1: What is Class and Object in java?

Class :

- A class is a blueprint or template useed to create An objects.
- It contains variables(fields / data members) and methods, constructor blocks and nested class

Object :

- An object is a real instance of class.
- It represents a real world entity. Used to access class variables and methods
- To represent the real data, to perform the operations defined in the class

2: Difference between primitive and reference data types

Primitive Data Types

- Store actual value
- Fixed size
- Faster than reference types
- Cannot call methods
- Default values are provided

Reference Data Types :

- Store memory address
- Size depends on object
- Slower than primitives
- Can call methods
- Default value is null

3 : **What are the access modifiers in java?**

- Private
- default (no keyword)
- protected
- public

private

- Accessible only within the same class
- Most restrictive access modifier

Default

- Accessible within the same package only
- If no access modifier is written, it is default

Protected

- Within the same package
- In subclasses (child classes) outside the package

Public

- Accessible from anywhere
- Least restrictive access modifier

What is encapsulation ?

The process of wrapping variables and methods together into single unit. It used to restrict the direct access to the data. It can achieve by making the variables using private provide getter and setter methods. It is used for Protect data from unauthorized access, Data hiding security and maintainable and flexible.

We can follow following Steps to achieve encapsulation:

1. Declare class variables as private
2. Provide public getter and setter methods to access and update them

What is Inheritance and why it is used ?

Inheritance is the process by which one class (child/subclass) acquires the properties and methods of another class (parent/superclass).It allows code reuse and creates a parent–child relationship between classes.

Inheritance is used to:

1. Reuse existing code
2. Reduce code duplication
3. Improve maintainability
4. Support method overriding
5. Achieve runtime polymorphism

Write Program to swap without using third variable ?

```
import java.util.Scanner;

public class Swap{
	public static void main(String[] args){
	Scanner sc = new Scanner(System.in);
	int a, b;
	System.out.println("Enter values of a and b");
	a= sc.nextInt();
	b=sc.nextInt();
	System.out.println("Before Swapping Values of a is "+a+" and Value of b :"+b);
	a =a+b;
	b=a-b;
	a=a-b;
	System.out.println("After Swapping Values of a is "+a+" and Value of b :"+b);

	}
}
```

Factorial Number :

```
import java.util.Scanner;

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

		System.out.println("Enter value of N");
		int n = sc.nextInt();

		int fact =1;
		for(int i =1;i<=n;i++){
			fact*=i;
		}
		System.out.println("Factorial upto the N is :"+fact);
	}
}
```

Fabonacci Series :

```
import java.util.Scanner;

public class Fibonacci {
   public static void main(String[] args) {
      int n, a = 0, b = 1, c;

      Scanner sc = new Scanner(System.in);
      System.out.print("Enter the value of n: ");
      n = sc.nextInt();

      System.out.print("Fibonacci series up to " + n + ": ");

      while (a <= n) {
         System.out.print(a + " ");
         c = a + b;
         a = b;
         b = c;
      }
   }
}
```

Positive negative or Zero

```
import java.util.Scanner;

public class PositiveNegative {

	public static void main(String[] args) {

		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");
			}
		}
	}

}
```

Reverze Number :

```
import java.util.Scanner;

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

		System.out.println("Enter a Number");
		int n = sc.nextInt();

		int rev =0;
		while(n!=0){
			int rem = n%10;
			rev = rev*10+rem;
			n = n/10;
		}
		System.out.println("Reverse Number is :"+rev);
	}
}
```

Palindrom Number :

```
import java.util.Scanner;

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

		System.out.println("Enter a Number");
		int n = sc.nextInt();
		int rev =0;
		int og = n;
		while(n!=0){
			int rem = n%10;
			rev = rev*10+rem;
			n = n/10;
		}

		if(rev == og){
			System.out.println("Number is Palindrome");
		}else{
			System.out.println("Number is Not Palindrome");
		}
	}
}
```