Topics-Abstract Classes

1) **Create an abstract class Compartment to represent a rail coach. Provide an abstract function notice in this class:**
   Derive FirstClass, Ladies, General, Luggage classes from the Compartment class.
   Override the notice function in each of them to print a notice message that is suitable to the specific type of compartment.
   Create a class TestCompartment. Write the main function to:
   Declare an array of Compartment of size 10.
   Create a compartment of a type decided by a randomly generated integer in the range 1 to 4.
   Check the polymorphic behavior of the notice method.

Ans

```
abstract class Compartment {
   public abstract String notice();
}

class FirstClass extends Compartment {
   public String notice() {
      return "This is a First Class compartment.";
   }
}

class Ladies extends Compartment {
   public String notice() {
      return "This is a Ladies compartment.";
   }
}

class General extends Compartment {
   public String notice() {
```

```
        return "This is a General compartment.";
    }
}

class Luggage extends Compartment {
    public String notice() {
        return "This is a Luggage compartment.";
    }
}

public class TestCompartment {
    public static void main(String[] args) {
        Compartment[] compartments = new Compartment[10];
        for(int i = 0; i < compartments.length; i++) {
            int rand = (int)(Math.random() * 4) + 1;
            switch(rand) {
                case 1: compartments[i] = new FirstClass(); break;
                case 2: compartments[i] = new Ladies(); break;
                case 3: compartments[i] = new General(); break;
                case 4: compartments[i] = new Luggage(); break;
            }
        }

        for(int i = 0; i < compartments.length; i++) {
            System.out.println("Compartment " + (i+1) + ": " + compartments[i].notice());
        }
    }
}
```