1.What is Java and why is it platform-independent?

Ans-

- Java is a high-level, object oriented programming language.
- James gosling - father of java.
- Java was invented by a team of engineers at sun microsystem led by james gosling.
- It was started in 1991 and first released in 1995.
- It follows “WORA”- write once run anywhere which makes java platform independent.
- Java code is compiled into bytecode not machine code.

2. Explain the features of Java.

Ans-

- Java is designed to be easy to learn and use.
- Clean and readable syntax.
- It is a oop language which supports four pillars –

1.encapsulation

2.polymorphism

3.inheritance

4.abstraction

- It is platform independent.
- Automatic memory management.
- Java is designed to prevent errors and crashes.
- Java supports multithreading (running multiple tasks at the same time).

3. What is the difference between JDK, JRE, and JVM?

Ans-

JDK:

- Used to develop, compile, and run Java programs.
- Executes bytecode.
- Compiles java code.
- It is platform dependent.
- It is used by developer.
- Toolkit to develop Java programs.

JRE:

- It provides the environment to run Java programs.

- It includes supporting files.
- Core Java class libraries.
- It is not used to develop code.

JVM:

- It is a virtual machine.
- It executes java bytecode.
- Core Java class libraries
- It loads .class files.
- It manages memory.
- JVM is a platform dependent.

4.What is bytecode in Java?

Ans-

Bytecode in Java is the intermediate code generated by the Java compiler that makes Java platform-independent. It is stored in a .class file. Same bytecode runs on Windows, Linux, macOS.

5. Explain the concept of object-oriented programming.

Ans-

Object-Oriented Programming (OOP) is a programming approach that organizes code using objects instead of only functions and logic.

An object represents a real-world entity.

A class is a blueprint for creating objects.

There are 4 pillars of oop's:

1.Encapsulation: Wrapping data and methods into a single class and restricting direct access. It protects data.

2. Inheritance: One class inherits properties and methods from another class.it is used to reuse the code again.

- Using interface we can achieve multiplication inheritance.
- In inheritance, we can override parent class method into child
- Super class reference and sub class object are also allowed.

3. Polymorphism: Is a many form of method and constructor.

Rules are:

For overloading:

1.same name

2.same scope

3.Different parameter.

For overriding:

1.same name

2.same sign/prototype

3.Different scope

4. Abstraction:

- We know what but don't know how.
- Using abstract class and interface we can achieve.
- In abstraction we hide a logic in between methods.
- "abstract" keyword is used.

HELLO WORLD

```
public class Hello {

		public static void main(String[] args) {
			System.out.println("Hello World");
		}
}
```

O/P 1 : Hello World

2:Addition Of two Numbers

```
import java.util.Scanner;

public class Addition {
	public static void main(String[] args) {
		Scanner sc = new Scanner(System.in);

		while(true) {
			System.out.println("Enter Two numbers");
			int a = sc.nextInt();
			int b = sc.nextInt();
```

```
                System.out.println("Addition is "+(a+b));
            }
        }
}
```

```
O/P 2:
Enter Two numbers
2 3
Addition is 5
Enter Two numbers
```

3: Largest Of three numbers

```
import java.util.Scanner;

public class Largest {

        public static void main(String[] args) {
            Scanner sc = new Scanner(System.in);

            while(true) {
                System.out.println("Enter Three Numbers ");
                int a = sc.nextInt();
                int b = sc.nextInt();
                int c = sc.nextInt();
                b=a>b?a:b;
                b=c>b?c:b;

                System.out.println("Largest Number is "+b);

            }
        }
}
```

```
o/P:
 Enter Three Numbers

2 3 4

Largest Number is 4

Enter Three Numbers

3 4 5

Largest Number is 5

Enter Three Numbers
```

4 : Odd Even

```
import java.util.Scanner;

public class OddEven {
```

```
	public static void main(String[] args) {
		// TODO Auto-generated method stub

		Scanner sc = new Scanner(System.in);


		while(true) {
			System.out.println("Enter A Number :");
			int a =sc.nextInt();
			if(a%2==0)
				System.out.println("Even Number");
			else
				System.out.println("Odd Number");
		}

	}

}
```

O/P :

```
Enter A Number :
2
Even Number
Enter A Number :
1
Odd Number
Enter A Number :
4
Even Number
Enter A Number :
3
Odd Number
```

5 : Positive Negative

```
import java.util.Scanner;

public class PositiveNegative {

	public static void main(String[] args) {
		// TODO Auto-generated method stub

		Scanner sc = new Scanner(System.in);
		while(true){
			System.out.println("Enter a Number");
			int a = sc.nextInt();

			if(a<0){
				System.out.println("Number is negative");
			}else if(a>0){
				System.out.println("Number is Positive");
			}else if(a == 0) {
				System.out.println("Number is zero");
			}
		}
	}
```

```
}
O/P :
Enter a Number
2
Number is Positive
Enter a Number
0
Number is zero
Enter a Number
-1
Number is negative
```