08-02-2025

Working with static Method with different return Type :

If a static method is available in the ELC class (the class which contains main method) then we can directly call the static method
from main method, On the other hand if a static method is available
in another class then to call the static method class name is required.

//1) Program to print the Table :

```
package com.ravi.static_method_demo1;

//BLC
public class Table
{
  public static void printTable(int num) //10 X 1 = 10
  {
          for(int i=1; i<=10; i++)
          {
                  System.out.println(num+" X "+i+" = "+(num*i));
          }
          System.out.println(".....");
  }

}


package com.ravi.static_method_demo1;

import java.util.Scanner;

//ELC
public class PrintingTable
{
        public static void main(String[] args)
        {
                Scanner sc = new Scanner(System.in);
                System.out.print("Enter the number upto which you want to print :");
      int n = sc.nextInt();

      for(int i=1; i<=n; i++)
      {
```

```
            Table.printTable(i);
        }

        sc.close();

            }

}
```

2) WAP to find out the area of Circle in String format, If the radius is -ve OR Zero then it should return -1.

```
   package com.ravi.static_method_demo1;

public class AreaOfCircle
{
  public static String getAreaOfCircle(double radius)
  {
     if(radius<=0)
     {
            return ""+(-1);
     }
     else
     {
            final double PI = 3.14;
            double area = PI * radius * radius;

            return ""+area;

     }


  }
}


package com.ravi.static_method_demo1;

import java.util.Scanner;

public class FindingAreaOfCircle
{
            public static void main(String[] args)
```

```
	{
		Scanner sc = new Scanner(System.in);
		System.out.print("Enter the radius of the Circle :");
		double radius = sc.nextDouble();

		String areaOfCircle = AreaOfCircle.getAreaOfCircle(radius);

		//Converting String into float
		float area = Float.parseFloat(areaOfCircle);

		System.out.printf("Area of Circle is :%.2f",area);

		sc.close();
	}

}
```

---

3) Finding the square and cube of a number by using following
criteria :
a) If number is 0 or negative should return -1.
b) If number is even, return the square of the number
c) If number is odd, return the cube of the number

```
package com.ravi.static_method_demo1;

public class Calculate
{
  public static int getSquareAndCube(int num)
  {
		if(num <=0)
		{
			return -1;
		}
		else if(num %2==0)
		{
			return num*num;
		}
		else
		{
			return num*num*num;
		}

  }
```

```
}

package com.ravi.static_method_demo1;

import java.util.Scanner;

public class FindingSquareAndCube
{
	public static void main(String[] args)
	{
		Scanner sc = new Scanner(System.in);
		System.out.print("Enter a Number :");
		int num = sc.nextInt();
		System.out.println(Calculate.getSquareAndCube(num));
		sc.close();

	}

}
```

---

4) WAP to take the details of Student and return all the
student details in String format.

```
package com.ravi.static_method_demo1;

public class Student
{
  public static String getStudentDetails(int roll, String name, String addr)
  {
		//[Student roll is : 101, name is : Scott, addrees is : Ameerpet]

		return "[Student roll is :"+roll+", name is :"+name+", address is :"+addr+"]";

  }

}

package com.ravi.static_method_demo1;

import java.util.Scanner;

public class StudentDetails {
```

```
	public static void main(String[] args)
	{
		Scanner sc = new Scanner(System.in);
		System.out.print("Enter Student Roll :");
		int roll = sc.nextInt();

		System.out.print("Enter Student Name :");
		String name = sc.nextLine();
		name = sc.nextLine();

		System.out.print("Enter Student Address :");
		String addr = sc.nextLine();

		String details = Student.getStudentDetails(roll, name, addr);
		System.out.println(details);

		sc.close();
	}

}
```

//Program to work with boolean as a return type :

```
package com.ravi.static_method_demo1;

public class EvenOrOdd
{
  public static boolean isEven(int number)
  {
	return number%2 ==0;
  }
}
```

```
package com.ravi.static_method_demo1;

import java.util.Scanner;

public class VerifyEvenOROddNumber {

	public static void main(String[] args)
	{
		Scanner sc = new Scanner(System.in);
		System.out.print("Enter a Number :");
```

```
		int num = sc.nextInt();

		System.out.println("Is "+num+" Even number ? "+EvenOrOdd.isEven(num));
		sc.close();
	}

}
```

//Program which will return character

```
package com.ravi.static_method_demo1;

public class FindGender
{
  public static char getGender(String gender)
  {
	return gender.charAt(0);
  }
}


package com.ravi.static_method_demo1;

import java.util.Scanner;

public class ChracterReturnType {

	public static void main(String[] args)
	{
		Scanner sc = new Scanner(System.in);
		System.out.println("Enter your Gender [Male/Female] :");
		String gender = sc.next();

		char gen = FindGender.getGender(gender);

		System.out.println("Your Gender is :"+gen);
		sc.close();

	}

}
```

==========

10-02-2025

Types of Variables in java :

In java based on the data type we have only 2 types of variables :

1) Primitive Variables
2) Reference Variables

Primitive Variables :

If any variable in java declared with primitive data type i.e
byte, short, int, long, float, double char and boolean then it is called Primitive Variable.

Example :

```
int x = 10;
```

On primitive variable we can't assign null literal as well as with primitive variable we can't invoke a method.

```
int x = null; //Invalid

int y = 23;
y.m1();  //Invalid
```

Reference Variable :

If we declare any variable with reference data type i.e with class name, interface name and so on then it is called Reference variable.

Example :

```
Scanner sc = new Scanner(System.in);  //sc is a reference variable
Student s;  //s is a reference variable
```

On reference variable we can assign null literal as well as we can invoke any method on reference variable.

```
Employee e1 = null;  //Valid

Scanner sc = new Scanner(System.in);
```

```
sc.nextInt();//valid
```

Now, Based on the declaration position and modifier, Variables are further divided into four types :

1) Class Level Variables :

a) Class Variable OR Static Field
b) Instance Variable OR Non static Field

2) Method Level Variables :

c) Local Variables
d) Parameter Variables

```
//Program on Primitive Variables

package com.ravi.variables;

class Test
{
	static int a = 100;  //Static Field
	int b = 200;	//non static Field

	public void accept(int c)  //Parameter variable
	{
		int d = 400;	//Local Variable

		System.out.println("Static Field :"+Test.a);

		Test t1 = new Test();
		System.out.println("Non static Field :"+t1.b);

		System.out.println("Parameter Variable :"+c);
		System.out.println("Local Variable :"+d);
	}
}

public class PrimitiveVariables
{
	public static void main(String[] args)
```

```
	{
		Test t1 = new Test();
		t1.accept(300);

	}
}
```

Note : In the above program the variable a,b,c and d all are primitive variables.

Program on Reference Variables :

```
package com.ravi.variables;

import java.util.Scanner;

class Demo
{
  Integer i1 = 100; //Non static Field
  static Scanner sc = new Scanner(System.in); //static Field

  public void accept(Integer i2) //i2 parameter Variable
  {
	Scanner scanner = new Scanner(System.in); //scanner is a local var.

  }
}
public class ReferenceVariables {

	public static void main(String[] args)
	{

	}
}
```
