ASSIGNMENT NO.2

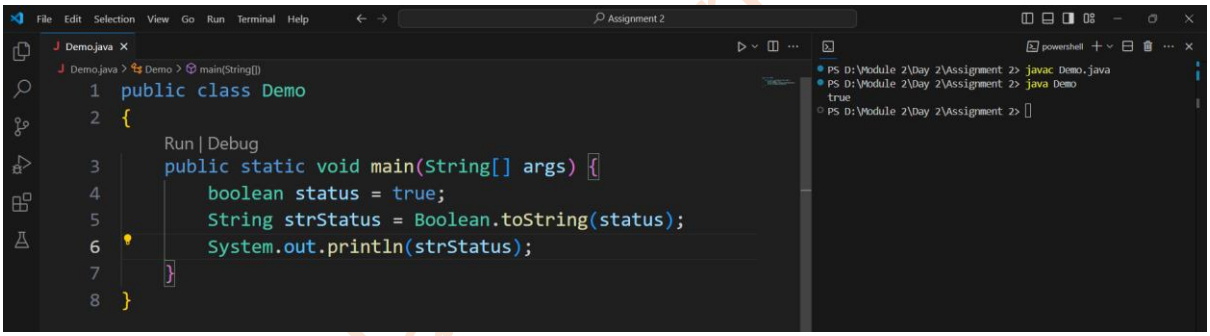
Note: Consider the following before starting the assignment:

- A **static field** declared inside a class is called a **class-level variable**. To access this variable, use the class name and the dot operator (e.g., `Integer.MAX_VALUE`).
- A **static method** defined inside a class is called a **class-level method**. To access this method, use the class name and the dot operator (e.g., `Integer.parseInt()`).
- When accessing static members within the same class, you do not need to use the class name.

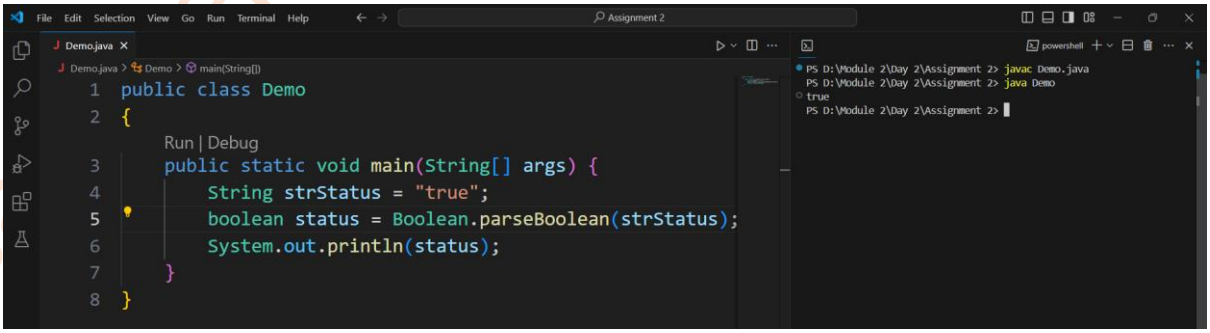
1. Working with `java.lang.Boolean`

a. Explore the Java API documentation for `java.lang.Boolean` and observe its modifiers and super types.

b. Declare a method-local variable `status` of type `boolean` with the value `true` and convert it to a `String` using the `toString` method. (Hint: Use `Boolean.toString(Boolean)`).



c. Declare a method-local variable `strStatus` of type `String` with the value `"true"` and convert it to a `boolean` using the `parseBoolean` method. (Hint: Use `Boolean.parseBoolean(String)`).



d. Declare a method-local variable `strStatus` of type `String` with the value `"1"` or `"0"` and attempt to convert it to a `boolean`. (Hint: `parseBoolean` method will not work as expected with `"1"` or `"0"`).

ASSIGNMENT NO.2

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        String strStatus1 = "1";
        String strStatus0 = "0";
        boolean status1 = Boolean.parseBoolean(strStatus1);
        boolean status0 = Boolean.parseBoolean(strStatus0);
        System.out.println(status1);
        System.out.println(status0);

    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
false
false
PS D:\Module 2\Day 2\Assignment 2>
```

e. Declare a method-local variable `status` of type `boolean` with the value `true` and convert it to the corresponding wrapper class using `Boolean.valueOf()`. (Hint: Use `Boolean.valueOf(boolean)`).

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        boolean status = true;
        boolean boolStatus = Boolean.valueOf(status);
        System.out.println(boolStatus);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
true
PS D:\Module 2\Day 2\Assignment 2>
```

f. Declare a method-local variable `strStatus` of type `String` with the value `"true"` and convert it to the corresponding wrapper class using `Boolean.valueOf()`. (Hint: Use `Boolean.valueOf(String)`).

```java
ublic class Demo

    Run | Debug
    public static void main(String[] args) {
        String strStatus = "true";
        boolean boolStatus = Boolean.valueOf(strStatus);
        System.out.println(boolStatus);
    }

```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
true
PS D:\Module 2\Day 2\Assignment 2>
```

g. Experiment with converting a `boolean` value into other primitive types or vice versa and observe the results.

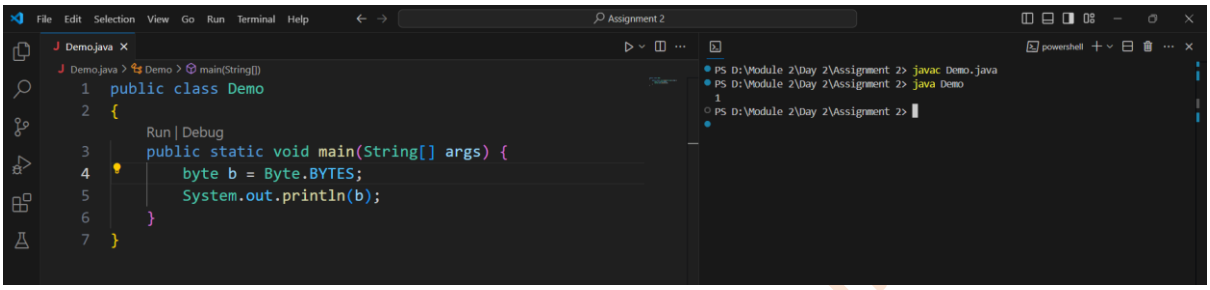
```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        boolean boolStatus = true;
        int num = boolStatus ? 1 : 0;
        System.out.println(num);

        boolean Status = false;
        int n = Status ? 1 : 0;
        System.out.println(n);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
1
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
1
0
PS D:\Module 2\Day 2\Assignment 2>
```

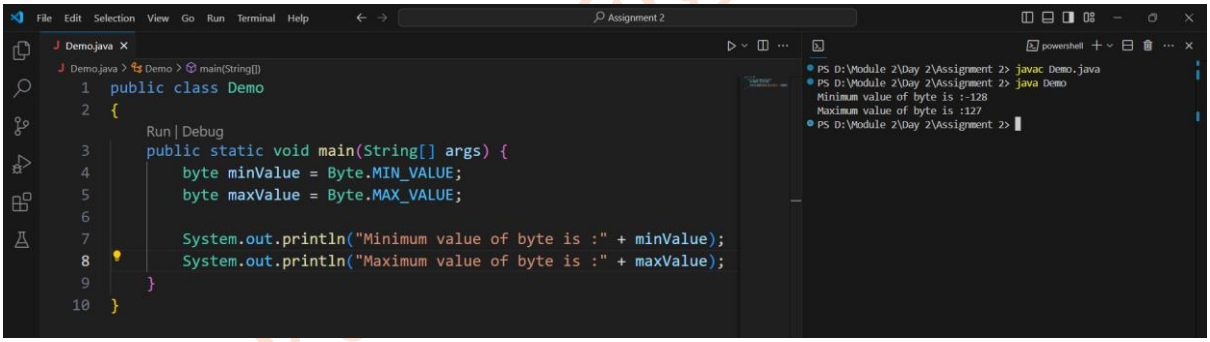
2. Working with `java.lang.Byte`

a. Explore the Java API documentation for `java.lang.Byte` and observe its modifiers and super types.

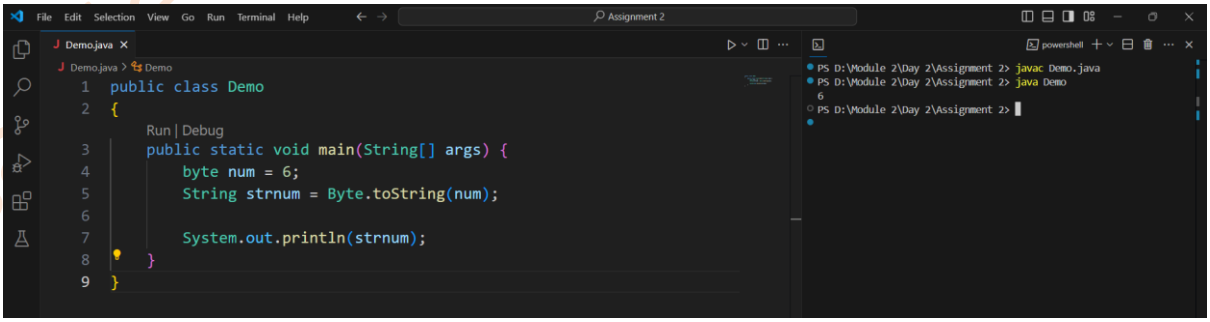
b. Write a program to test how many bytes are used to represent a `byte` value using the `BYTES` field. (Hint: Use `Byte.BYTES`).



c. Write a program to find the minimum and maximum values of `byte` using the `MIN_VALUE` and `MAX_VALUE` fields. (Hint: Use `Byte.MIN_VALUE` and `Byte.MAX_VALUE`).



d. Declare a method-local variable `number` of type `byte` with some value and convert it to a `String` using the `toString` method. (Hint: Use `Byte.toString(byte)`).



e. Declare a method-local variable `strNumber` of type `String` with some value and convert it to a `byte` value using the `parseByte` method. (Hint: Use `Byte.parseByte(String)`).

ASSIGNMENT NO.2

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        String strnum = "50";
        byte b = Byte.parseByte(strnum);

        System.out.println(b);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
50
PS D:\Module 2\Day 2\Assignment 2>
```

f. Declare a method-local variable `strNumber` of type `String` with the value `"Ab12Cd3"` and attempt to convert it to a `byte` value. (Hint: `parseByte` method will throw a `NumberFormatException`).

g. Declare a method-local variable `number` of type `byte` with some value and convert it to the corresponding wrapper class using `Byte.valueOf()`. (Hint: Use `Byte.valueOf(byte)`).

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        byte number = 3;
        byte n = Byte.valueOf(number);
        System.out.println(n);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
3
PS D:\Module 2\Day 2\Assignment 2>
```

h. Declare a method-local variable `strNumber` of type `String` with some `byte` value and convert it to the corresponding wrapper class using `Byte.valueOf()`. (Hint: Use `Byte.valueOf(String)`).

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        String strNumber = "100";
        Byte bytNum = Byte.valueOf(strNumber);
        System.out.println(bytNum);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
100
PS D:\Module 2\Day 2\Assignment 2>
```

i. Experiment with converting a `byte` value into other primitive types or vice versa and observe the results.

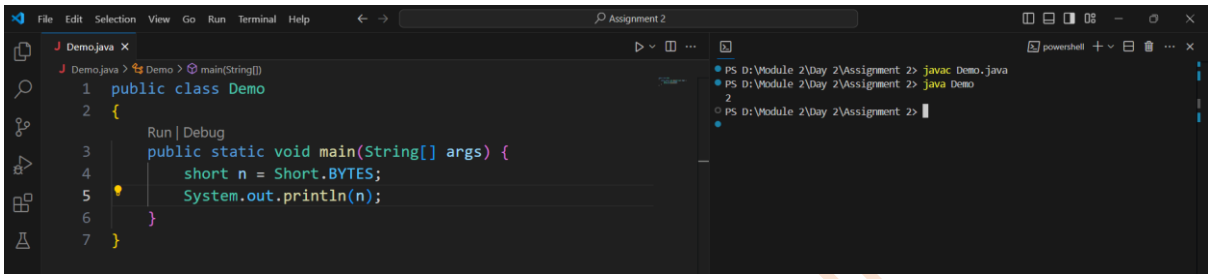
```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        byte num = 25;
        int n = Integer.valueOf(num);
        double d = Double.valueOf(num);

        System.out.println(n);
        System.out.println(d);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
25
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
25
25.0
PS D:\Module 2\Day 2\Assignment 2>
```

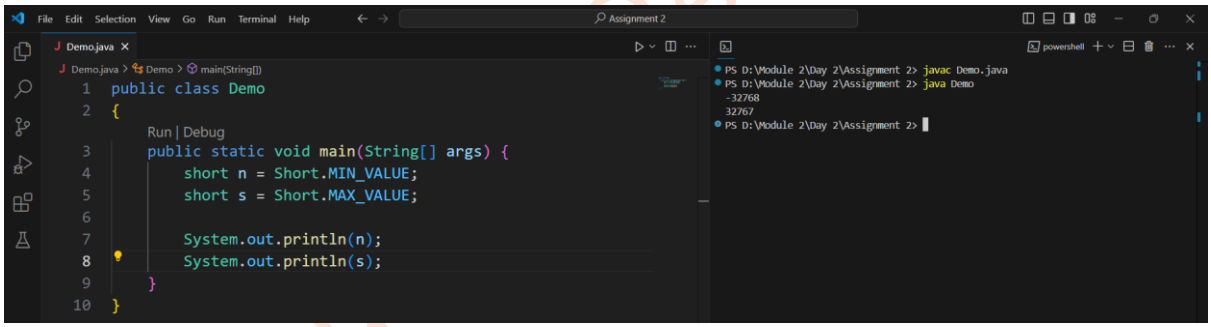
3. Working with `java.lang.Short`

a. Explore the Java API documentation for `java.lang.Short` and observe its modifiers and super types.

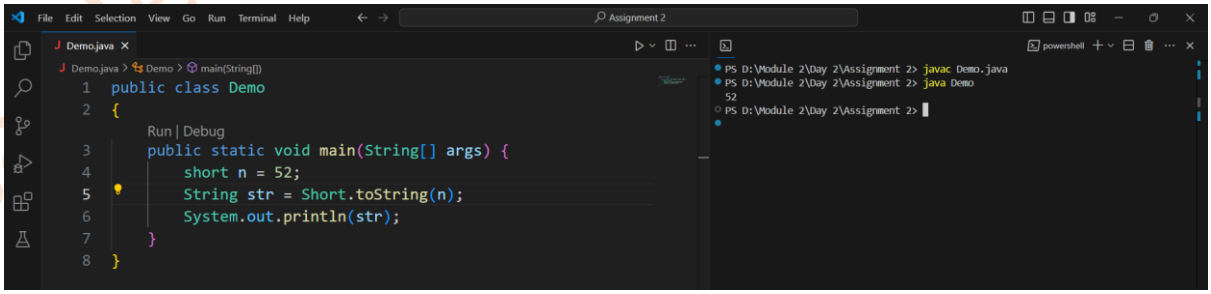
b. Write a program to test how many bytes are used to represent a `short` value using the `BYTES` field. (Hint: Use `Short.BYTES`).



c. Write a program to find the minimum and maximum values of `short` using the `MIN_VALUE` and `MAX_VALUE` fields. (Hint: Use `Short.MIN_VALUE` and `Short.MAX_VALUE`).



d. Declare a method-local variable `number` of type `short` with some value and convert it to a `String` using the `toString` method. (Hint: Use `Short.toString(short)`).



e. Declare a method-local variable `strNumber` of type `String` with some value and convert it to a `short` value using the `parseShort` method. (Hint: Use `Short.parseShort(String)`).

ASSIGNMENT NO.2

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        String str = "50";
        short s = Short.parseShort(str);
        System.out.println(s);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
50
PS D:\Module 2\Day 2\Assignment 2>
```

f. Declare a method-local variable `strNumber` of type `String` with the value `"Ab12Cd3"` and attempt to convert it to a `short` value. (Hint: `parseShort` method will throw a `NumberFormatException`).

g. Declare a method-local variable `number` of type `short` with some value and convert it to the corresponding wrapper class using `Short.valueOf()`. (Hint: Use `Short.valueOf(short)`).

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        short st = 20;
        short s = Short.valueOf(st);
        System.out.println(s);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
20
PS D:\Module 2\Day 2\Assignment 2>
```

h. Declare a method-local variable `strNumber` of type `String` with some `short` value and convert it to the corresponding wrapper class using `Short.valueOf()`. (Hint: Use `Short.valueOf(String)`).

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        String st = "20000";
        short s = Short.valueOf(st);
        System.out.println(s);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
20000
PS D:\Module 2\Day 2\Assignment 2>
```

i. Experiment with converting a `short` value into other primitive types or vice versa and observe the results.

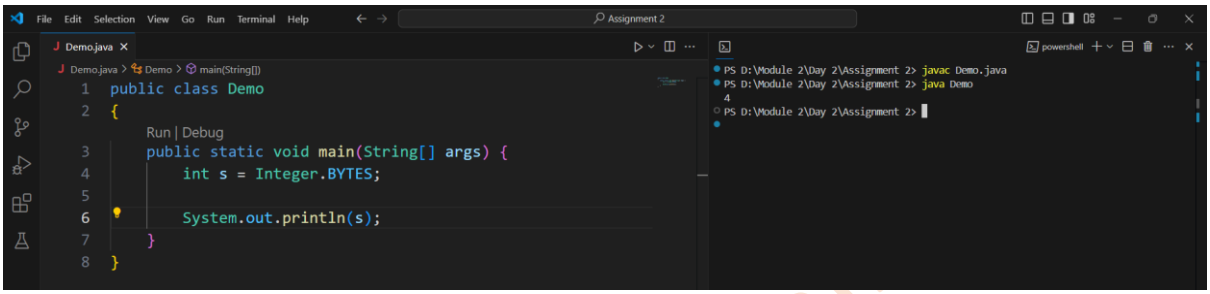
```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        short s = 3000;
        int n = Short.valueOf(s);
        double d = Short.valueOf(s);
        System.out.println(n);
        System.out.println(d);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
3000
3000.0
PS D:\Module 2\Day 2\Assignment 2>
```

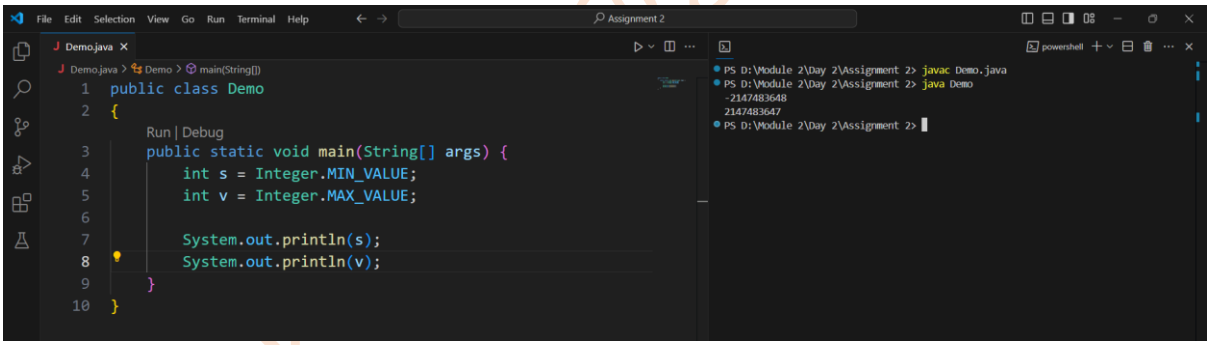
4. Working with `java.lang.Integer`

a. Explore the Java API documentation for `java.lang.Integer` and observe its modifiers and super types.

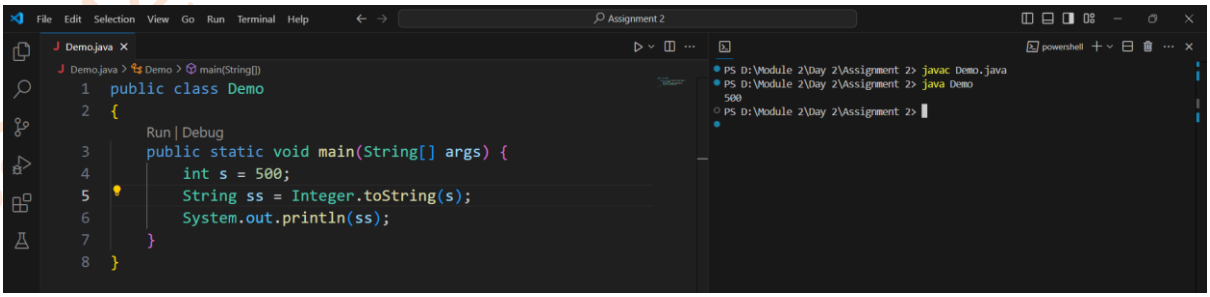
b. Write a program to test how many bytes are used to represent an `int` value using the `BYTES` field. (Hint: Use `Integer.BYTES`).



c. Write a program to find the minimum and maximum values of `int` using the `MIN_VALUE` and `MAX_VALUE` fields. (Hint: Use `Integer.MIN_VALUE` and `Integer.MAX_VALUE`).



d. Declare a method-local variable `number` of type `int` with some value and convert it to a `String` using the `toString` method. (Hint: Use `Integer.toString(int)`).



e. Declare a method-local variable `strNumber` of type `String` with some value and convert it to an `int` value using the `parseInt` method. (Hint: Use `Integer.parseInt(String)`).

ASSIGNMENT NO.2

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        String num = "5200";
        int ss = Integer.parseInt(num);
        System.out.println(ss);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
5200
PS D:\Module 2\Day 2\Assignment 2>
```

f. Declare a method-local variable `strNumber` of type `String` with the value `"Ab12Cd3"` and attempt to convert it to an `int` value. (Hint: `parseInt` method will throw a `NumberFormatException`).

g. Declare a method-local variable `number` of type `int` with some value and convert it to the corresponding wrapper class using `Integer.valueOf()`. (Hint: Use `Integer.valueOf(int)`).

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        int num = 10000;
        int ss = Integer.valueOf(num);
        System.out.println(ss);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
10000
PS D:\Module 2\Day 2\Assignment 2>
```

h. Declare a method-local variable `strNumber` of type `String` with some integer value and convert it to the corresponding wrapper class using `Integer.valueOf()`. (Hint: Use `Integer.valueOf(String)`).

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        String num = "10000";
        int ss = Integer.valueOf(num);
        System.out.println(ss);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
10000
PS D:\Module 2\Day 2\Assignment 2>
```

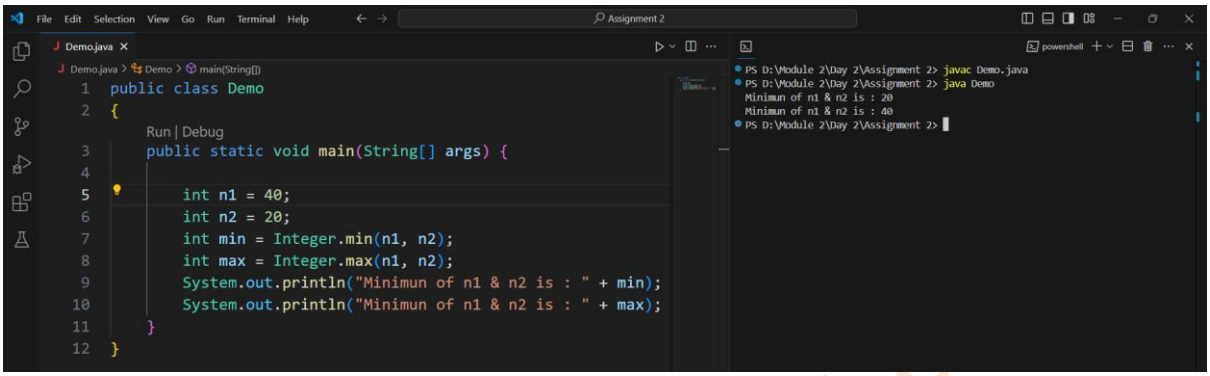
i. Declare two integer variables with values `10` and `20`, and add them using a method from the `Integer` class. (Hint: Use `Integer.sum(int, int)`).

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {

        int n1 = 10;
        int n2 = 20;
        int sum = Integer.sum(n1, n2);
        System.out.println("Sum of n1 & n2 is : " + sum);
    }
}
```

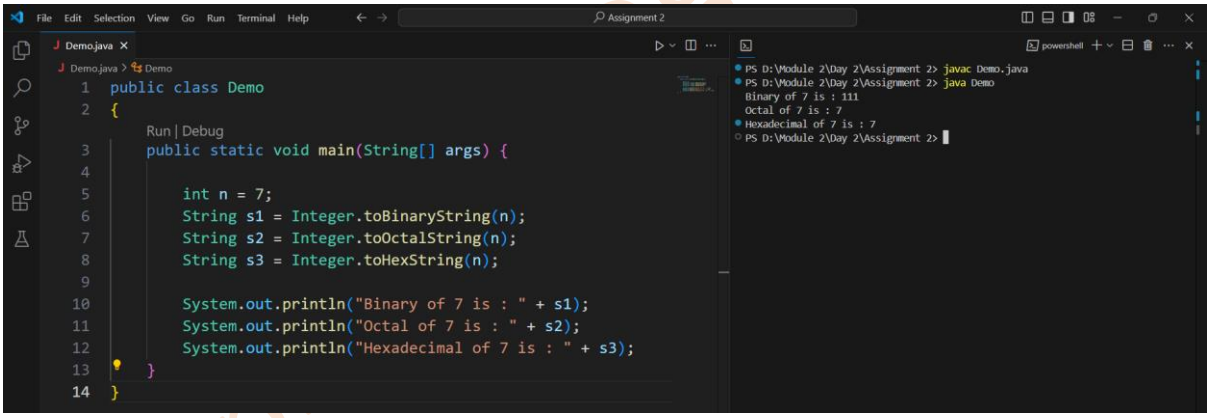
```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
Sum of n1 & n2 is : 30
PS D:\Module 2\Day 2\Assignment 2>
```

ASSIGNMENT NO.2

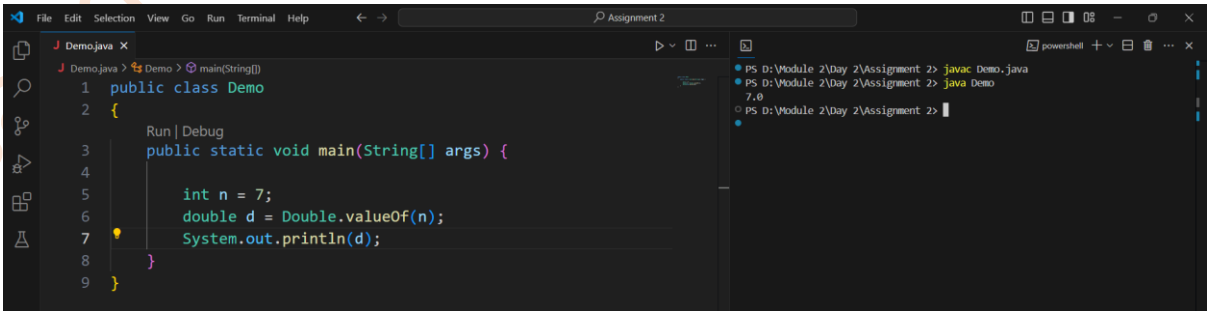
j. Declare two integer variables with values `10` and `20`, and find the minimum and maximum values using the `Integer` class. (Hint: Use `Integer.min(int, int)` and `Integer.max(int, int)`).



k. Declare an integer variable with the value `7`. Convert it to binary, octal, and hexadecimal strings using methods from the `Integer` class. (Hint: Use `Integer.toBinaryString(int)`, `Integer.toOctalString(int)`, and `Integer.toHexString(int)`).



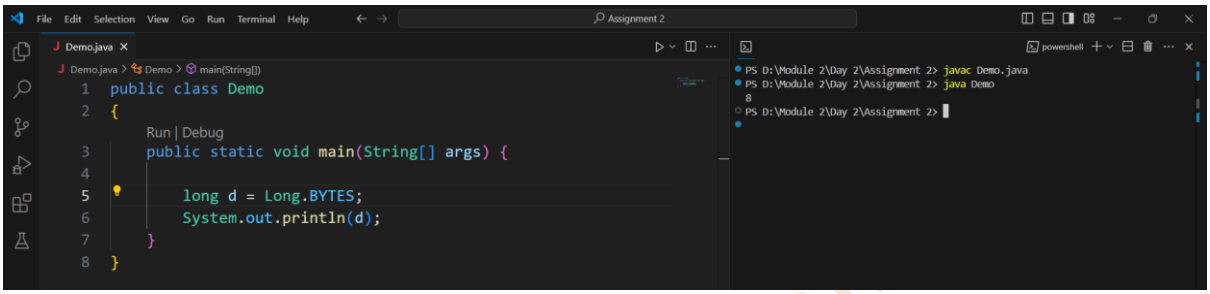
l. Experiment with converting an `int` value into other primitive types or vice versa and observe the results.



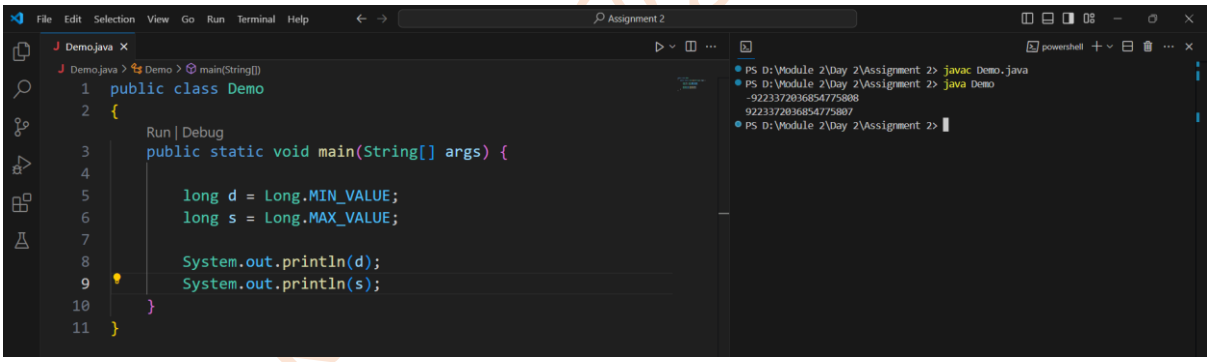
5. Working with `java.lang.Long`

a. Explore the Java API documentation for `java.lang.Long` and observe its modifiers and super types.

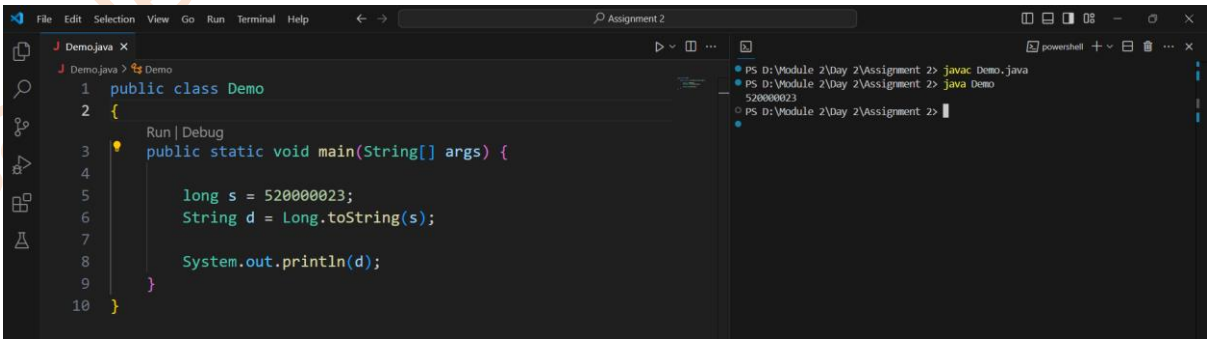
b. Write a program to test how many bytes are used to represent a `long` value using the `BYTES` field. (Hint: Use `Long.BYTES`).



c. Write a program to find the minimum and maximum values of `long` using the `MIN_VALUE` and `MAX_VALUE` fields. (Hint: Use `Long.MIN_VALUE` and `Long.MAX_VALUE`).

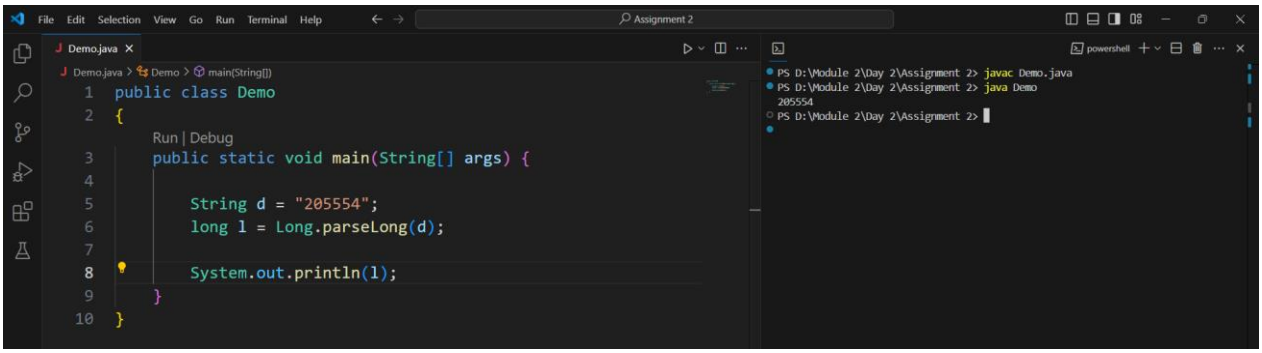


d. Declare a method-local variable `number` of type `long` with some value and convert it to a `String` using the `toString` method. (Hint: Use `Long.toString(long)`).



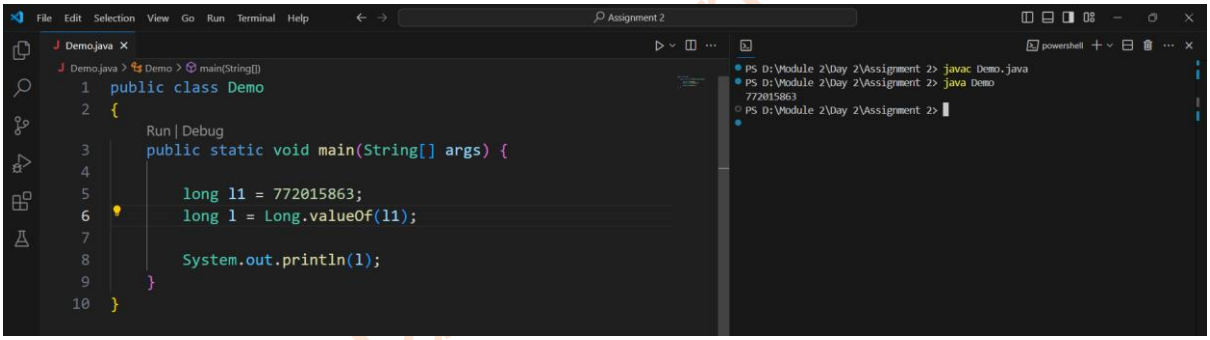
e. Declare a method-local variable `strNumber` of type `String` with some value and convert it to a `long` value using the `parseLong` method. (Hint: Use `Long.parseLong(String)`).

ASSIGNMENT NO.2

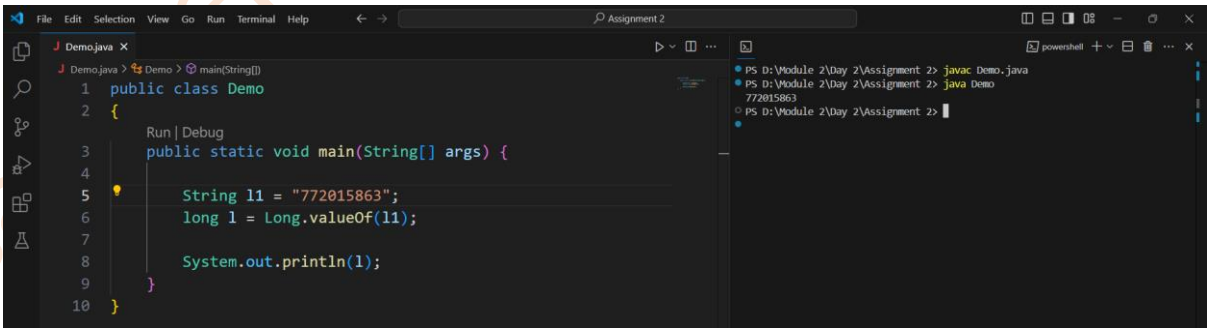


f. Declare a method-local variable `strNumber` of type `String` with the value `"Ab12Cd3"` and attempt to convert it to a `long` value. (Hint: `parseLong` method will throw a `NumberFormatException`).

g. Declare a method-local variable `number` of type `long` with some value and convert it to the corresponding wrapper class using `Long.valueOf()`. (Hint: Use `Long.valueOf(long)`).

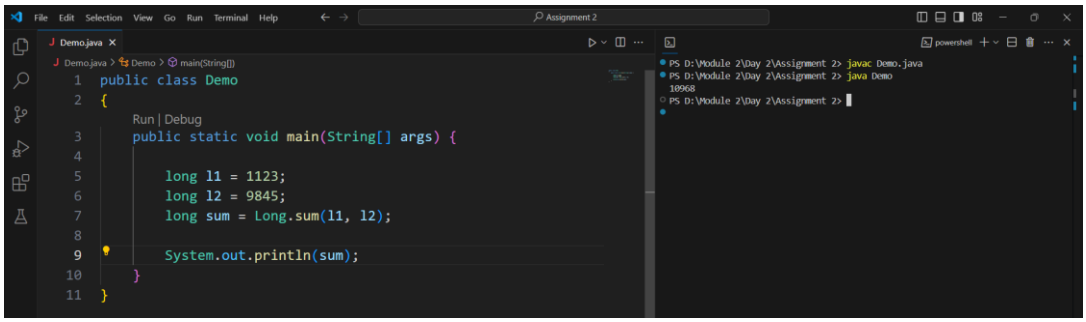


h. Declare a method-local variable `strNumber` of type `String` with some `long` value and convert it to the corresponding wrapper class using `Long.valueOf()`. (Hint: Use `Long.valueOf(String)`).

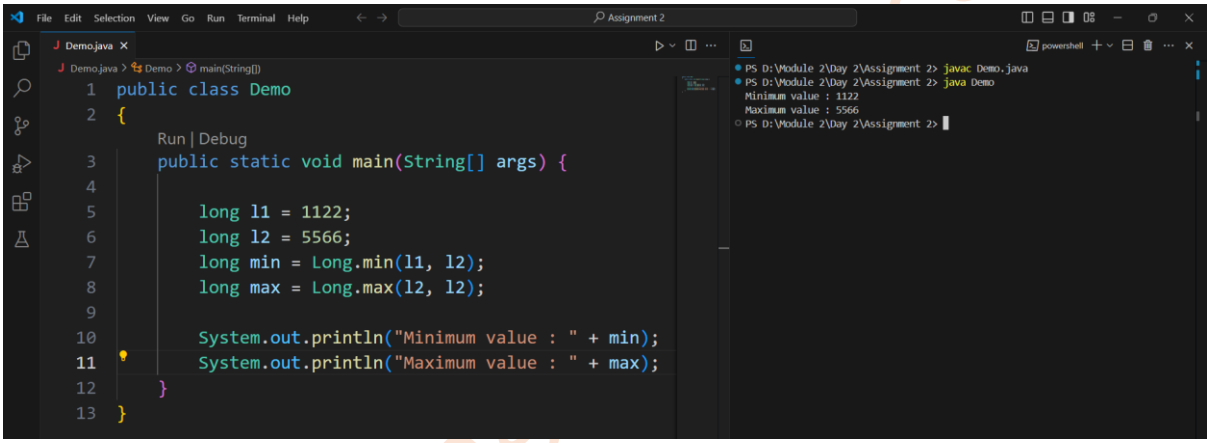


i. Declare two long variables with values `1123` and `9845`, and add them using a method from the `Long` class. (Hint: Use `Long.sum(long, long)`).

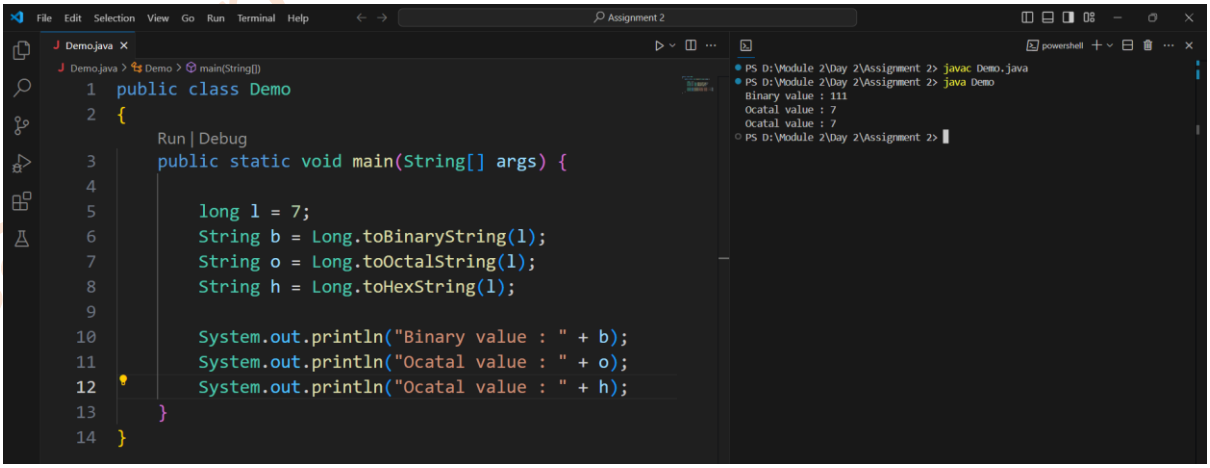
ASSIGNMENT NO.2



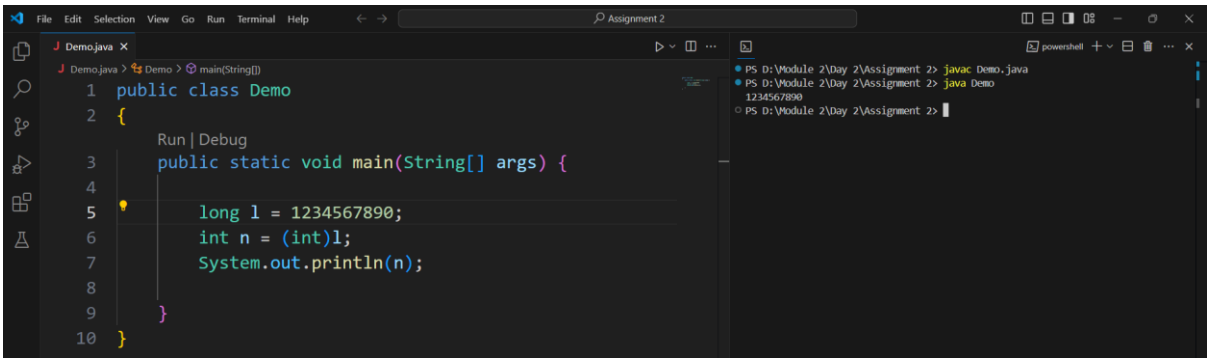
j. Declare two long variables with values `1122` and `5566`, and find the minimum and maximum values using the `Long` class. (Hint: Use `Long.min(long, long)` and `Long.max(long, long)`).



k. Declare a long variable with the value `7`. Convert it to binary, octal, and hexadecimal strings using methods from the `Long` class. (Hint: Use `Long.toBinaryString(long)`, `Long.toOctalString(long)`, and `Long.toHexString(long)`).



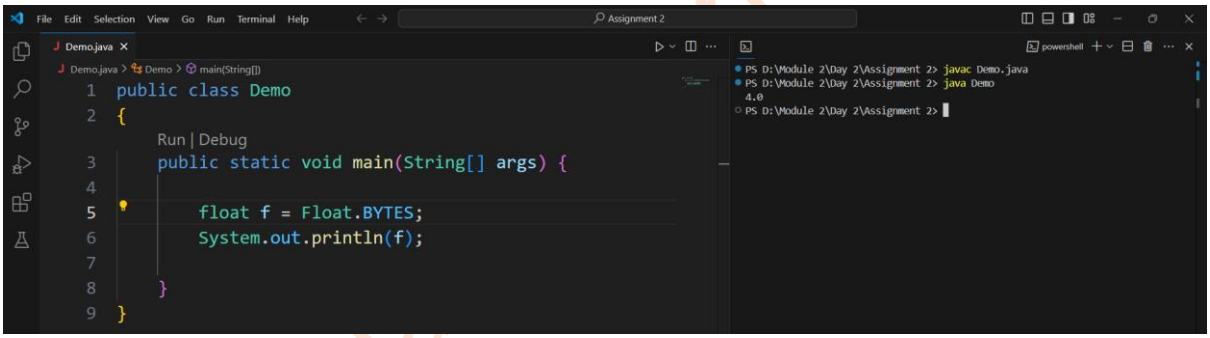
l. Experiment with converting a `long` value into other primitive types or vice versa and observe the results.



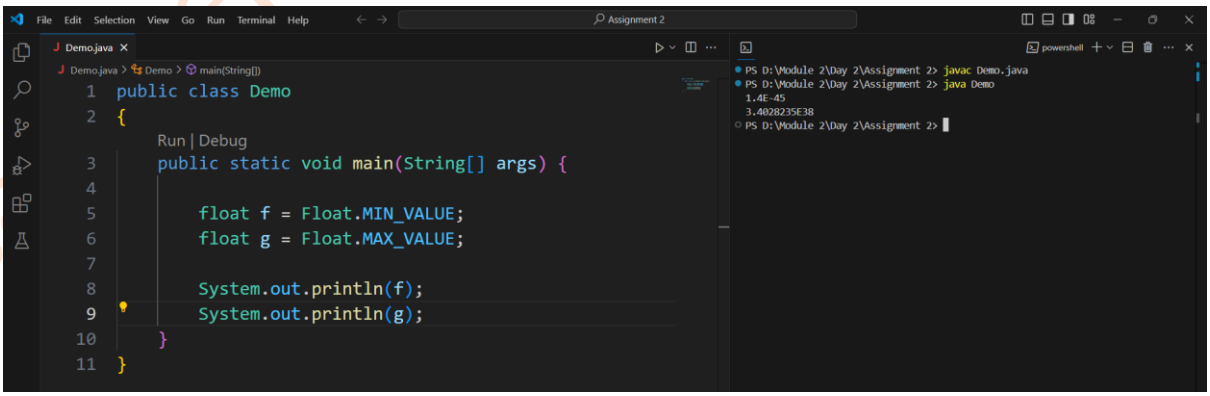
6. Working with `java.lang.Float`

a. Explore the [Java API documentation for `java.lang.Float`] and observe its modifiers and super types.

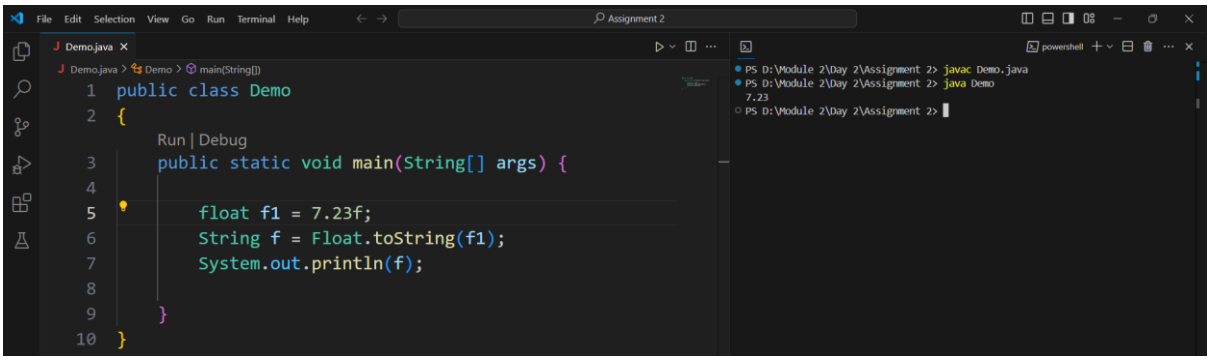
b. Write a program to test how many bytes are used to represent a `float` value using the `BYTES` field. (Hint: Use `Float.BYTES`).



c. Write a program to find the minimum and maximum values of `float` using the `MIN_VALUE` and `MAX_VALUE` fields. (Hint: Use `Float.MIN_VALUE` and `Float.MAX_VALUE`).



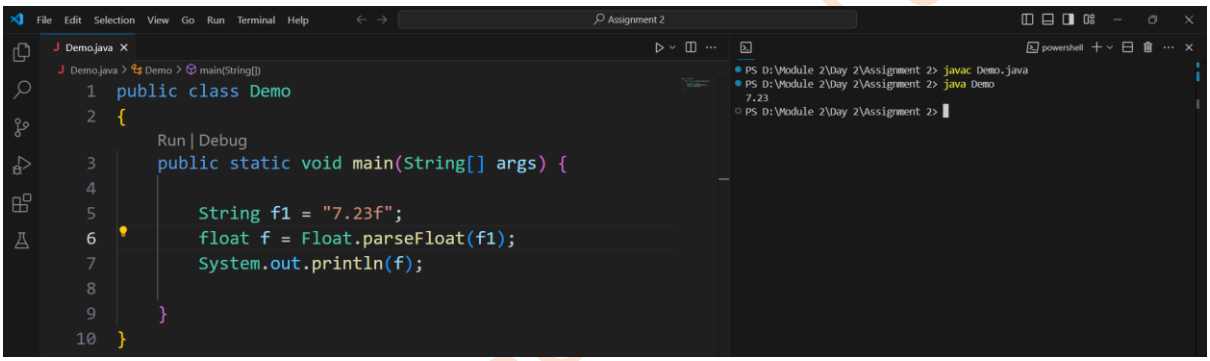
d. Declare a method-local variable `number` of type `float` with some value and convert it to a `String` using the `toString` method. (Hint: Use `Float.toString(float)`).



```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {

        float f1 = 7.23f;
        String f = Float.toString(f1);
        System.out.println(f);

    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
7.23
PS D:\Module 2\Day 2\Assignment 2>
```

e. Declare a method-local variable `strNumber` of type `String` with some value and convert it to a `float` value using the `parseFloat` method. (Hint: Use `Float.parseFloat(String)`).

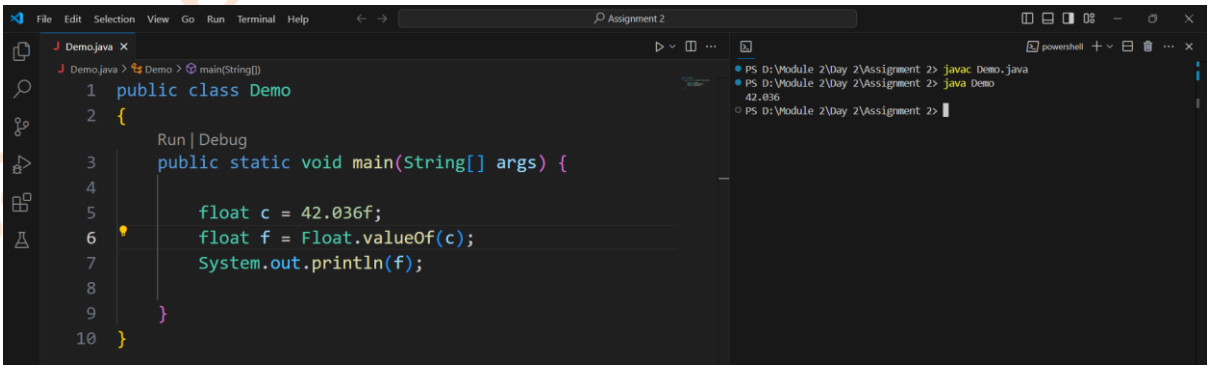


```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {

        String f1 = "7.23f";
        float f = Float.parseFloat(f1);
        System.out.println(f);

    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
7.23
PS D:\Module 2\Day 2\Assignment 2>
```

f. Declare a method-local variable `strNumber` of type `String` with the value `"Ab12Cd3"` and attempt to convert it to a `float` value. (Hint: `parseFloat` method will throw a `NumberFormatException`).

g. Declare a method-local variable `number` of type `float` with some value and convert it to the corresponding wrapper class using `Float.valueOf()`. (Hint: Use `Float.valueOf(float)`).



```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {

        float c = 42.036f;
        float f = Float.valueOf(c);
        System.out.println(f);

    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
42.036
PS D:\Module 2\Day 2\Assignment 2>
```

h. Declare a method-local variable `strNumber` of type `String` with some `float` value and convert it to the corresponding wrapper class using `Float.valueOf()`. (Hint: Use `Float.valueOf(String)`).

ASSIGNMENT NO.2

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {

        String c = "42.036f";
        float f = Float.valueOf(c);
        System.out.println(f);

    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
42.036
PS D:\Module 2\Day 2\Assignment 2>
```

i. Declare two float variables with values `112.3` and `984.5`, and add them using a method from the `Float` class. (Hint: Use `Float.sum(float, float)`).

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        float c = 112.3f;
        float d = 984.5f;
        float f = Float.sum(c, d);
        System.out.println(f);

    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
1096.8
PS D:\Module 2\Day 2\Assignment 2>
```

j. Declare two float variables with values `112.2` and `556.6`, and find the minimum and maximum values using the `Float` class. (Hint: Use `Float.min(float, float)` and `Float.max(float, float)`).

```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        float c = 112.2f;
        float d = 556.6f;
        float min = Float.min(c, d);
        float max = Float.max(c, d);
        System.out.println("Minimum : " + min);
        System.out.println("Maximum : " + max);
    }
}
```

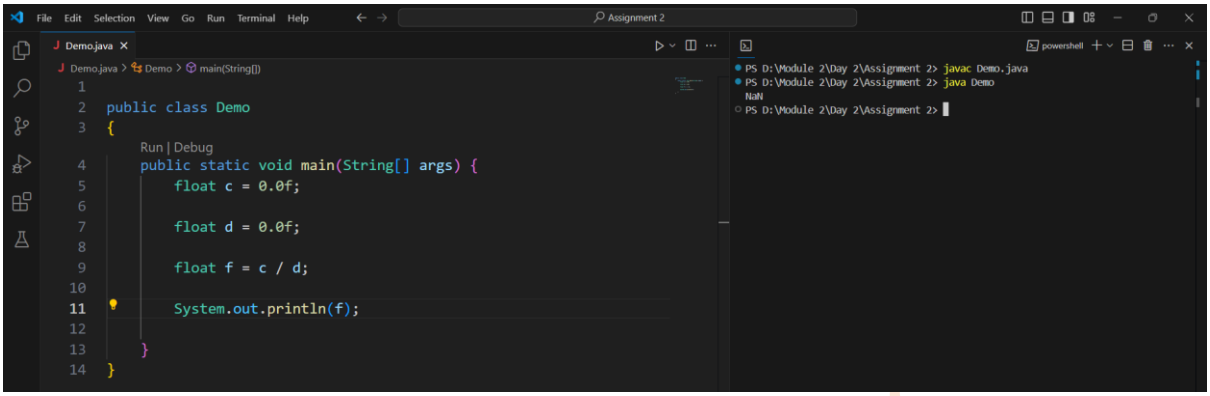
```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
Minimum : 112.2
Maximum : 556.6
PS D:\Module 2\Day 2\Assignment 2>
```

k. Declare a float variable with the value `-25.0f`. Find the square root of this value. (Hint: Use `Math.sqrt()` method).

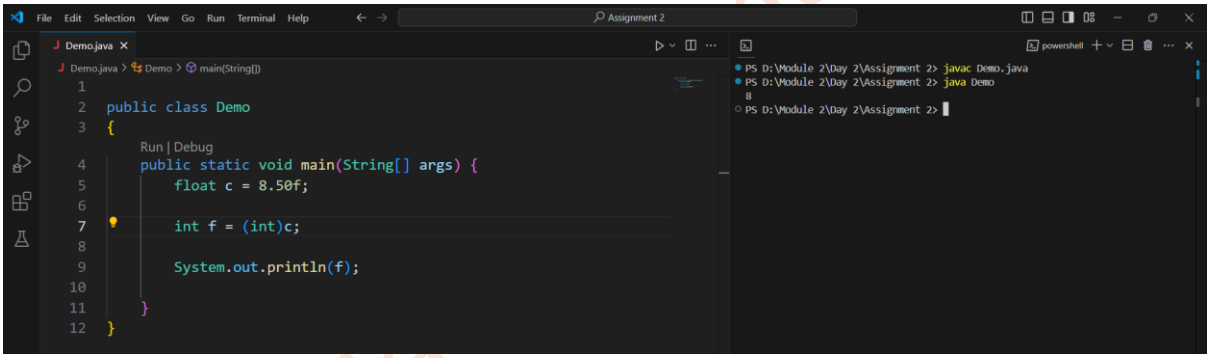
```java
public class Demo
{
    Run | Debug
    public static void main(String[] args) {
        float c = -25.0f;

        double sqrt = Math.sqrt(Math.abs(c));

        System.out.println("square root of -25.0f : " + sqrt);

    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
square root of -25.0f : 5.0
PS D:\Module 2\Day 2\Assignment 2>
```

l. Declare two float variables with the same value, `0.0f`, and divide them. (Hint: Observe the result and any special floating-point behavior).



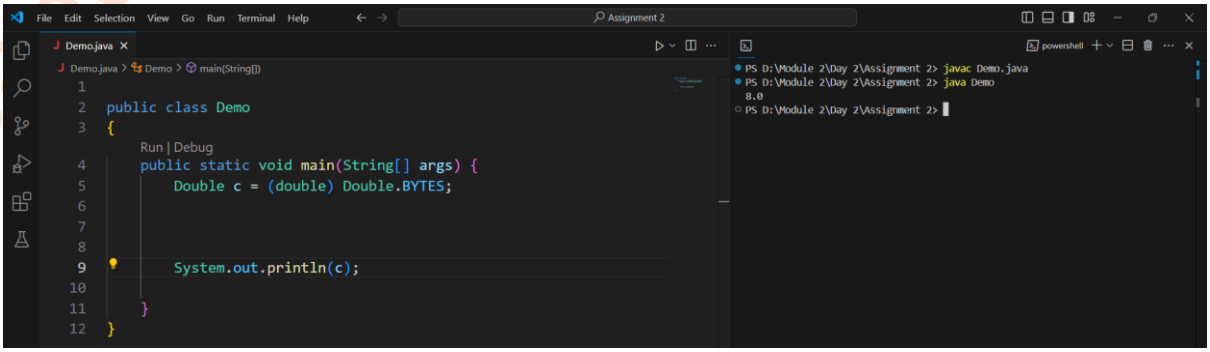
m. Experiment with converting a `float` value into other primitive types or vice versa and observe the results.



7. Working with `java.lang.Double`

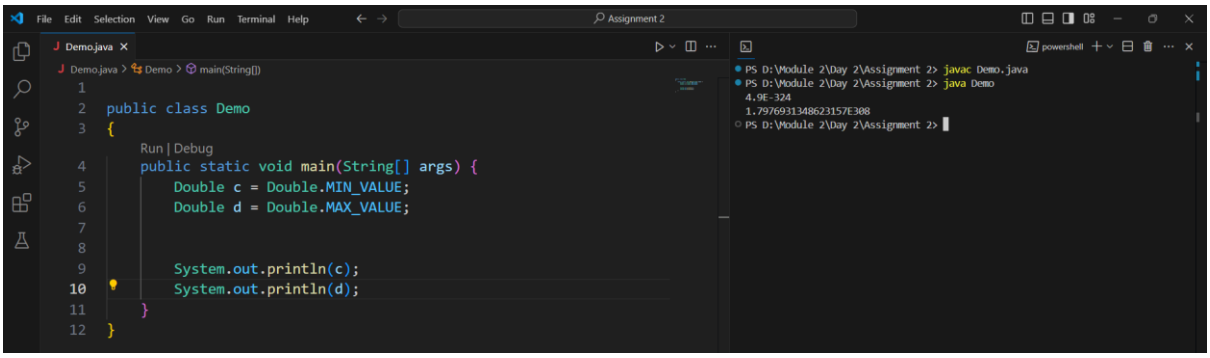
a. Explore the Java API documentation for `java.lang.Double` and observe its modifiers and super types.

b. Write a program to test how many bytes are used to represent a `double` value using the `BYTES` field. (Hint: Use `Double.BYTES`).

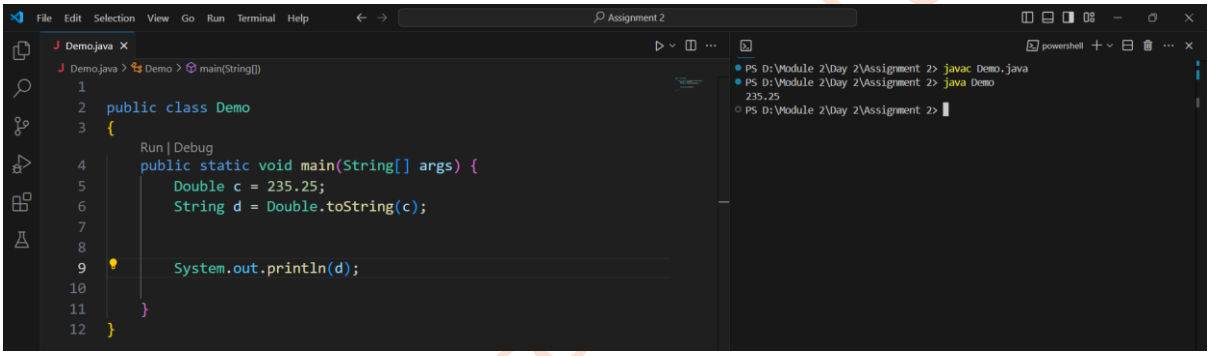


c. Write a program to find the minimum and maximum values of `double` using the `MIN_VALUE` and `MAX_VALUE` fields. (Hint: Use `Double.MIN_VALUE` and `Double.MAX_VALUE`).

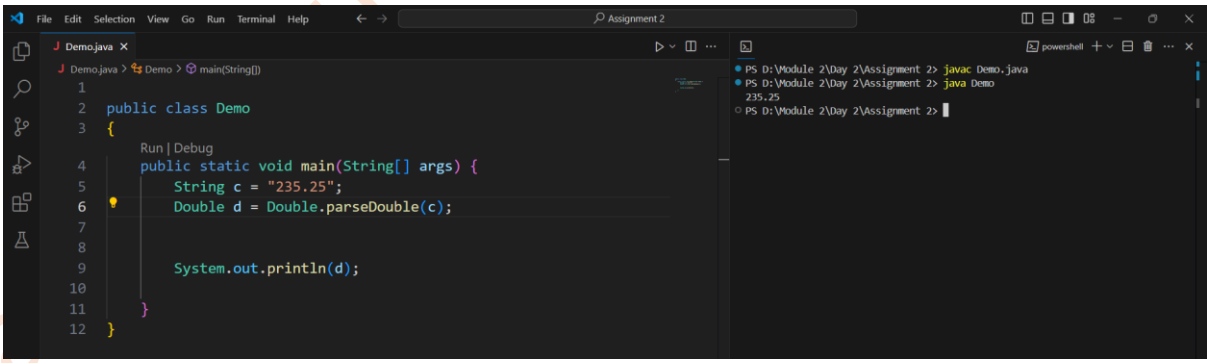
ASSIGNMENT NO.2



d. Declare a method-local variable `number` of type `double` with some value and convert it to a `String` using the `toString` method. (Hint: Use `Double.toString(double)`).



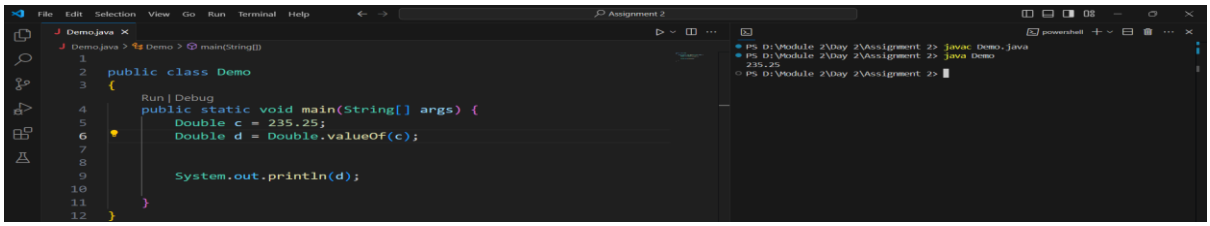
e. Declare a method-local variable `strNumber` of type `String` with some value and convert it to a `double` value using the `parseDouble` method. (Hint: Use `Double.parseDouble(String)`).



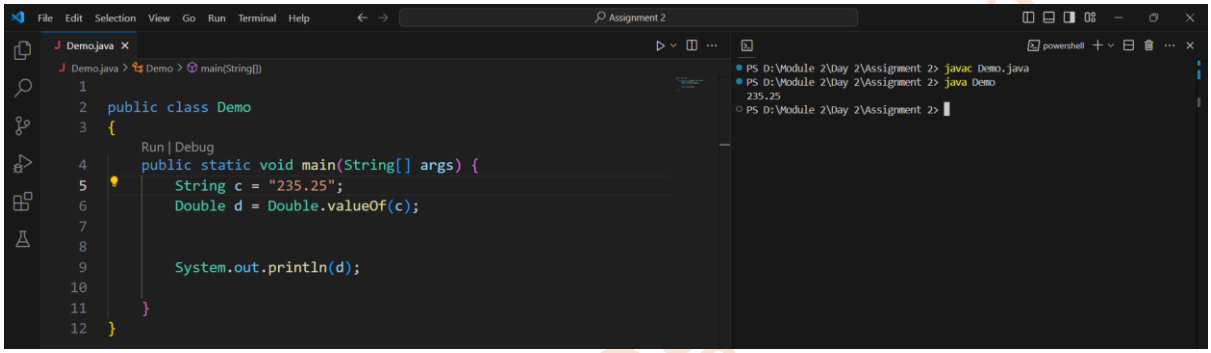
f. Declare a method-local variable `strNumber` of type `String` with the value `"Ab12Cd3"` and attempt to convert it to a `double` value. (Hint: `parseDouble` method will throw a `NumberFormatException`).

g. Declare a method-local variable `number` of type `double` with some value and convert it to the corresponding wrapper class using `Double.valueOf()`. (Hint: Use `Double.valueOf(double)`).

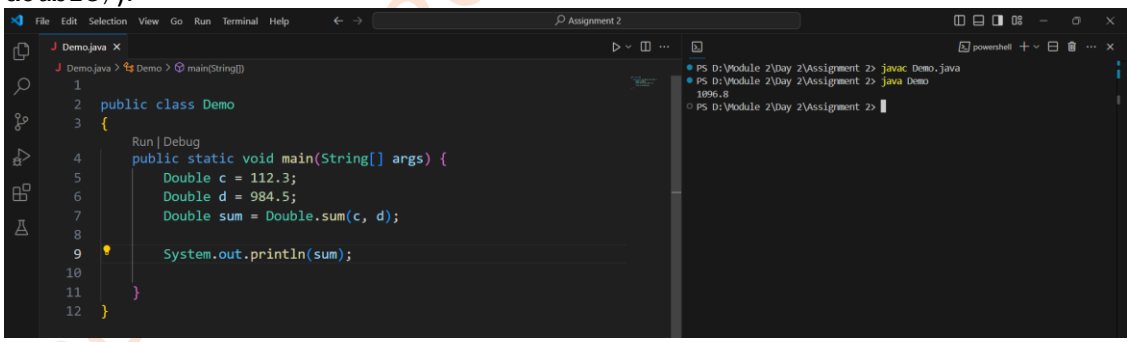
ASSIGNMENT NO.2



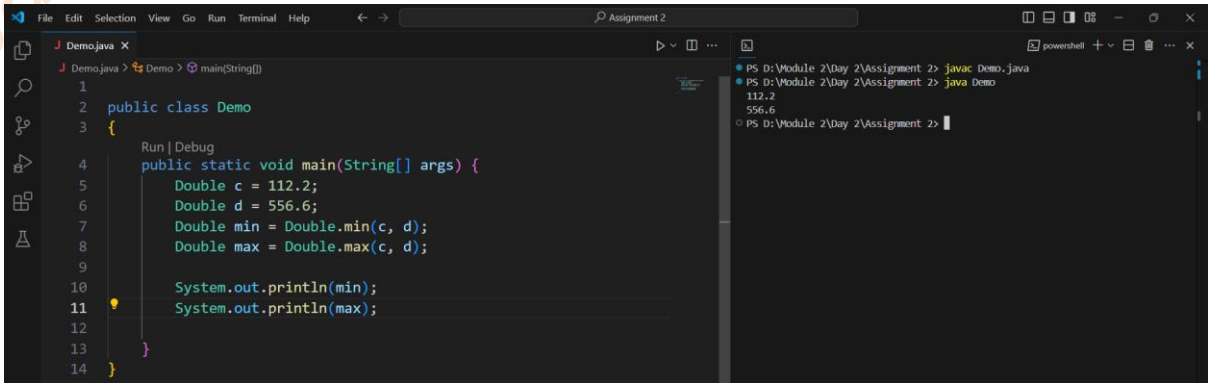
h. Declare a method-local variable `strNumber` of type `String` with some `double` value and convert it to the corresponding wrapper class using `Double.valueOf()`. (Hint: Use `Double.valueOf(String)`).



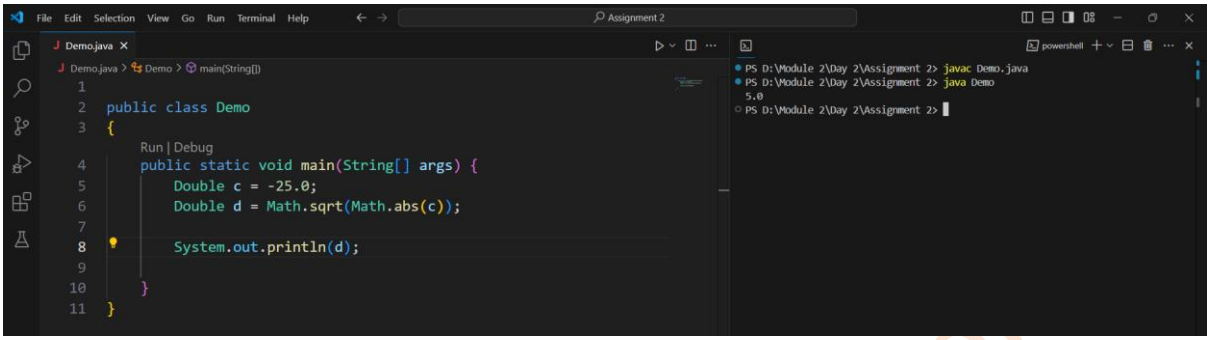
i. Declare two double variables with values `112.3` and `984.5`, and add them using a method from the `Double` class. (Hint: Use `Double.sum(double, double)`).



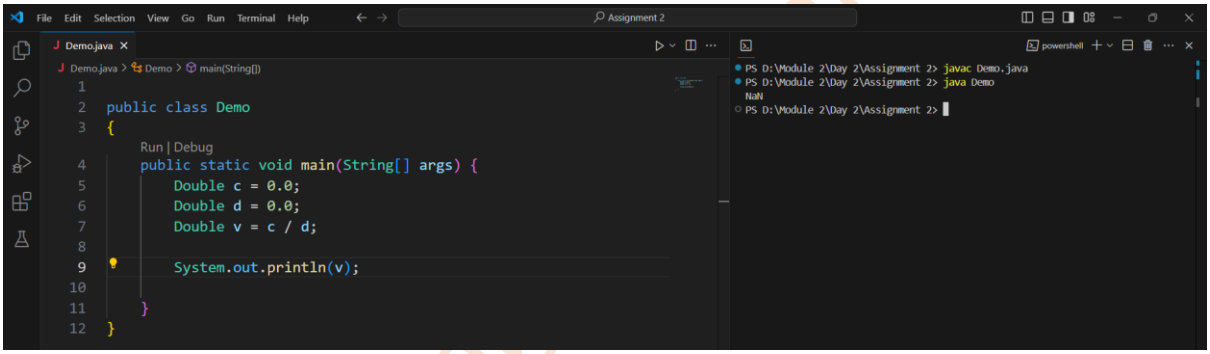
j. Declare two double variables with values `112.2` and `556.6`, and find the minimum and maximum values using the `Double` class. (Hint: Use `Double.min(double, double)` and `Double.max(double, double)`).



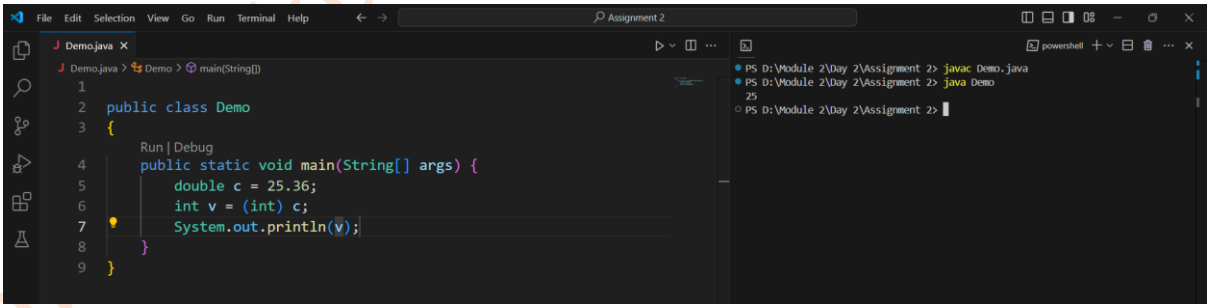
k. Declare a double variable with the value `-25.0`. Find the square root of this value. (Hint: Use `Math.sqrt()` method).



l. Declare two double variables with the same value, `0.0`, and divide them. (Hint: Observe the result and any special floating-point behavior).



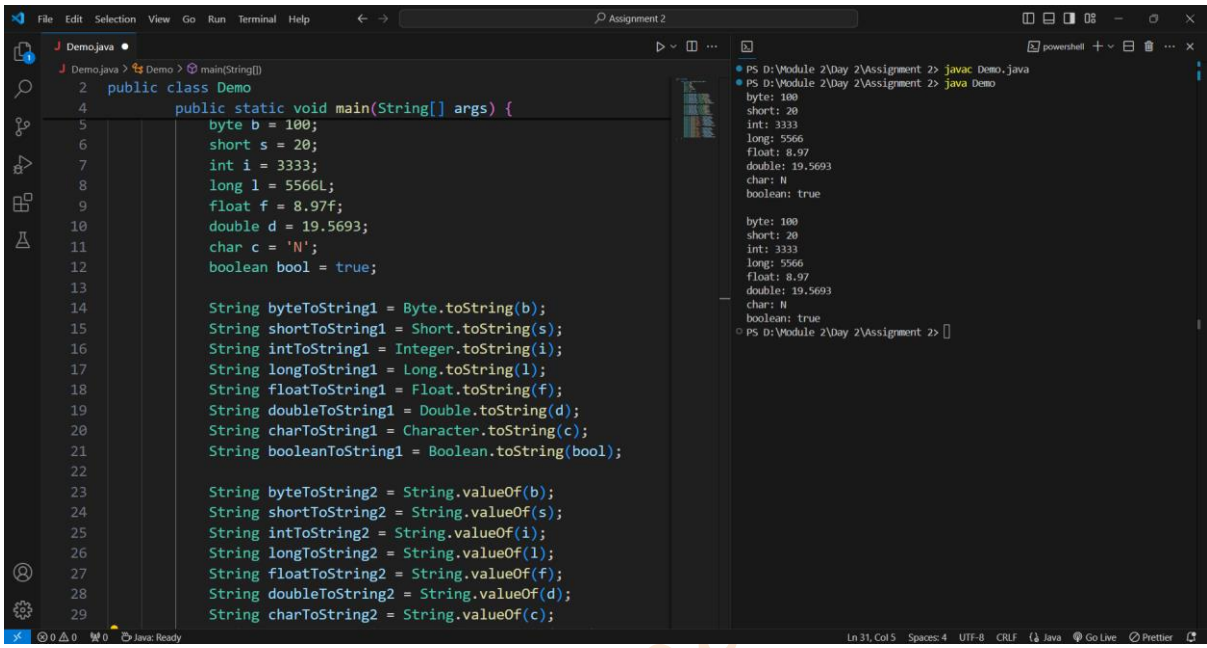
m. Experiment with converting a `double` value into other primitive types or vice versa and observe the results.



8. Conversion between Primitive Types and Strings

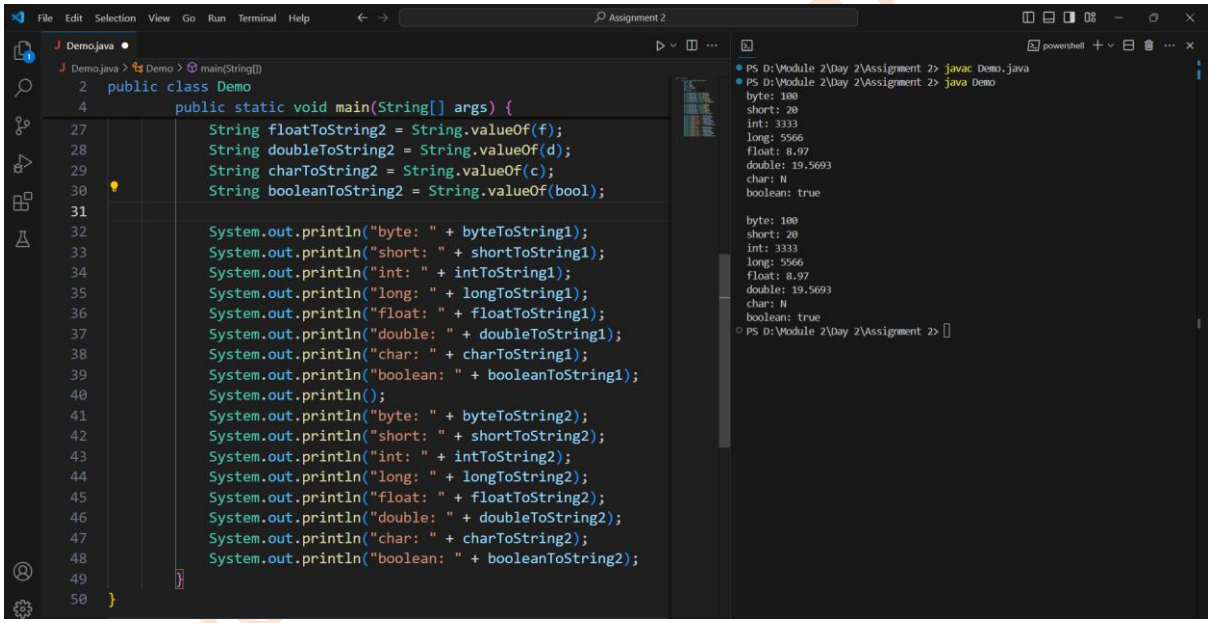
Initialize a variable of each primitive type with a user-defined value and convert it into `String`:

- First, use the `toString` method of the corresponding wrapper class. (e.g., `Integer.toString()`).
- Then, use the `valueOf` method of the `String` class. (e.g., `String.valueOf()`).



```java
public class Demo
    public static void main(String[] args) {
        byte b = 100;
        short s = 20;
        int i = 3333;
        long l = 5566L;
        float f = 8.97f;
        double d = 19.5693;
        char c = 'N';
        boolean bool = true;

        String byteToString1 = Byte.toString(b);
        String shortToString1 = Short.toString(s);
        String intToString1 = Integer.toString(i);
        String longToString1 = Long.toString(l);
        String floatToString1 = Float.toString(f);
        String doubleToString1 = Double.toString(d);
        String charToString1 = Character.toString(c);
        String booleanToString1 = Boolean.toString(bool);

        String byteToString2 = String.valueOf(b);
        String shortToString2 = String.valueOf(s);
        String intToString2 = String.valueOf(i);
        String longToString2 = String.valueOf(l);
        String floatToString2 = String.valueOf(f);
        String doubleToString2 = String.valueOf(d);
        String charToString2 = String.valueOf(c);
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
byte: 100
short: 20
int: 3333
long: 5566
float: 8.97
double: 19.5693
char: N
boolean: true

byte: 100
short: 20
int: 3333
long: 5566
float: 8.97
double: 19.5693
char: N
boolean: true
PS D:\Module 2\Day 2\Assignment 2>
```

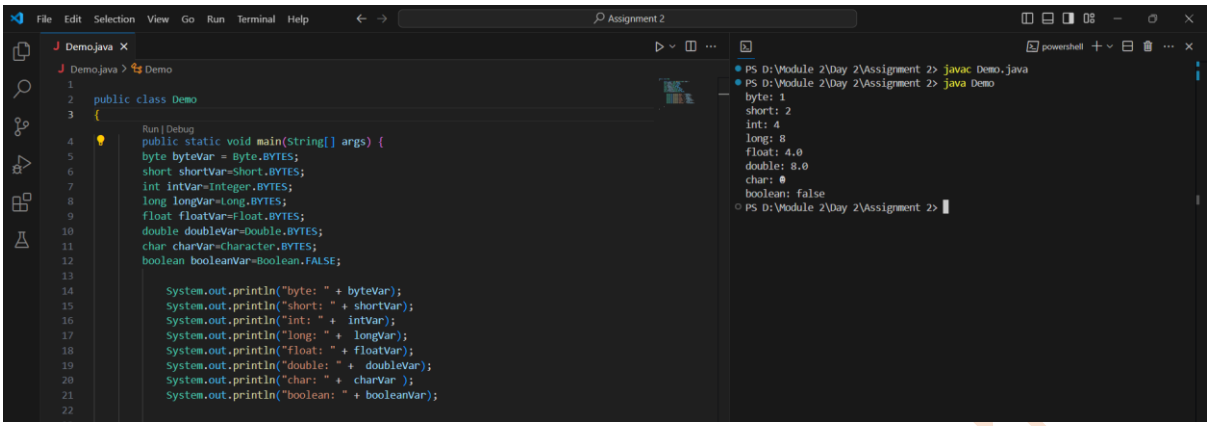


```java
public class Demo
    public static void main(String[] args) {
        String floatToString2 = String.valueOf(f);
        String doubleToString2 = String.valueOf(d);
        String charToString2 = String.valueOf(c);
        String booleanToString2 = String.valueOf(bool);

        System.out.println("byte: " + byteToString1);
        System.out.println("short: " + shortToString1);
        System.out.println("int: " + intToString1);
        System.out.println("long: " + longToString1);
        System.out.println("float: " + floatToString1);
        System.out.println("double: " + doubleToString1);
        System.out.println("char: " + charToString1);
        System.out.println("boolean: " + booleanToString1);
        System.out.println();
        System.out.println("byte: " + byteToString2);
        System.out.println("short: " + shortToString2);
        System.out.println("int: " + intToString2);
        System.out.println("long: " + longToString2);
        System.out.println("float: " + floatToString2);
        System.out.println("double: " + doubleToString2);
        System.out.println("char: " + charToString2);
        System.out.println("boolean: " + booleanToString2);
    }
}
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
byte: 100
short: 20
int: 3333
long: 5566
float: 8.97
double: 19.5693
char: N
boolean: true

byte: 100
short: 20
int: 3333
long: 5566
float: 8.97
double: 19.5693
char: N
boolean: true
PS D:\Module 2\Day 2\Assignment 2>
```

9. Default Values of Primitive Types

Declare variables of each primitive type as fields of a class and check their default values. (Note: Default values depend on whether the variables are instance variables or static variables).

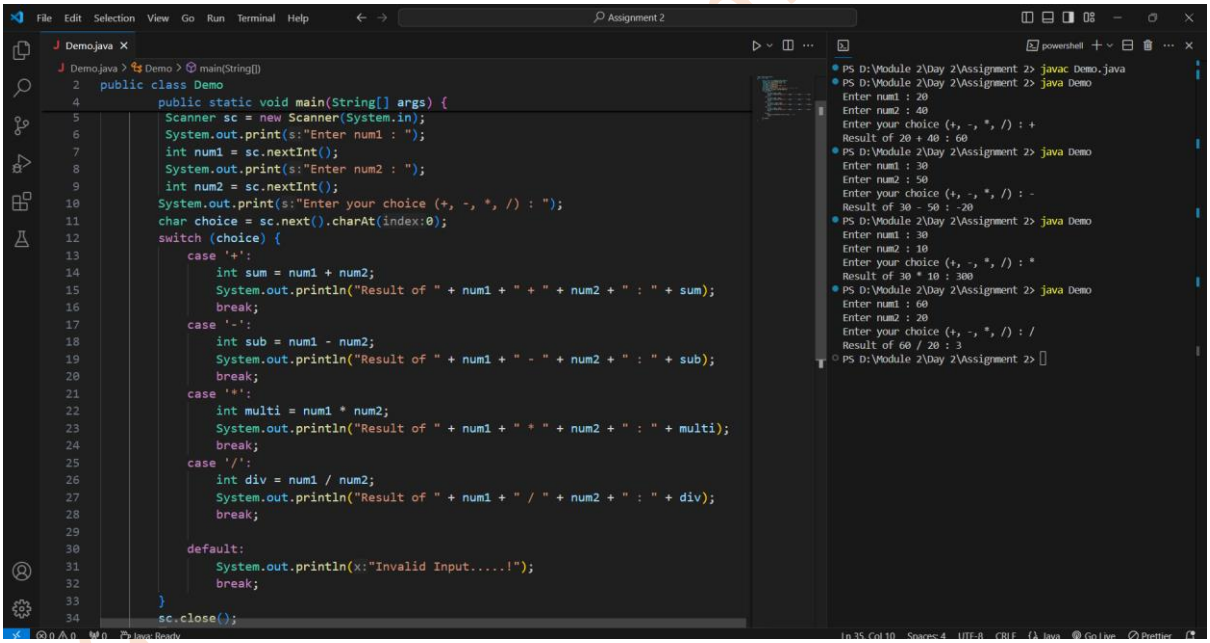


```java
public class Demo
{
    public static void main(String[] args) {
    byte byteVar = Byte.BYTES;
    short shortVar=Short.BYTES;
    int intVar=Integer.BYTES;
    long longVar=Long.BYTES;
    float floatVar=Float.BYTES;
    double doubleVar=Double.BYTES;
    char charVar=Character.BYTES;
    boolean booleanVar=Boolean.FALSE;

        System.out.println("byte: " + byteVar);
        System.out.println("short: " + shortVar);
        System.out.println("int: " +  intVar);
        System.out.println("long: " +  longVar);
        System.out.println("float: " + floatVar);
        System.out.println("double: " +  doubleVar);
        System.out.println("char: " +  charVar );
        System.out.println("boolean: " + booleanVar);
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
byte: 1
short: 2
int: 4
long: 8
float: 4.0
double: 8.0
char: 0
boolean: false
PS D:\Module 2\Day 2\Assignment 2>
```

10. Arithmetic Operations with Command Line Input

Write a program that accepts two integers and an arithmetic operator (`+`, `-`, `*`, `/`) from the command line. Perform the specified arithmetic operation based on the operator provided. (Hint: Use `switch-case` for operations).



```java
public class Demo
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print(s:"Enter num1 : ");
        int num1 = sc.nextInt();
        System.out.print(s:"Enter num2 : ");
        int num2 = sc.nextInt();
        System.out.print(s:"Enter your choice (+, -, *, /) : ");
        char choice = sc.next().charAt(index:0);
        switch (choice) {
            case '+':
                int sum = num1 + num2;
                System.out.println("Result of " + num1 + " + " + num2 + " : " + sum);
                break;
            case '-':
                int sub = num1 - num2;
                System.out.println("Result of " + num1 + " - " + num2 + " : " + sub);
                break;
            case '*':
                int multi = num1 * num2;
                System.out.println("Result of " + num1 + " * " + num2 + " : " + multi);
                break;
            case '/':
                int div = num1 / num2;
                System.out.println("Result of " + num1 + " / " + num2 + " : " + div);
                break;

            default:
                System.out.println(x:"Invalid Input.....!");
                break;
        }
        sc.close();
```

```
PS D:\Module 2\Day 2\Assignment 2> javac Demo.java
PS D:\Module 2\Day 2\Assignment 2> java Demo
Enter num1 : 20
Enter num2 : 40
Enter your choice (+, -, *, /) : +
Result of 20 + 40 : 60
PS D:\Module 2\Day 2\Assignment 2> java Demo
Enter num1 : 30
Enter num2 : 50
Enter your choice (+, -, *, /) : -
Result of 30 - 50 : -20
PS D:\Module 2\Day 2\Assignment 2> java Demo
Enter num1 : 30
Enter num2 : 10
Enter your choice (+, -, *, /) : *
Result of 30 * 10 : 300
PS D:\Module 2\Day 2\Assignment 2> java Demo
Enter num1 : 60
Enter num2 : 20
Enter your choice (+, -, *, /) : /
Result of 60 / 20 : 3
PS D:\Module 2\Day 2\Assignment 2>
```