Assignment 1: Implement a Simple Calculator
Create a functional interface called SimpleCalculator with a method calculate that takes two
integers and returns an integer. Implement this interface using lambda expressions for
addition, subtraction, multiplication, and division.

sol:

```
interface SimpleCalculator {
    int calculate(int i, int j);
}
public class Test {
    public static void main(String[] args) {

        SimpleCalculator addition = (a, b) -> a + b;

        SimpleCalculator subtraction = (a, b) -> a - b;

        SimpleCalculator multiplication = (a, b) -> a * b;

        SimpleCalculator division = (a, b) -> {
            if (b == 0) {
                throw new ArithmeticException("Division by zero is not
allowed.");
            }
            return a / b;
        };

        System.out.println("Addition of two Intergers       : " +
addition.calculate(10, 5));
        System.out.println("Subtraction of two Intergers    : " +
subtraction.calculate(10, 5));
        System.out.println("Multiplication of two Intergers : " +
multiplication.calculate(10, 5));
        System.out.println("Division of two Intergers       : " +
division.calculate(10, 5));
    }
}
```

Assignment 2: String Transformation
Implement a functional interface called StringTransformer that takes a single string and
returns a transformed string. Create lambda expressions for converting a string to uppercase,
reversing a string, and finding the length of a string.

sol:

```
interface StringTransformer {
```

```
    String transform(String input);
}
public class Test {
    public static void main(String[] args) {
        StringTransformer st = input -> input.toUpperCase();

        StringTransformer st1 = input -> new
StringBuilder(input).reverse().toString();

        StringTransformer st2 = input -> String.valueOf(input.length());

        String originalString = "Manikanta";

        System.out.println("Given String        : " + originalString);
        System.out.println("String Uppercase    : "
+st.transform(originalString));
        System.out.println("String Reversed     : " +
st1.transform(originalString));
        System.out.println("String Length       : " +
st2.transform(originalString));
    }
}
```

Assignment 3: Simple Predicate Example
Implement a functional interface called NumberPredicate to check if a number satisfies
certain conditions. Use lambda expressions for checking if a number is positive, negative, and
even.

sol:

```
interface NumberPredicate {
    boolean test(int number);
}
public class Test {
    public static void main(String[] args) {

        NumberPredicate Positive = number -> number > 0;

        NumberPredicate Negative = number -> number < 0;

        NumberPredicate Even = number -> number % 2 == 0;


        int[] numbers = {10, -5, 0, 15, -8};

        for (int number : numbers) {
            System.out.println("Number: " + number);
            System.out.println("Number is Positive :" + Positive.test(number));
            System.out.println("Number is Negative :" + Negative.test(number));
            System.out.println("Number is Even     :"     + Even.test(number));
```

```
            System.out.println("---------------------------");
        }
    }
}
```

Assignment 4: Temperature Converter
Implement a functional interface called TemperatureConverter that converts temperatures
between Celsius and Fahrenheit using lambda expressions.
sol:

```
interface TemperatureConverter {
    double convert(double temperature);
}
public class Test {
    public static void main(String[] args) {

        TemperatureConverter cf = celsius -> (celsius * 9/5) + 32;

        TemperatureConverter fc = fahrenheit -> (fahrenheit - 32) * 5/9;

        double tempInCelsius = 25.0;
        double tempInFahrenheit = 77.0;

        System.out.println(tempInCelsius + " °C is " + cf.convert(tempInCelsius)
+ " °F");
        System.out.println(tempInFahrenheit + " °F is " + fc
.convert(tempInFahrenheit) + " °C");
    }
}
```