SET-I

19BQ1A05L1
SK. Vaseem Naazleen

Q1. Write about the role of JVM, JAVA API in developing the platform independent java program with suitable example.

JVM (Java Virtual Machine) is an abstract machine. It is specification that provides runtime environment in which java bytecode can be executed.

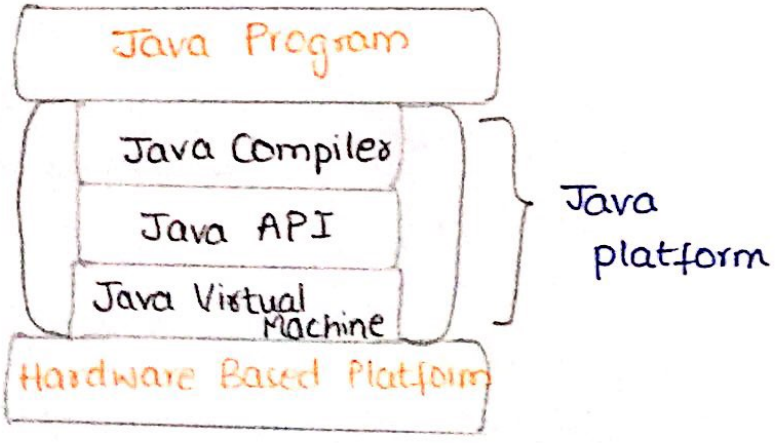
JVM are available for many hardware and software platforms (i.e, JVM is platform dependent)

JVM take care of creating of class file related to your program and you can use same class file in different operating system.

JVM acts as interface between your OS and your program code.

1. It also responsible for calling main method and running your program.
2. It also provides security related features
3. It is also responsible for calling garbage collector for clean up operations.

Java Application Programming Interface (API) is the area of Java development kit JDK. An API includes classes, Interfaces, packages and also their methods, fields, and constructors.



The JAX-RS (Java API for Restful Web Services) Client provides following ~~categories &~~ capabilities:

- Invoke REST endpoints
- Use Similar paths/templates as sever API async invoker
- Improved support.
- Sever-Sent Events support

Examples

- Using Java API top sites like Twitter, Yahoo uses this kind of pattern
- Most social media sites like facebook.com benefits REST web services.
- Mobile App development, is developing quickly as well as their sever connection, by using this REST pattern since it is quicker in processing request and response data.

Q2: Write ~~about the role~~ an example program explain the concept of nested classes in Java.

In Java, it is possible to define a class within another class, such classes are known as nested classes They enable you to logically group classes that are only used in once place, thus this increase the use of encapsulation, and creates more readable & maintainable code.

- To scope of a nested class is bounded by the scope of its enclosing class. Thus in above example, class Nestedclass doesnot exist independently of class Outer Class.
- A nested class has access to the members, including private members, of the class in which it is nested, however, the reverse is not true i.e, the enclosing class doesnot have access to the members of the nested class.
- A nested class is also a member of its enclosing class.
- As a member of its enclosing class, a nested class can be declared private, public, protected or package private (default)
- Nested classes are divided into 2 categories:
  1. static nested class: Nested classes that are declared static are called static nested classes.
  2. inner class: An inner class is a non-static nested class.

Syntax:

```
class OuterClass {
  ...
  class NestedClass {
    ...
  }
}
```

Program:

```
public class College {

    static int numOfStudents;
    static int numOfFaculty;

    class Student {
        private String name;
        private String rollNO;
        private String course;

        public Student() { numOfstudents ++; }
        public Student (String name, String rollNO, String course) {
            this.name = name;
            this.rollNO = rollNO;
            this.course = course;
            numOfcourse++;
        }
        public String getName() {
            return name;
        }
        public void setName (String name) {
            this.name = name;
        }
        public String getRollNO() {
            return rollNO;
        }
        public String getCourse() {
            return course;
        }
                                    String rollNO
        public void setRollNO(a) {
            this.rollNO = rollNO;
        }
        public void setCourse (String course) {
            this.course = course;
        }
        public string getInformation() {
            return rollNO + " who's name is " + name + "studies"
                                + course;
        }
```

```
}
class Faculity {
    private String name;
    private String subject;
    private int salary;
    public Faculity() { numOfFaculity++; }
    public Faculity(String name, String subject, int salary) {
        this.name = name;
        this.subject = subject;
        this.salary = salary;
        numOffaculity++;
    }
    public String getInformation() {
        return name + " teaches " + subjects + " and earns " +
                                    String.valueOf(salary);
    }
}

//Driver Code
public static void main(String[] args) {
    College VVIT = new College();
    College.Student student1 = VVIT.new Student("Vasceem Naazleen",
                                    "19BQ1A05L1", "CSE");
    College.Student student2 = VVIT.new Student("Sharon",
                                    "19BQ1A05L2", "CSE");

    College.Faculity faculity1 = VVIT.new Faculity("Krishna Prasad",
                                    "Data structure", 40000);
    College.Faculity facuity2 = VIT.new Faculity("Naga Sriharsha",
                                    "Java", 40000);
    System.out.println(student1.getInformation());
    System.out.println(student2.getInformation());

    System.out.println("No.of Students in college: " +
                                    VIT.numofstudents);
    System.out.println("No.of teachers in college: "+VVIT.numof
                                    Faculity);
}
```

Q: Design a class RailwayTicket with the following description: - - -

Ans: Program

```
import java.util.HashMap;
import java.util.Scanner;

public class RailwayTicket {
        private string name;
        private String coach;
        private long moblieNumber;
        private int amount;
        private int totalAmount:
        private HashMap<String, Integer> amounts =
                        new HashMap<string, Integer>(4);

        public void accept() {
                Scanner in = new Scanner(System.in);
                System.out.println("Enter your name: ");
                this.name = in.next();
                system.out.println("Enter your coach: ");
                this.coach = in.next();
                System.out.println("Enter your Moblie number: ");
                this.mobileNumber = in.nextLong();
                System.out.println("Enter amount ");
                this.amount = in.nextInt();

                amounts.put("First_AC", 700);
                amounts.put("Second_AC", 500);
                amounts.put("Third_AC", 250);
                amounts.put("sleeper", 0);
    }
    public void update() {
            totalAmount = amount + amounts.get(coach);

    }
```

```
}
class Faculty {
    private String name;
    private String subject;
    private int salary;
    public Faculty() { numOfFaculty++; }
    public Faculty(String name, String subject, int salary) {
        this.name = name;
        this.subject = subject;
        this.salary = salary;
        numOfFaculty++;
    }
    public String getInformation() {
        return name + " teaches " + subject + " and earns " +
                                    String.valueOf(salary);
    }
}

//Driver Code
public static void main(String[] args) {
    College VVIT = new College();
    College.Student student1 = VVIT.new Student("Vasccm Naazleen",
                                    "19BQ1A05L1", "CSE");
    College.Student student2 = VVIT.new Student("Sharon",
                                    "19BQ1A05L2", "CSE");

    College.Faculty faculty1 = VVIT.new Faculty("Krishna Prasad",
                                    "Data structure", 40000);
    College.Faculty facuity2 = VVIT.new Faculty("Naga Sriharsha",
                                    "Java", 40000);
    System.out.println(student1.getInformation());
    System.out.println(student2.getInformation());

    System.out.println("No.of students in college: " +
                                    VVIT.numOfStudents);
    System.out.println("No.of teachers in college: "+VVIT.numOf
                                    Faculty);
}
```

```
public void display () {
        System.out.println("Name: " +name);
        System.out. println(" Coach: "+ coach);
        system.out.println ("Mobile number : "+ mobileNumber);
        System.out.println (" Total Number : " + totalAmount);

}

public static void main(string[] args) {
        RailwayTicket ticket = new RailwayTicket ();
        ticket. accept();
        ticket. update();
        ticket. display();

}

}.
```

4. Design a class to overload a function volume as follows:

```
public class Volume {
        public double volume (double r) {
                return (4.0 / 3.0) * (22.0 / 7.0) * Math.pow(r, 3);
        }
        public double volume (double h, double r) {
                return (22.0 / 7.0) * Math.pow (r, 2) * h;
        }
        public double volume(double l, double b, double h) {
                return l* b * h;
        }
        public static void main (string[] args) {
                Volume vol = new Volume();
                System.out.println (vol.volume(3));
                System.out.println(vol.volume(3, 4));
                System.out.println (vol.volume(3, 4, 5));
        }
}
```