGrade
- employeeNo : int
- employeeName : String
- customer1Feedback: float
- customer2Feedback: float
- customer3Feedback: float
- averageFeedback : float
- grade : char
+ EmployeeGrade()
+ calculateAverageFeedback() : void
+ calculateGrade() : void
+ displayInfo() : void
```

EmployeeGrade() : This is the default constructor which initializes the member variables with values as follows.
Employee No :101,
Name: “Ram”,
customer1Feedback: 4.1f,
customer2Feedback: 4.0f,
customer3Feedback: 4.3f

The implementation of the default Constructor – EmployeeGrade would be as follows.

```
public EmployeeGrade() {
employeeNo = 101;
employeeName = “Ram”;
customer1Feebdack = 4.1f;
customer2Feedback = 4.0f,
customer3Feedback = 4.3f
}
```

Step 2: Compile EmployeeGrade and fix the errors, if any.

Step 3: Create a starter class with the name Starter as given below.

```
Filename Starter.java
/*
* This java file is a starter class which instantiates
EmployeeGrade.
* Calculate the average feedback and the grade
* Display the information
*/

public class Starter {
/**
* Instantiate the EmployeeGrade. Set the values for the member
* variables and invoke the method to calculate the average
* feedback and grade. Invoke the method to display the employee
* information.
* @param args The command line arguments
*/
public static void main (String[] args) {
// To-do: Create instance(ram) for EmployeeGrade.
// To-do: Invoke methods for calculating Avg feedback & grade
// To-do: Invoke method to display the employee information
}
}
```

Save the file as : Starter.java.

Step 4: Compile Starter.java. Fix the errors, if any.
Step 5: Execute Starter.java. Check the output. It should display the information about employee 101.

2) **Objective:** To understand the usage and the implementation of constructors.

Problem Description: Create a class to store the employee information and calculate the average feedback and grade based on their feedback. Use the **parameterized constructor** to initialize the member variables. Create the starter class to instantiate the employee and invoke the methods.

Note: *When the parameterized constructors are defined but not the default constructor, if you try to invoke the default constructor, you would get compilation error.*

What will happen, if we try to make constructor as private and invoke it from a different class…..

3) Objective: To understand static blocks, variables and methods.

Problem Description: Create a class that has a static variable, block and method that keeps the count of student who undergo course. The static block should initialize the value of the static variable and the method should return the value of the static variable.

Note: Assumption is there are already 100 students available. Hence the student count should start from 101.

Step 1: Create Course.java and StudentInfo.java under the working folder for the assignment as given below.

```
Filename Course.java
/*
* This java file is a class which
* uses static blocks, variables and methods
* to maintain the count of student in a course.
*/
public class Course {
private static int studentCount;
private int courseId;
private int studentId;
// Static Block
static{
// To -Do: Initialize the static variable to 100.
System.out.println(studentCount);
System.out.println(courseId);
}
Course() {
studentCount++;
studentId = studentCount;
}
/**
* Returns the value of the static variable
*/
public static int getStudentCount (){
// To-Do: Return the value of the static variable
}
/**
* Returns the value of courseId
*/
public int getCourseId () {
// To-Do: Return the value of courseId
}
/**
* Sets the value of courseId
*/
public void setCourseId (int courseId){
// To-Do: Set the value of courseId
}
/**
* Returns the value of studentId
*/
public int getStudentId () {
// To-Do: Return the value of studentId
}
}
```

Static blocks are executed automatically when the class is loaded. If there is more than one static block, then they are executed in the order in which they are coded.

Step 2: Compile the program. Fix the errors

The static block (be it a block or method) can access only the static members. Hence, the System.out.println(courseId) would raise an error at the time of compilation. Remove the statement within the static block.

Step 3: Create a starter class StudentInfo as given below.

```
Filename StudentInfo.java
/*
* This java file is a starter class
* that invokes the methods of Course class.
*/
public class StudentInfo {
public static void main (String args []) {
System.out.println("No. of students already in the course : " +
Course.getStudentCount());
Course student = new Course ();
// Setting the courseId for the student
Course.setCourseId(1001);
System.out.println ("Student ID : " +
student.getStudentID() + " Course ID: "
+student.getCourseId());
System.out.println ("Student Count: "
+Course.getStudentCount());
Course student1=new Course();
Course1.setCourseId(1002);
System.out.println ("Student ID : " +
student1.getStudentID() + " Course ID: "
+student1.getCourseId());
System.out.println ("Student Count: "
+Course.getStudentCount());
// Can also call static methods using objects
System.out.println ("Student Count: "
+student1.getStudentCount());
}
}
```

Step 4: Compile StudentInfo.java. Fix the error, if any.
Step 5: Execute the program

Note: *Objects need not be created to call static methods. These methods can be called using the class name only.*