Department of Statistics & Computer Science
University of Kelaniya
Academic Year – 2022/2023
COSC 12043 / BECS 12243 - Object Oriented Programming
Tutorial 10

---

1. The following Java programs have errors. Point out the statement(s) that contain errors. Then you need to explain what each of the errors is, and how it can be fixed.

   a.
```
public class OOPExercises {
    public static void main(String[] args) {
        A objA = new A();
        System.out.println("objA.a = "+objA.a);
        objA.a = 222;
    }
}
public class A {
    private int a = 100;
    public void setA( int value) {
        a = value;
    }
    public int getA() {
        return a;
    }
}
```

   b.
```
public class OOPExercises {
    public static void main(String[] args) {
        System.out.println("objA.a = "+getA() );
        setA(123);
    }
}
public class A {
    private int a = 100;
    public void setA( int value) {
```

```
        a = value;
    }
    public int getA() {
        return a;
    }
}
```

c.
```
public class OOPExercises {
    public static void main(String[] args) {
        A objA = new A( );
        double result = 55;
        objA.setA(result);
        System.out.println("objA.a = "+ objA.getA());
    }
}
class A {
    private int a = 100;
    public void setA( int value) {
        a = value;
    }
    public int getA() {
        return a;
    }
}
```

d.
```
public class OOPExercises {
    public static void main(String[] args) {
        A objA = new A( );
        objA.setA(55);
        System.out.println("objA.a = "+ objA.getA());
    }
}
```

```
class A {
   final int a = 100;
   public void setA( int value) {
      a = value;
   }
   public int getA() {
      return a;
   }
}
```

2. Write a class called Point which has two fields to store x and y coordinates of a point. This class should include the following:

   public Point(int a, int b): Constructs a new point that contains the given coordinates.

   public String displayPoint(): Returns a String representation of a point such as (11, 4).

   a. Declare the method distance() in the Point class that calculates the distance of the point from (0,0).
   b. Also we need to define another method with the same name "distance" which calculates the distance of the point from a given point (passed as a parameter).
   c. Write a client class named PointClient. It should create two objects of class Point and initialize them to (20, 2) and (3, 6). Print the two-point objects.
   d. Print out the distance of (3, 6) from (0,0) inside PointClient.
   e. Print out the distance of (3, 6) from (20, 2) inside PointClient.

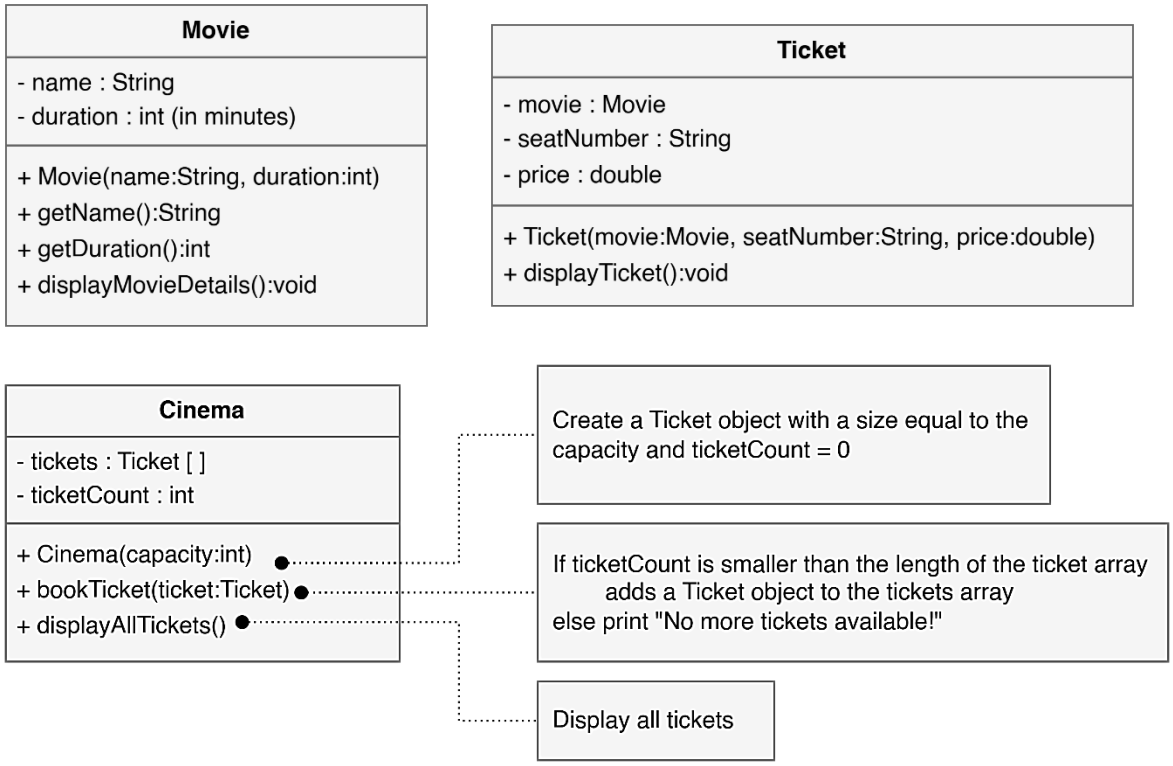
3. Design a class named **Employee** with instance variables name, idNumber, and payRate, and a static variable employeeCounter initialized to zero to track the total number of employees. The class should include a constructor that accepts name and payRate as parameters, increments the employeeCounter, and sets the idNumber to employeeCounter + 1000. It should also have a method calcPay(double hours) to calculate the total pay for the employee based on the number of hours worked and a static method numberOfEmployees() to return the current number of employees.

   Create a class named **EmployeeTest** to test all functionalities of the Employee class by creating multiple instances with sample data, testing the calcPay method, displaying details of each employee (name, ID, and calculated pay), and verifying the numberOfEmployees method to ensure it accurately tracks the total employees.

Sample Output:

```
Name: Saman, ID: 1001, Pay Rate: Rs.1000.0
Name: Amal, ID: 1002, Pay Rate: Rs.2500.0
Name: Kamala, ID: 1003, Pay Rate: Rs.1250.0
Saman's Pay for 40 hours: Rs.40000.0
Amal's Pay for 35 hours: Rs.87500.0
Kamala's Pay for 20 hours: Rs.25000.0
Total Employees: 3
```

4. Create a Java program with three separate files for classes Movie, Ticket, and Cinema, forming a simple cinema ticket booking system. Based on the information provided in the diagram, write a Java program to implement the classes.



Below is a test driver to test the classes. Run the test driver to get the output of the code.

```
public class CinemaBookingSystem {
      public static void main(String[] args) {
            Movie movie1 = new Movie("Avatar 2", 180);
            Movie movie2 = new Movie("Interstellar", 165);

            Cinema cinema = new Cinema(3);
            cinema.bookTicket(new Ticket(movie1, "A1", 1200.00));
            cinema.bookTicket(new Ticket(movie2, "B5", 1500.00));
```

```
            cinema.displayAllTickets();
        }
    }
```

5. Create a class **Patient** with the attributes patientID (String), name (String), and disease (String). Add a method displayDetails() to print the patient's details. Create a **Doctor** class in a separate Java file with the attributes doctorID (String), name (String), and specialization (String). Include a method treatPatient(Patient patient) in the Doctor class to display a message in the format:
   `Dr. [Doctor's Name] is treating [Patient's Name] for [Disease].`
   Write a main method to create two Patient objects, initialize their details, and display their information using the displayDetails() method. Simulate a doctor treating a patient using the treatPatient() method.

6. Create a Java program to manage orders in a restaurant. The program should include a **Dish** class with attributes for the name and price of the dish, a constructor to initialize these attributes, and a method to display the dish details. It should also include an **Order** class with attributes for the dish (as a Dish object) and the quantity of the dish ordered, along with a constructor to initialize these attributes and a method to calculate the total price of the order (price multiplied by quantity). Include a **Restaurant** class that uses an array to store multiple Order objects, with the array size fixed to represent the maximum number of orders the restaurant can handle. Add methods to add an order to the array and display all orders with their total prices.
   a. Write the Java program with **separate files for each class**, create at least two dishes, place at least two orders, and display all the orders and their total prices.
   b. Run the following code segment to see the output.

```
public class RestaurantManagement {
    public static void main(String[] args) {
        Dish dish1 = new Dish("Pasta", 550.00);
        Dish dish2 = new Dish("Pizza", 1200.00);

        Restaurant restaurant = new Restaurant(3);
        restaurant.addOrder(new Order(dish1, 2));
        restaurant.addOrder(new Order(dish2, 1));
        restaurant.addOrder(new Order(dish2, 5));
        restaurant.addOrder(new Order(dish1, 3));
    }}
```