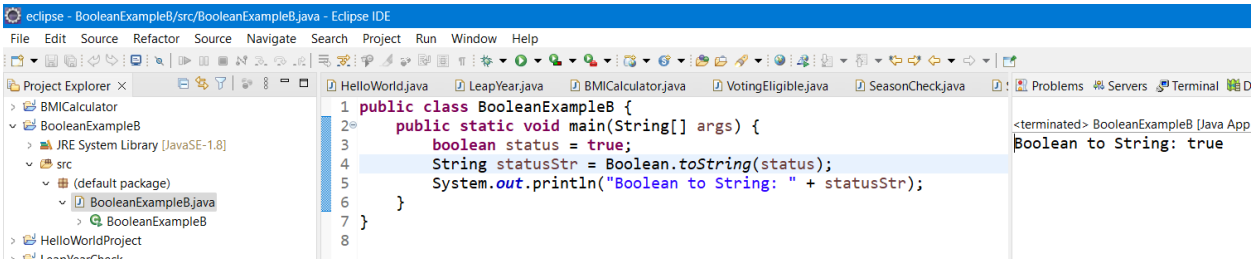
Assignment

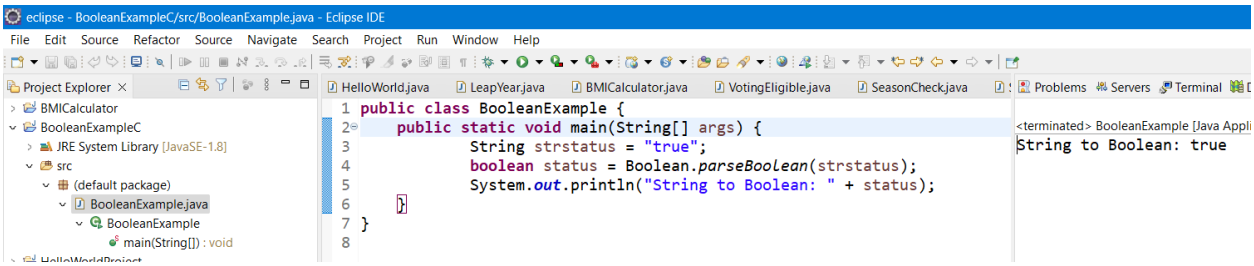
1. Working with java.lang. Boolean

a. Explore the lava API documentation for java.lang. Boolean and observe its modifiers and super types. (Reading)

b. Declare a 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)).

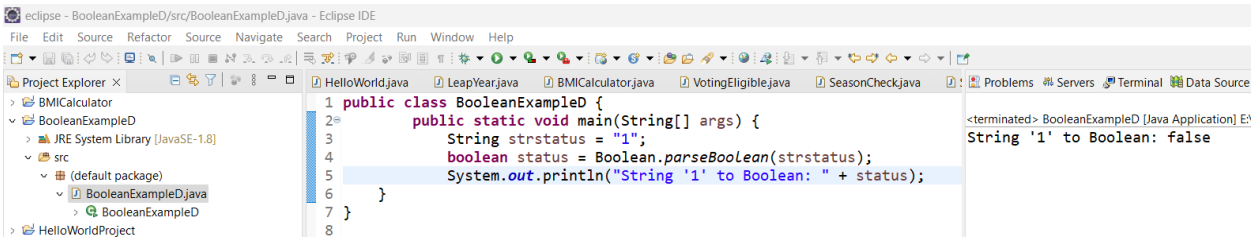


c. Declare a method-local variable strstatus of type String with the value "true" and convert it to a boolean using the parseBoolean method. (Hint: Use Boolean.parseBoolean (String)).

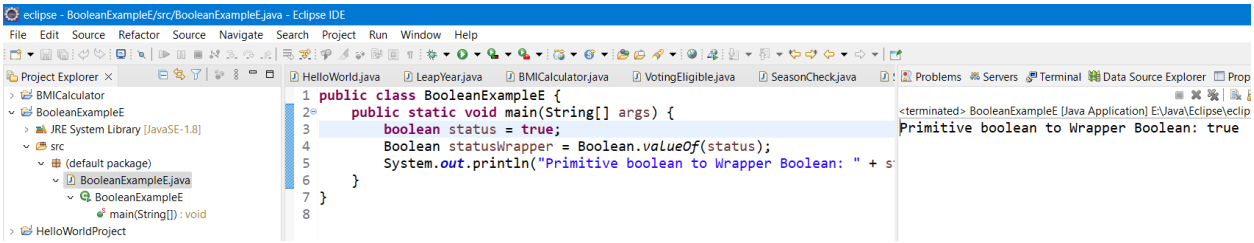


d. Declare a method-local variable strstatus of type String with the value "1" or "0" and attempt to convert it to a boolean. (Hint: parseBoolean

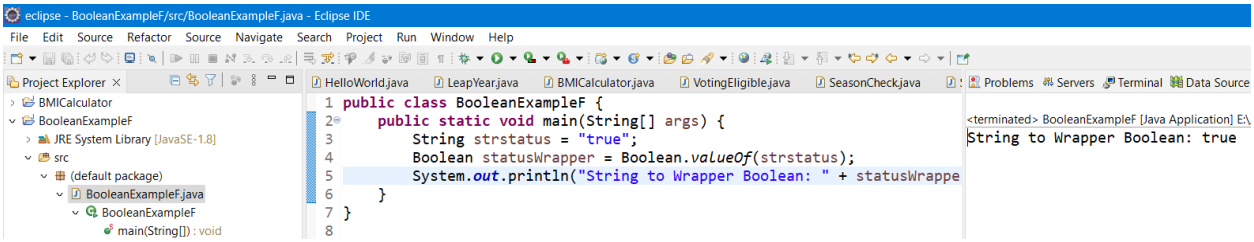
method will not work as expected with "1" or "0").



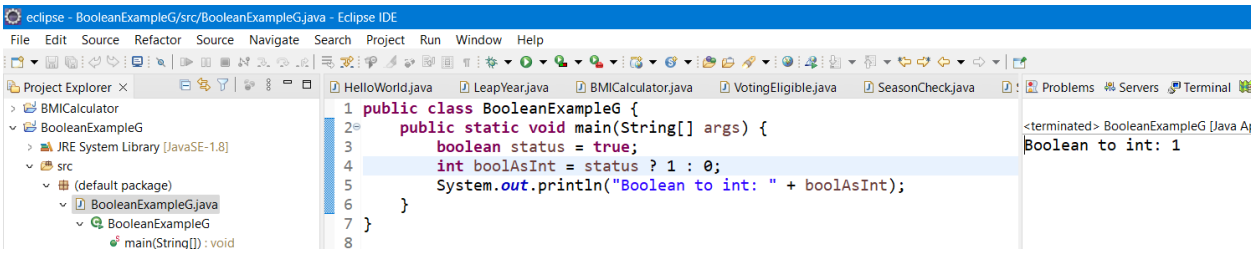
e. Declare a method-local variable status of type boolean with the value true and convert it to the corresponding wrapper class using Boolean.valueOf(). (Hint: Use Boolean.valueOf (boolean)).

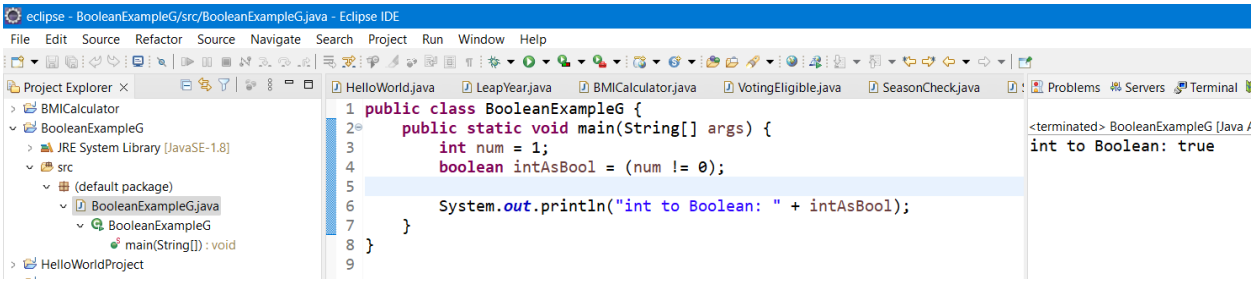


f. Declare a method-local variable strstatus of type string with the value "true" and convert it to the corresponding wrapper class using Boolean.valueOf(). (Hint: Use Boolean.valueOf(String)).



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

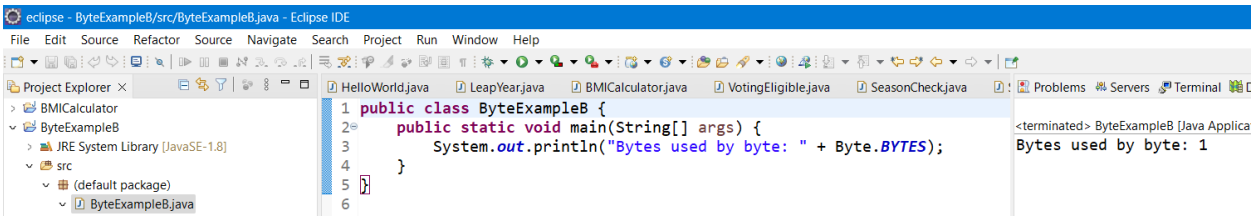




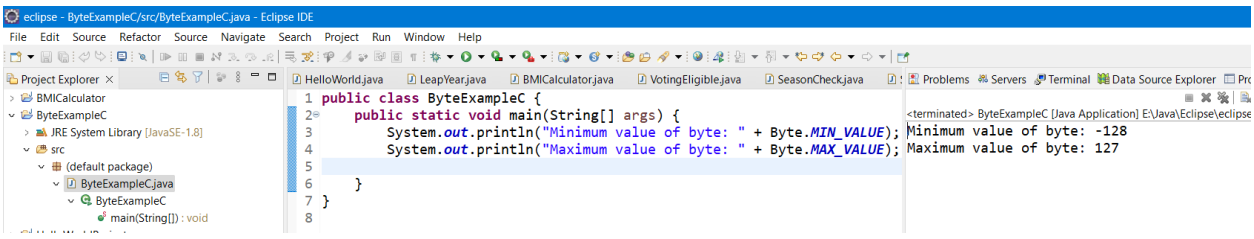
2. Working with java.lang. Byte

a. Explore the Java API documentation for java.lang. Byte and observe its modifiers and super types. (Reading Only)

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

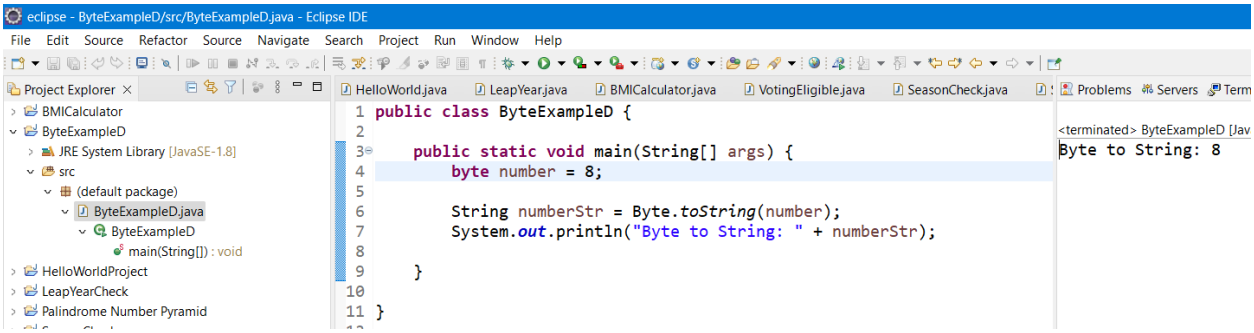


c. Write a program to find the minimum and maximum values of byte using the MIN VALUE and MAX VALUE fields. (Hint: Use Byte. MIN VALUE and Byte.MAX_VALUE).

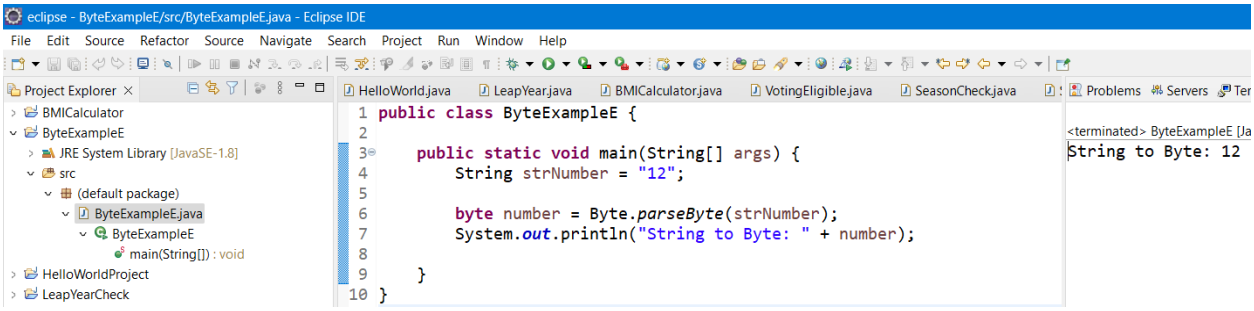


d. Declare a method-local variable number of type byte with some value and convert it to a String using the tostring method. (Hint: Use Byte.toString

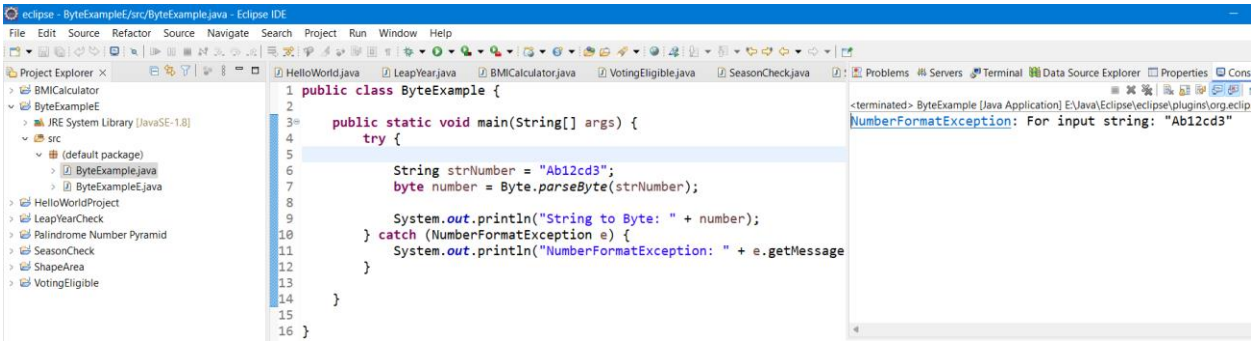
(byte)).



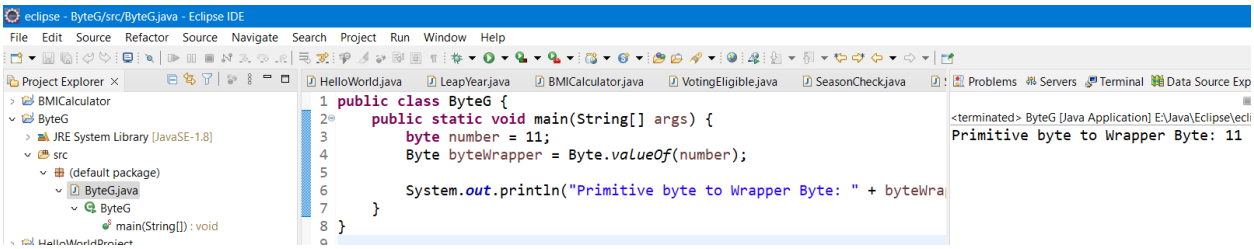
e. Declare a method-local variable strNumber of type string with some value and convert it to a byte value using the parseByte method. (Hint: Use Byte.parseByte (String)).



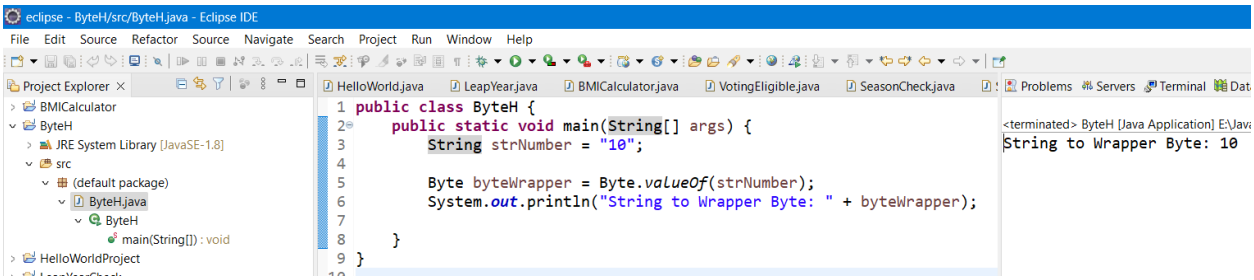
f. Declare a method-local variable strNumber of type string with the value "Ab12cd3" and attempt to convert it to a byte value. (Hint: parseByte method will throw a NumberFormatException).



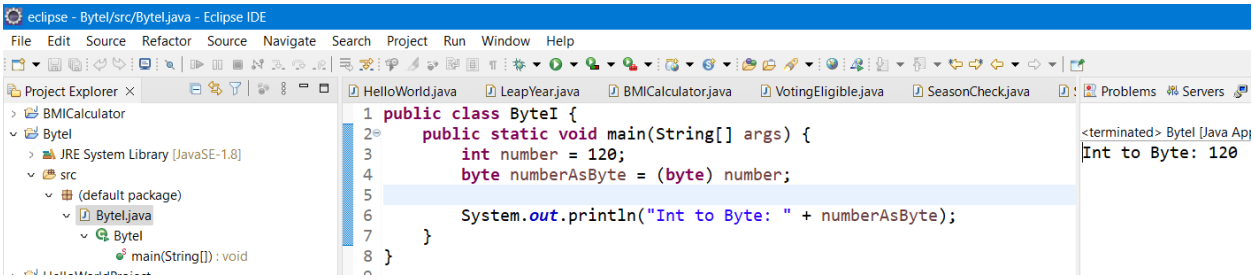
g. Declare a method-local variable number of type byte with some value and convert it to the corresponding wrapper class using Byte.valueOf(). (Hint: Use Byte.valueOf(byte)).



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)).



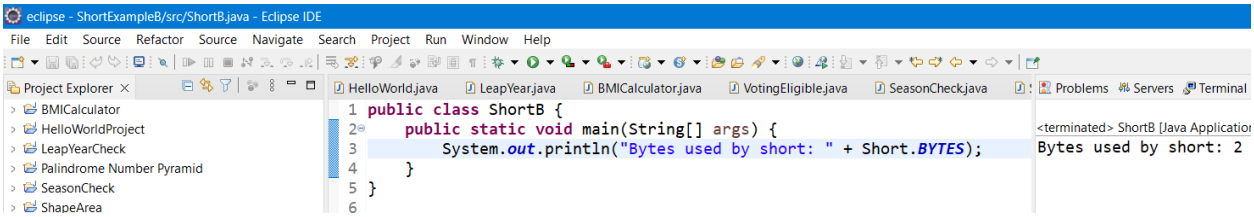
i. Experiment with converting a byte value into other primitive types or vice versa & observe the results.



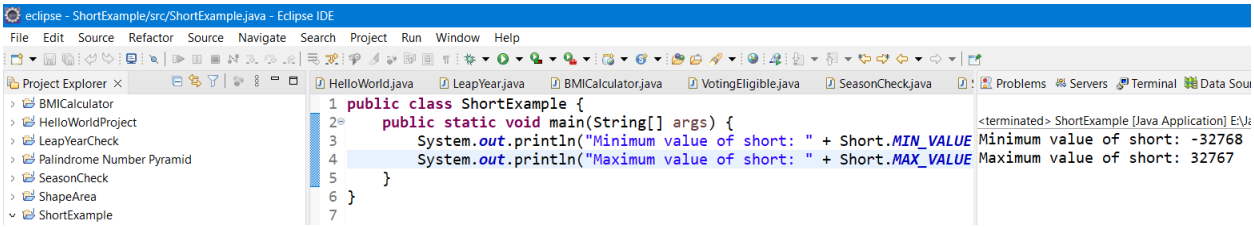
3. Working with java.lang.Short

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

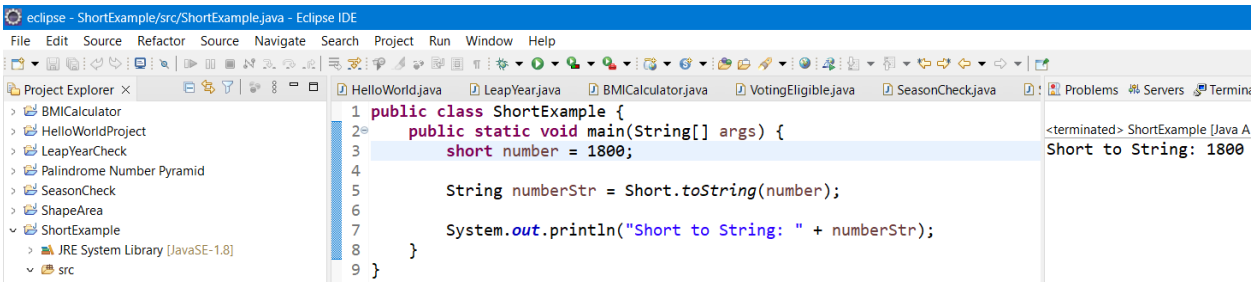
b. Write a program to test how many bytes are used to represent a short value using the BYTES field. (Hint: Use Short. BYTES).



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).

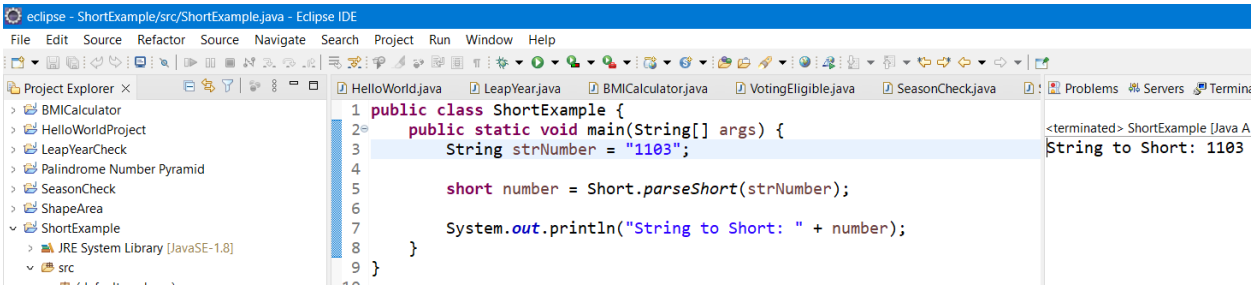


d. Declare a method-local variable number of type short with some value and convert it to a String using the tosString method. (Hint: Use Short.toString(short)).

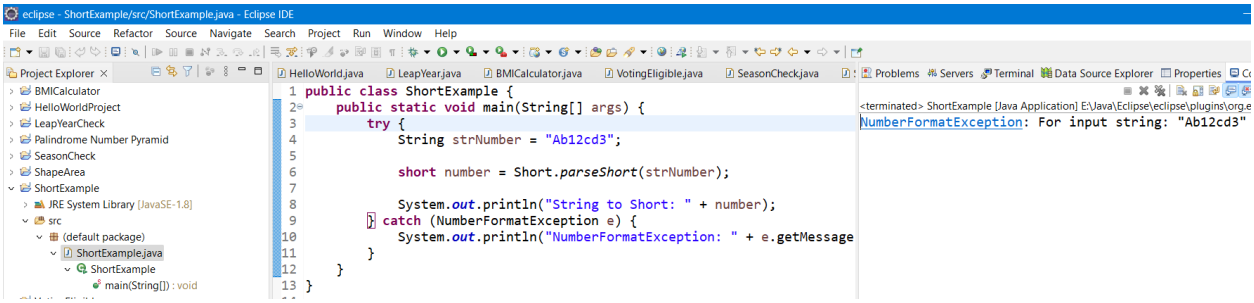


e. Declare a method-local variable strNumber of type string with some value and convert it to a short value using the parseShort method. (Hint: Use

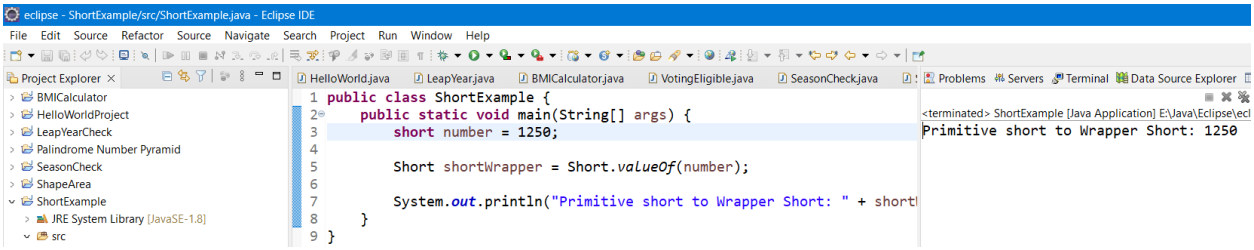
Short.parseShort (String)).



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).

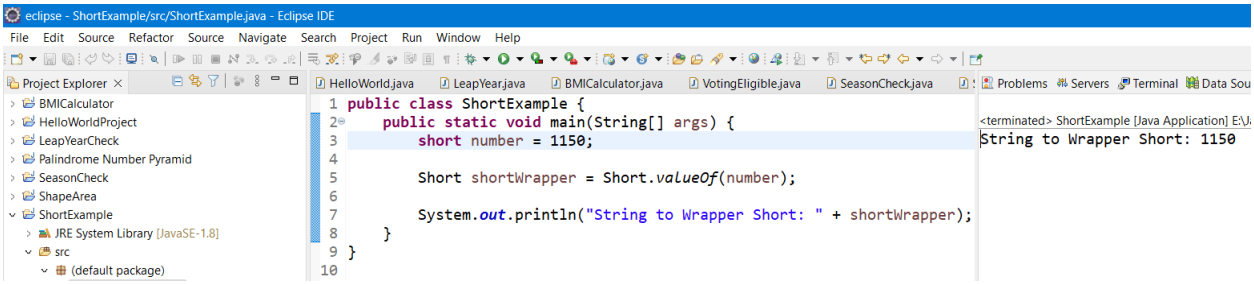


g. Declare a method-local variable number of type short with some value and convert it to the corresponding wrapper class using Short.valueOf(). (Hint: Use Short.valueOf(short)).

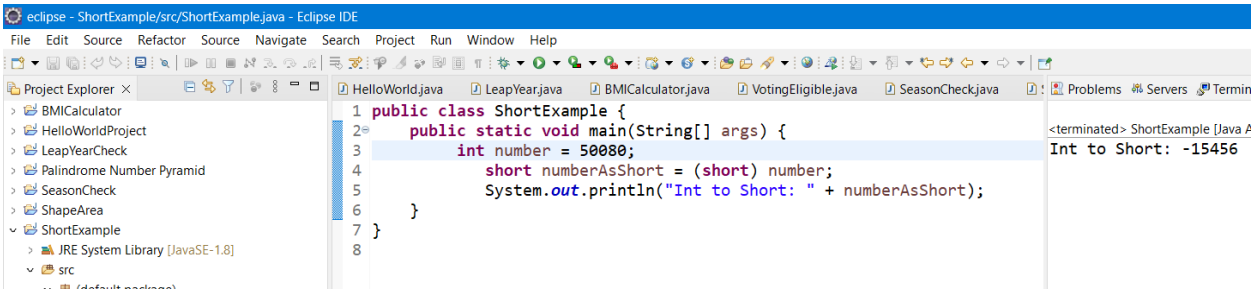


h. Declare a method-local variable strNumber of type String with some short value and convert it to the corresponding wrapper class using Short.valueOf().

(Hint: Use Short.valueOf (String)).



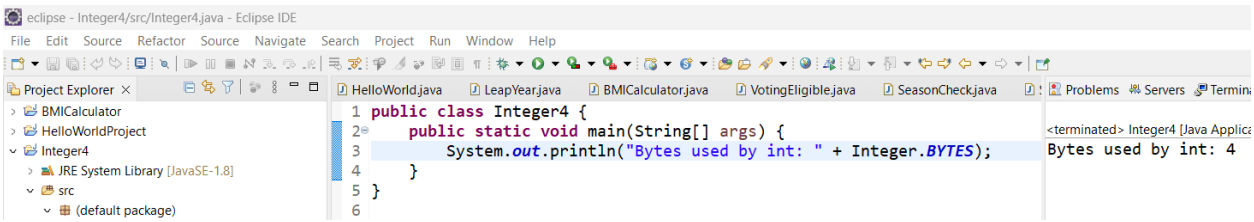
i. Experiment with converting a short value into other primitive types or vice versa and observe the results.



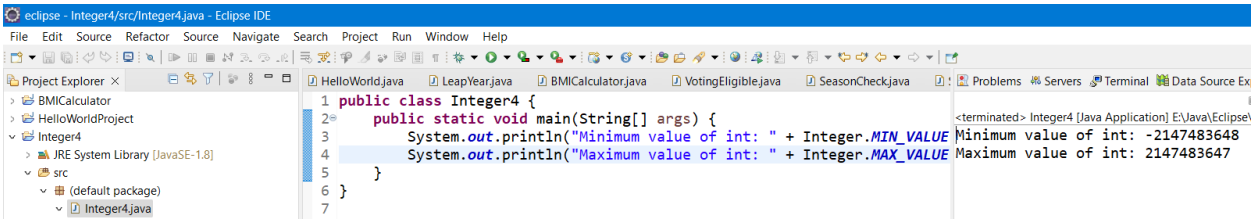
4. Working with java.lang. Integer

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

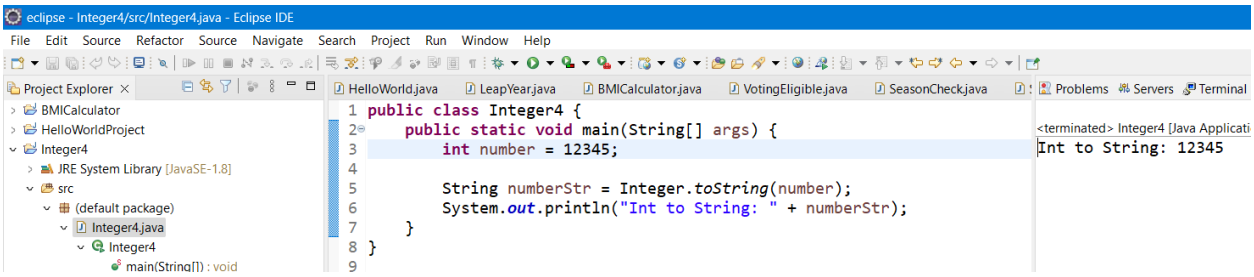
b. Write a program to test how many bytes are used to represent an int value using the BYTES field. (Hint: Use Integer.BYTES).



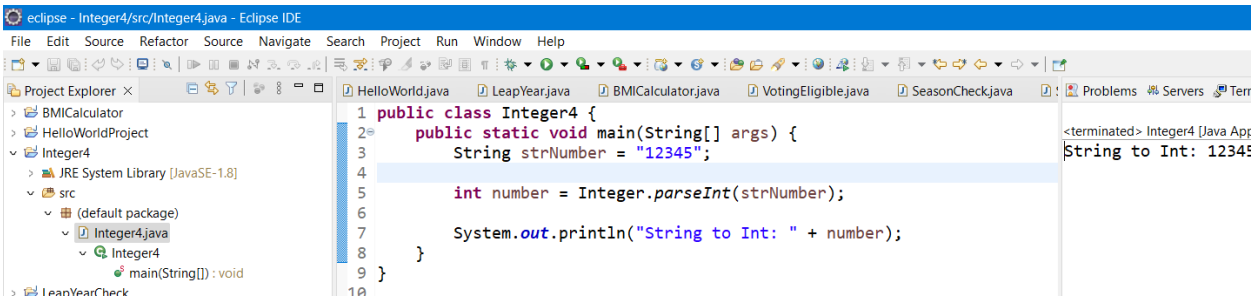
c. Write a program to find the minimum and maximum values of int using the MIN VALUE and MAX_VALUE fields. (Hint: Use Integer.MIN_VALUE and Integer.MAX_VALUE).



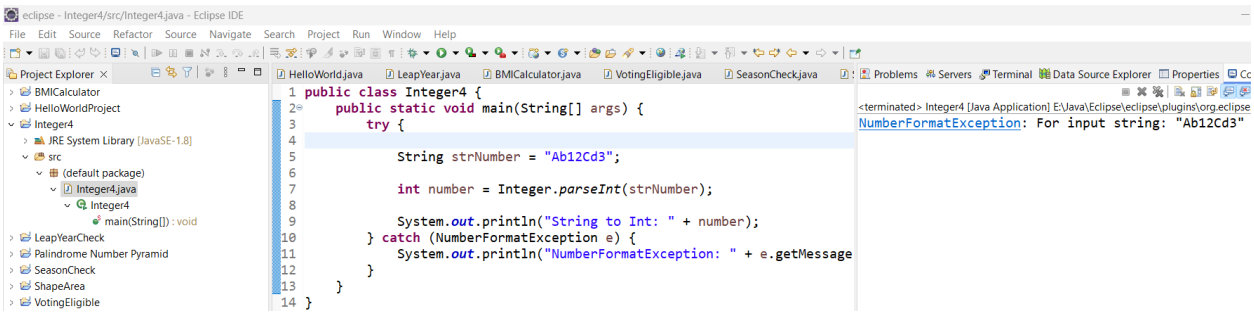
d. Declare a method-local variable number of type int with some value and convert it to a string using the tostring method. (Hint: Use Integer.toString(int)).



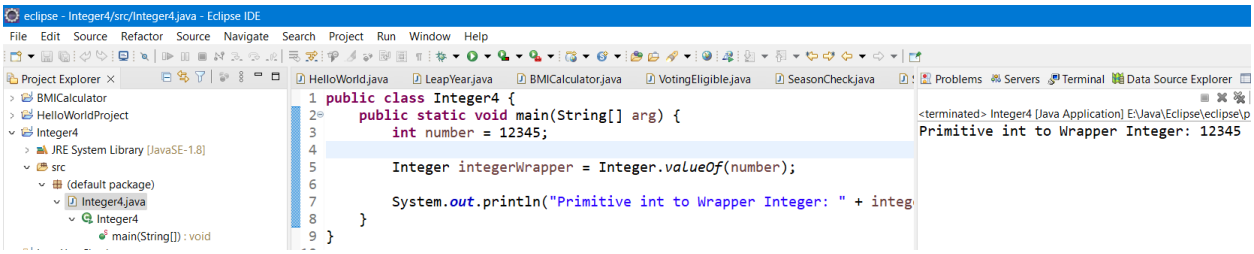
e. Declare a method-local variable strNumber of type String with some value and convert it to an int value using the parseint method. (Hint: Use Integer.parseInt (String)).



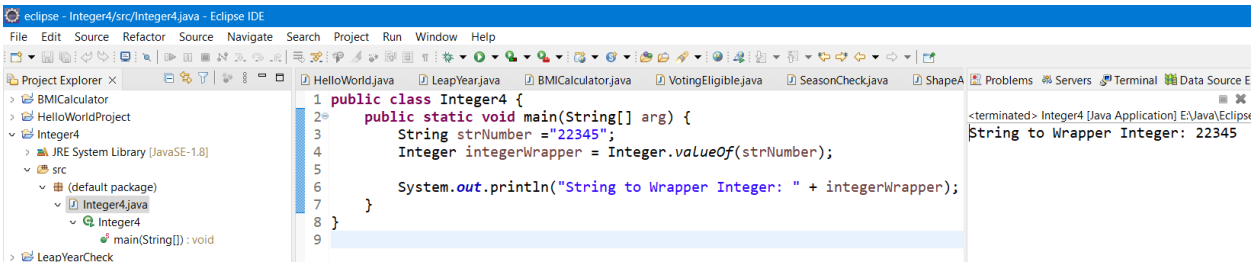
f. Declare a method-local variable strNumber of type string with the value "Ab12Cd3" and attempt to convert it to an int value. (Hint: parseInt method will throw a NumberFormatException).



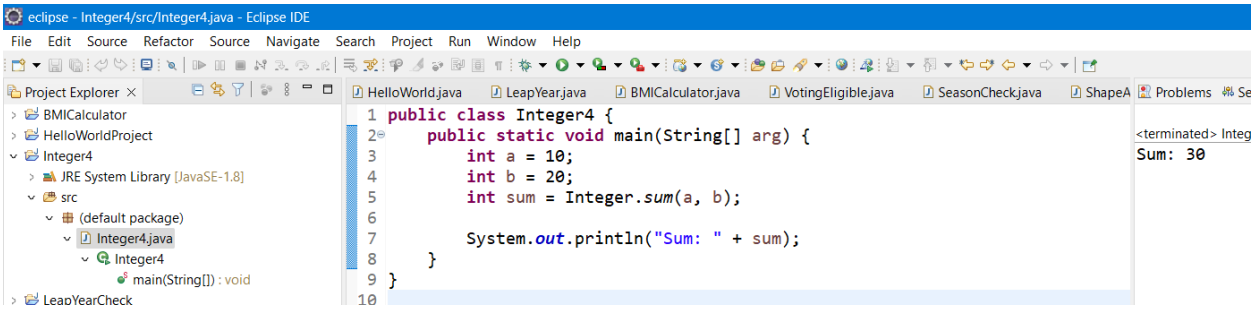
g. Declare a method-local variable number of type int with some value and convert it to the corresponding wrapper class using Integer.valueOf(). (Hint: Use Integer.valueOf(int)).



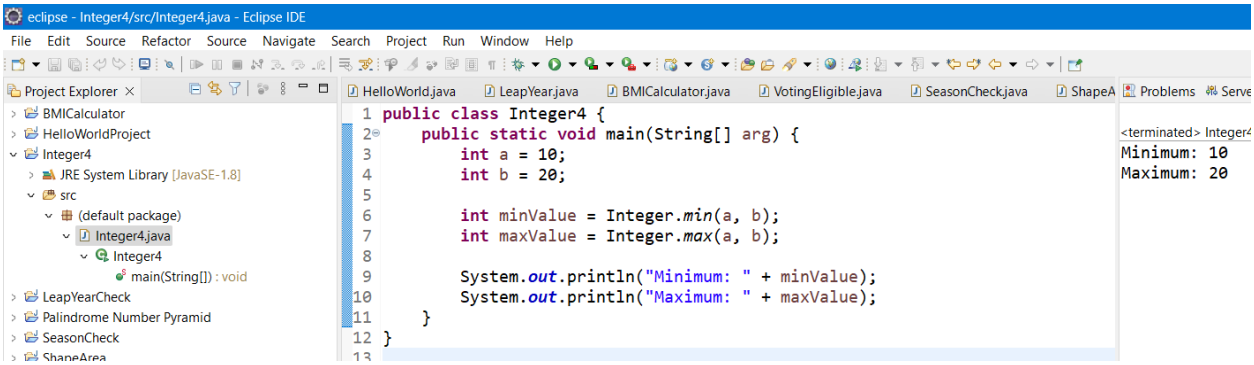
h. Declare a method-local variable strNumber of type String with some integer value and convert it to the corresponding wrapper class using Integer.valueOf(). (Hint: Use Integer.valueOf (String)).



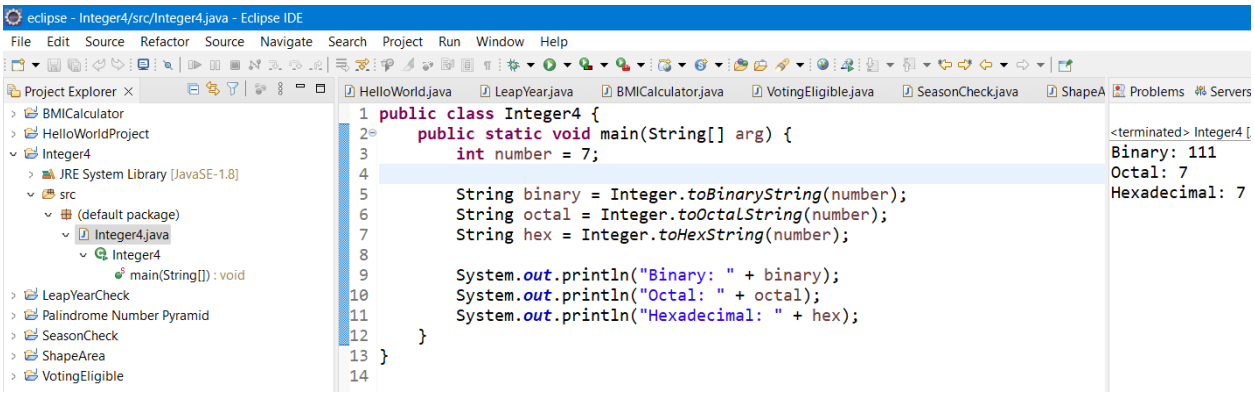
i. Declare two integer variables with values 10 and 20, and add them using a method from the Integer class. (Hint: Use Integer.sum(int, int)).



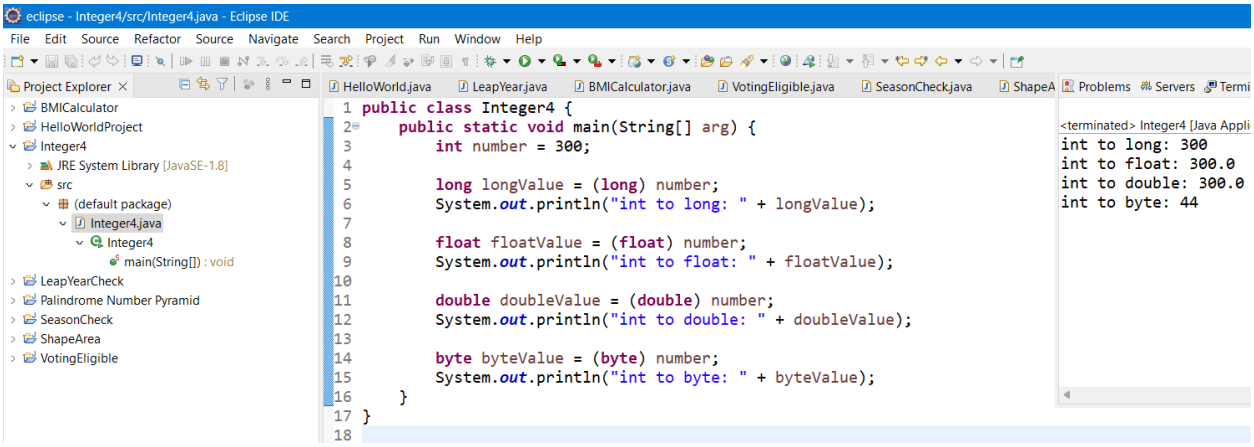
j. Declare two integer variables with values 10 and 20, and find the minimum and maxirnum values using the Integer class. (Hint: Use Integer.min(int, int) and Integer.max(int, int)).



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 & observe the results.

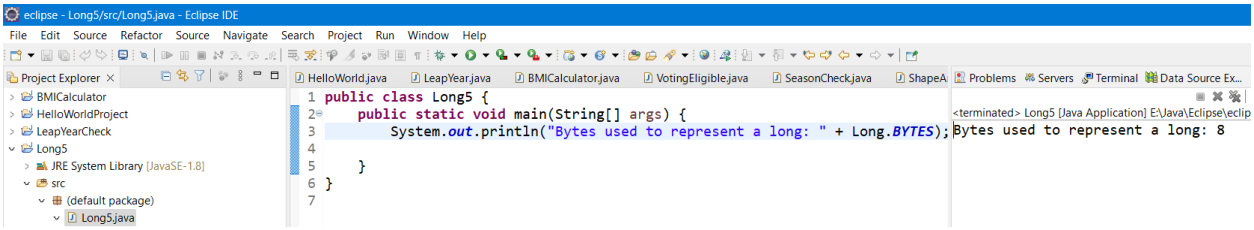



```
1  public class Integer4 {
2      public static void main(String[] arg) {
3          int number = 300;
4
5          long longValue = (long) number;
6          System.out.println("int to long: " + longValue);
7
8          float floatValue = (float) number;
9          System.out.println("int to float: " + floatValue);
10
11         double doubleValue = (double) number;
12         System.out.println("int to double: " + doubleValue);
13
14         byte byteValue = (byte) number;
15         System.out.println("int to byte: " + byteValue);
16     }
17 }
18
```

5. Working with java.lang. Long

a. Explore the lava API documentation for java.lang. Long and observe its modifiers and super types. (Reading)

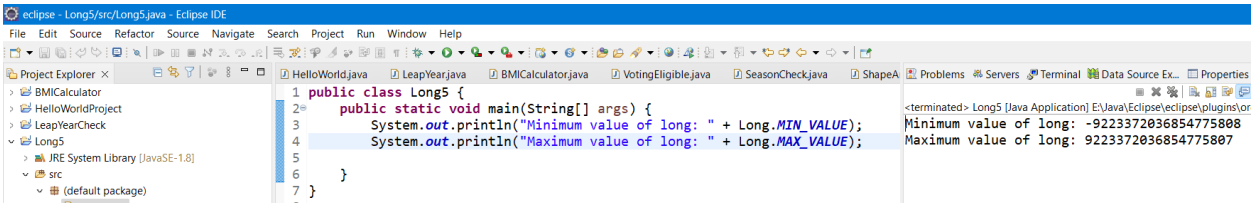
b. Write a program to test how many bytes are used to represent a long value using the BYTES field. (Hint: Use Long. BYTES).



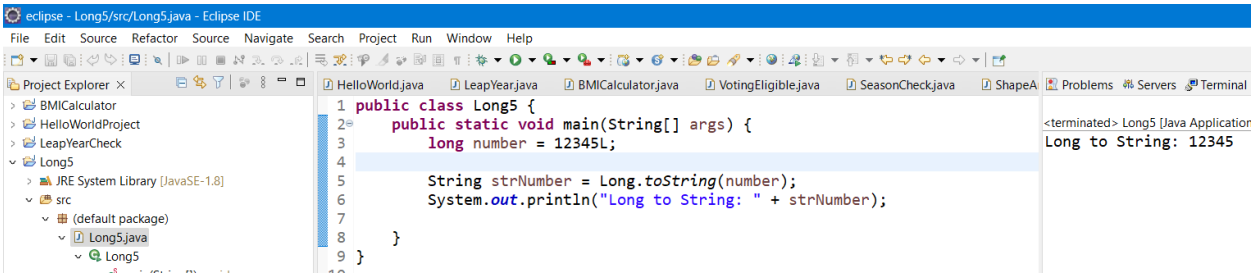

```
1  public class Long5 {
2      public static void main(String[] args) {
3          System.out.println("Bytes used to represent a long: " + Long.BYTES);
4
5      }
6  }
7
```

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

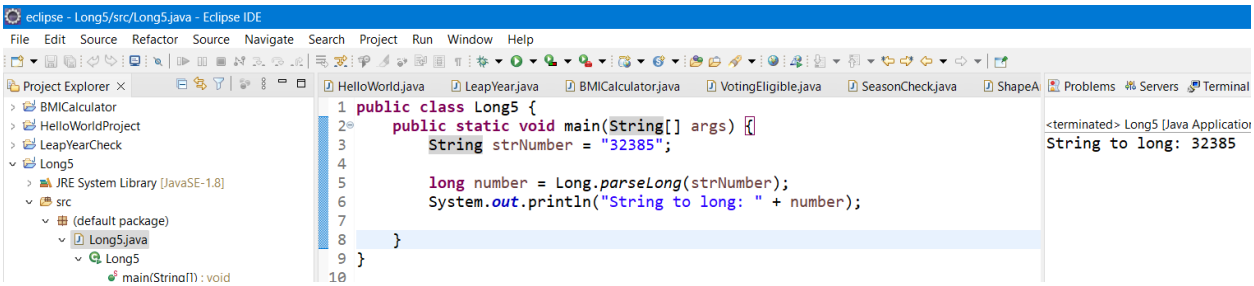
Long.MAX_VALUE).



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)).

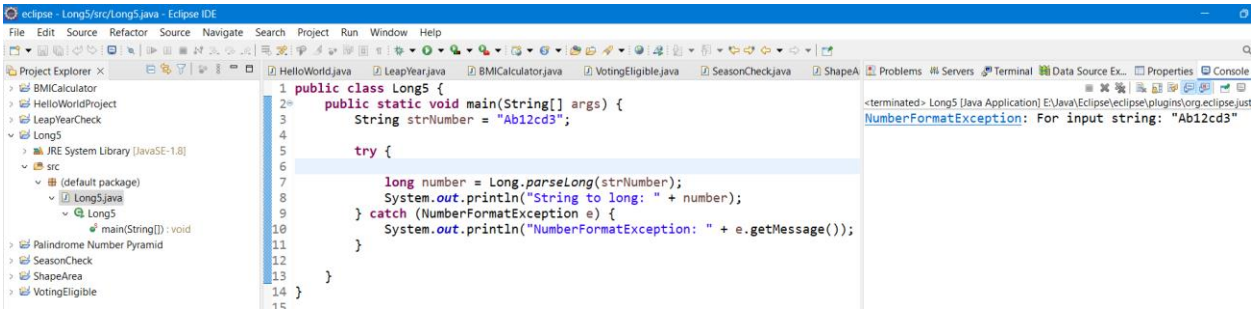


e. Declare a method-local variable strNumber of type string with some value and convert it to a long value using the parseLong method. (Hint: Use Long.parseLong (String)).

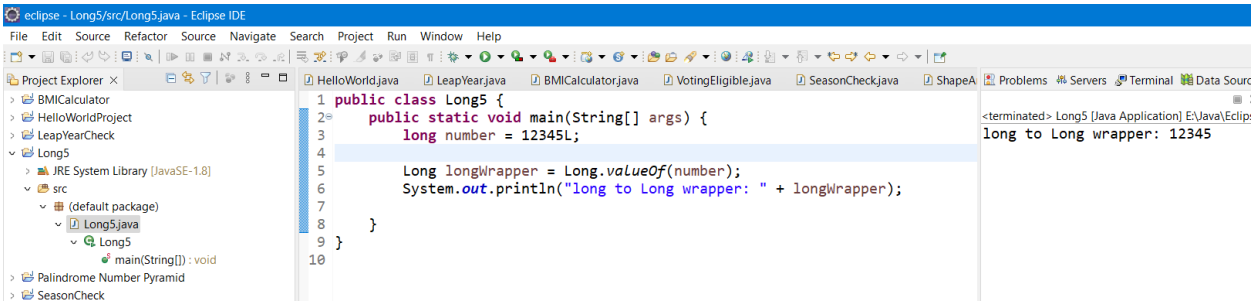


f. Declare a method-local variable strNumber of type string with the value "Ab12cd3" and attempt to convert it to a long value. (Hint: parseLong method

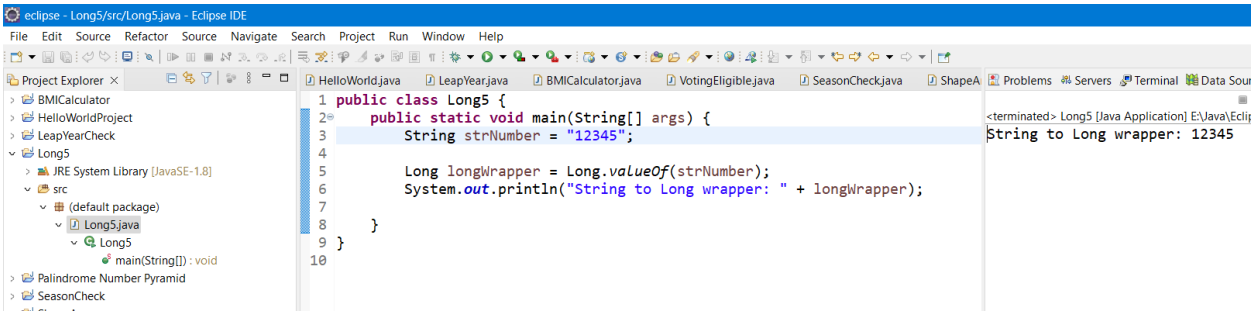
will throw a NumberFormatException).



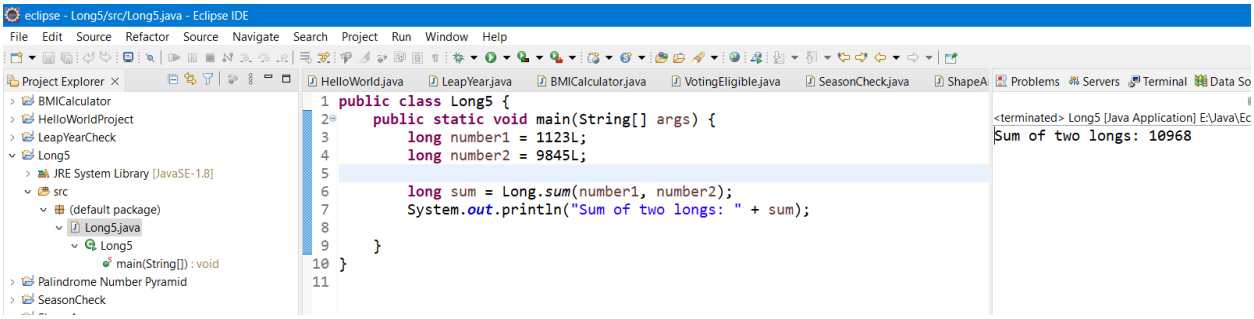
g. Declare a method-local variable number of type long with some value and convert it to the corresponding wrapper class using Long.valueOf(). (Hint: Use Long.valueOf(long)).



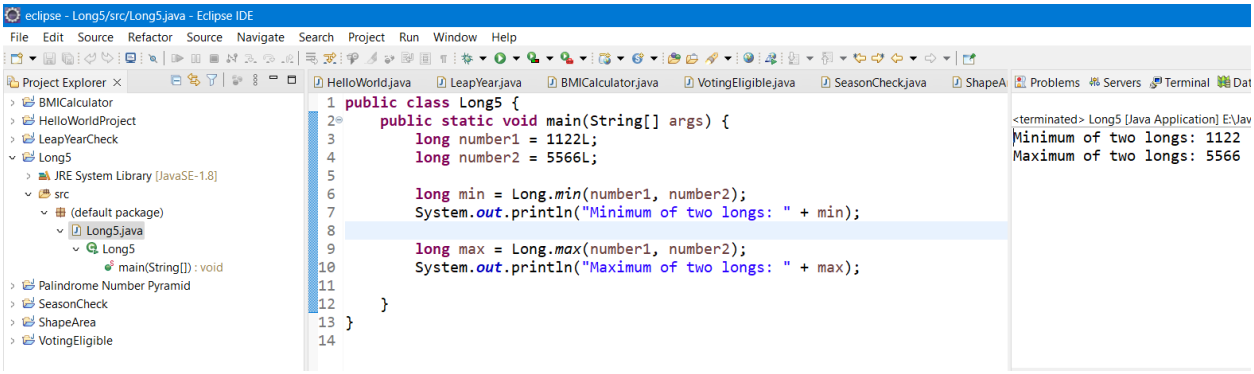
h. Declare a method-local variable strNumber of type string with some long value and convert it to the corresponding wrapper class using Long.valueOf(). (Hint: Use Long.valueOf(String)).



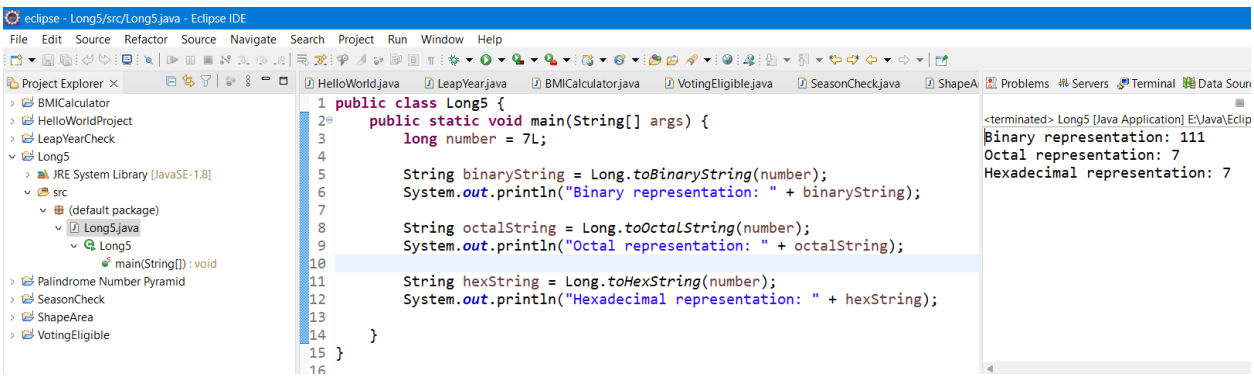
i. Declare two long variables with values 1123 and 9845, and add them using a method from the Long class. (Hint: Use Long, sum (long, long))



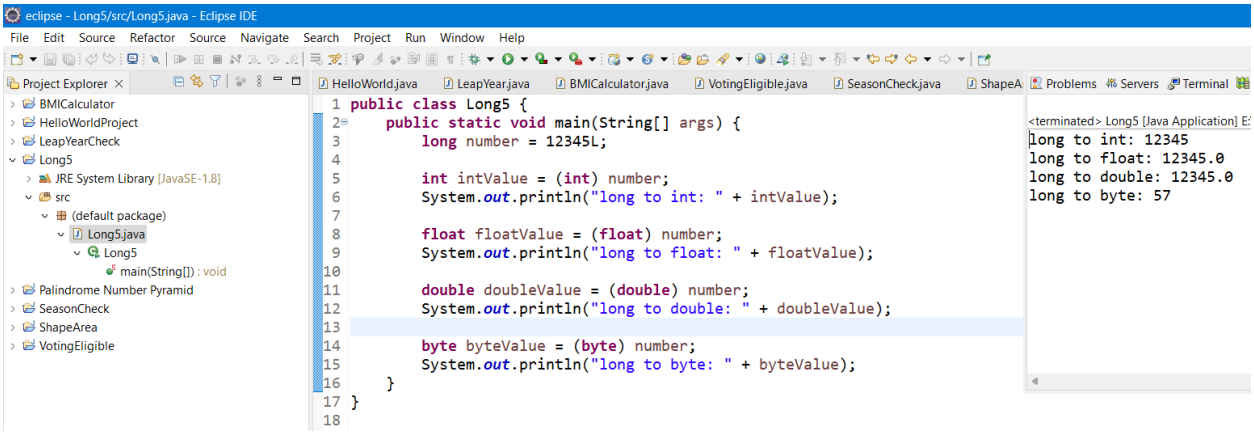
j. Declare two long variables with values 1122 and 5566, and find the minimum and maximum values using the Long class. (Hint: Use Long min (long, long) and Long-max (long, long)).



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)).



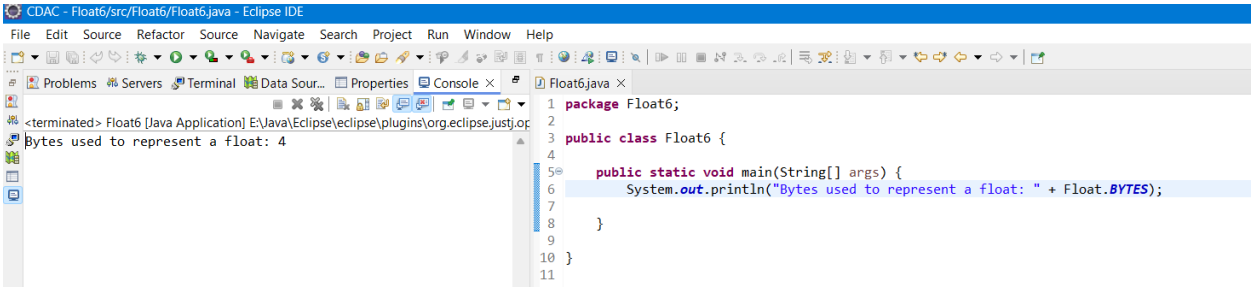
l. Experiment with converting a long value into other primitive types or vice versa and observe the results.



6. Working with java.lang. Float

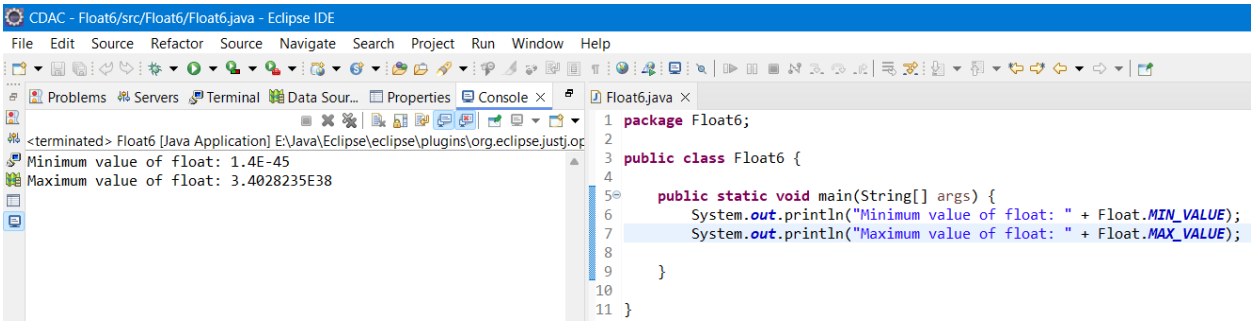
a. Explore the Java API documentation for java.lang. Float and observe its modifiers and super types. (Reading)

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

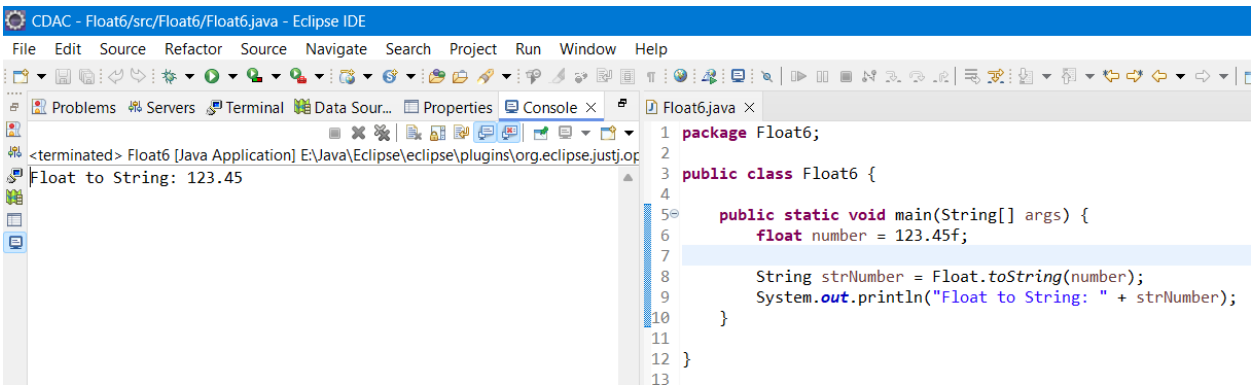


c. Write a program to find the minimum and maximum values of float using the MIN VALUE and MAX VALUE fields. (Hint: Use Float. MIN VALUE and

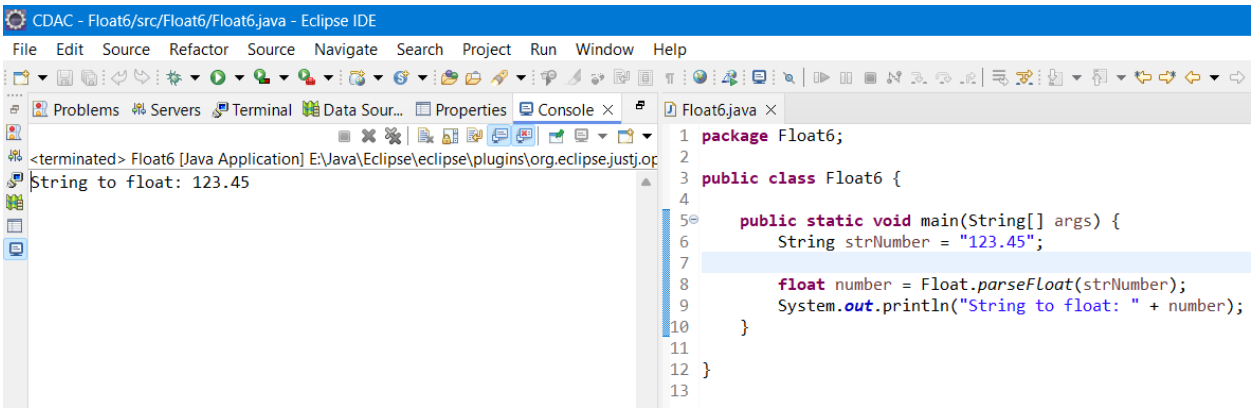
Float.MAX_VALUE).



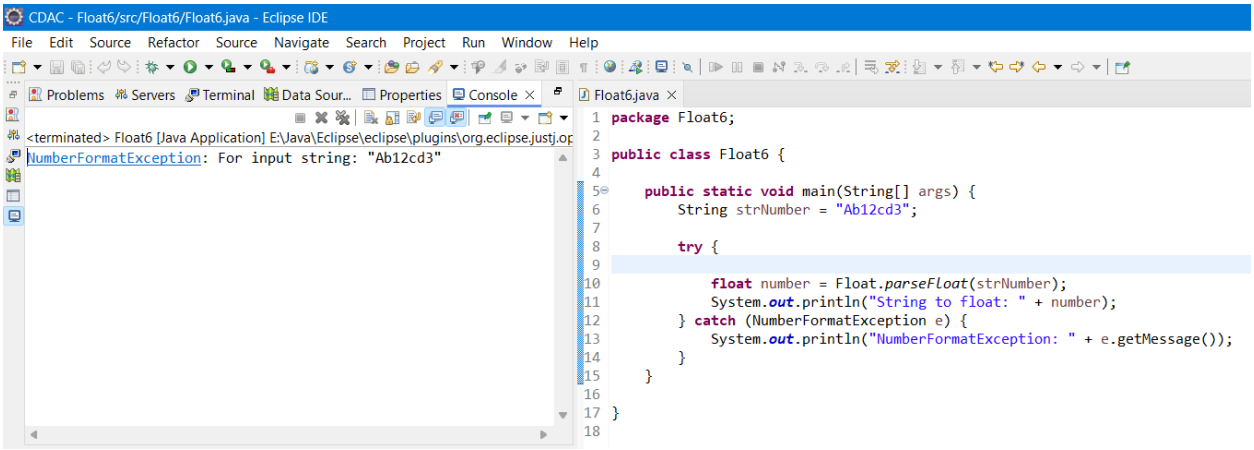
d. Declare a method-local variable numb number of type float with some value and convert it to a string using the tostring method. (Hint: Use Float.toString (float)).



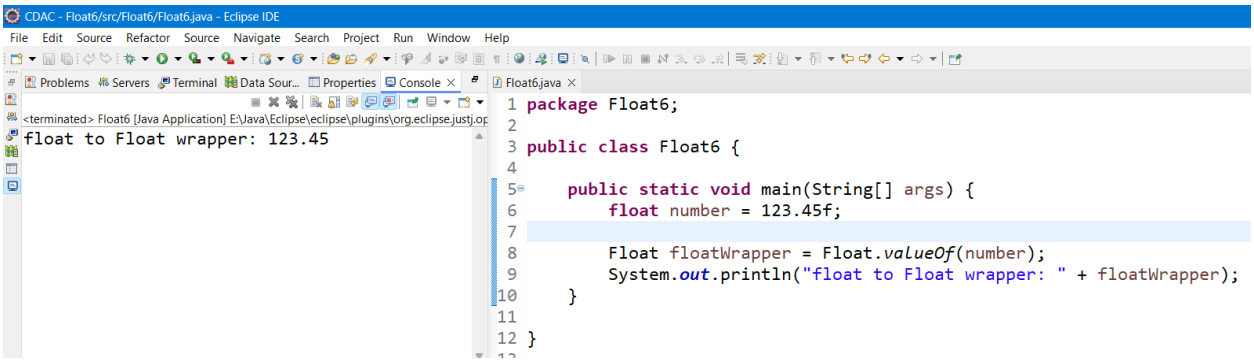
e. Declare a method-local variable strNumber of type String with some value and convert it to a float value using the parseFloat method. (Hint: Use Float.parseFloat (String)).



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).

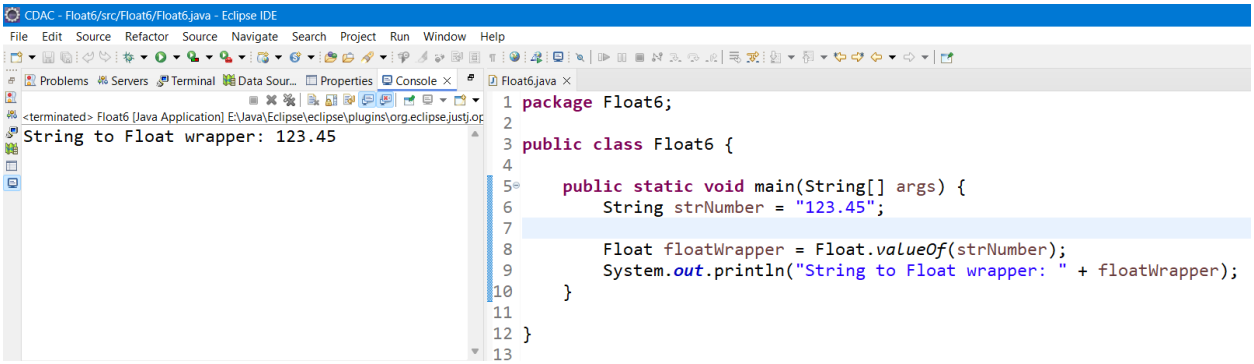


g. Declare a method-local variable number of type float with some value and convert it to the corresponding wrapper class using Float.valueOf(). (Hint: Use Float.valueOf (float)).

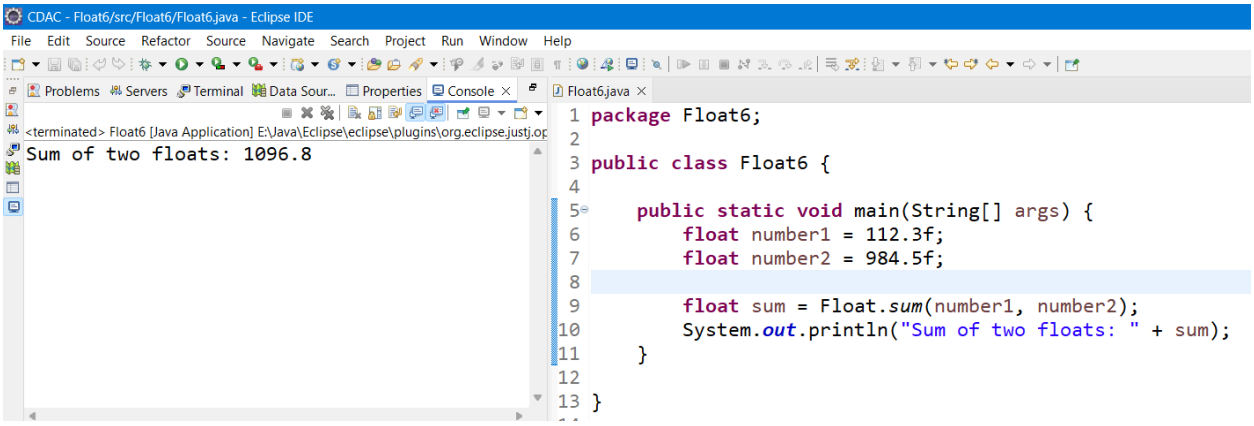


h. Declare a method-local variable strNumber of type String with some float value and convert it to the corresponding wrapper class using Float.valueOf().

(Hint: Use Float.valueOf(String)).

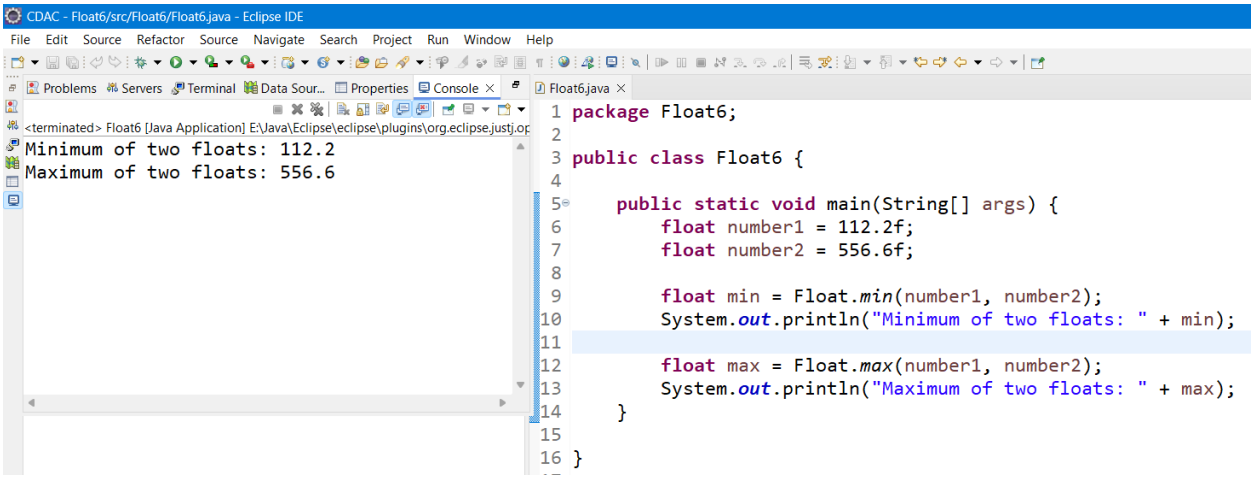


i. Declare two float variables with values 112.3 and 984.5, and add them using a method from the Float class. (Hint: Use Float.sum(float, float)).

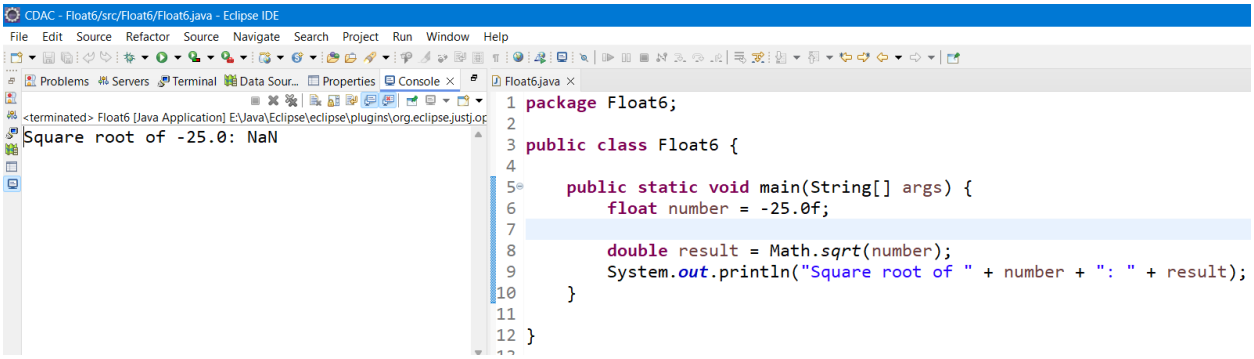


j. Declare two float variables with values 112.2 and 556.6, and find the minimum and maximum values using the Float class. (Hint: Use Float min

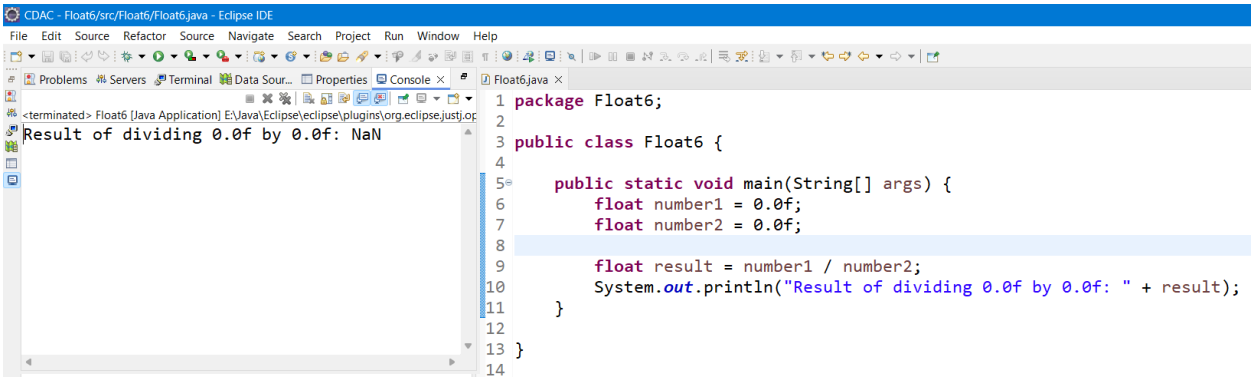
(float, float) and Float.max (float, float)).



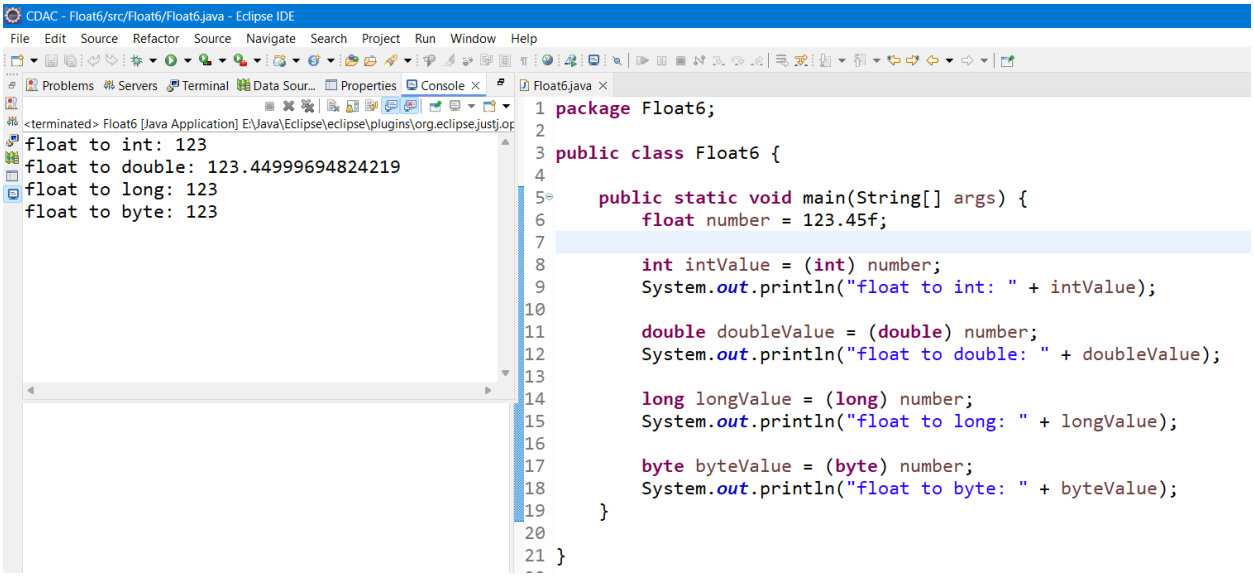
k. Declare a float variable with the value-25.0f. Find the square root of this value. (Hint: Use Math.sqrt() method).



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



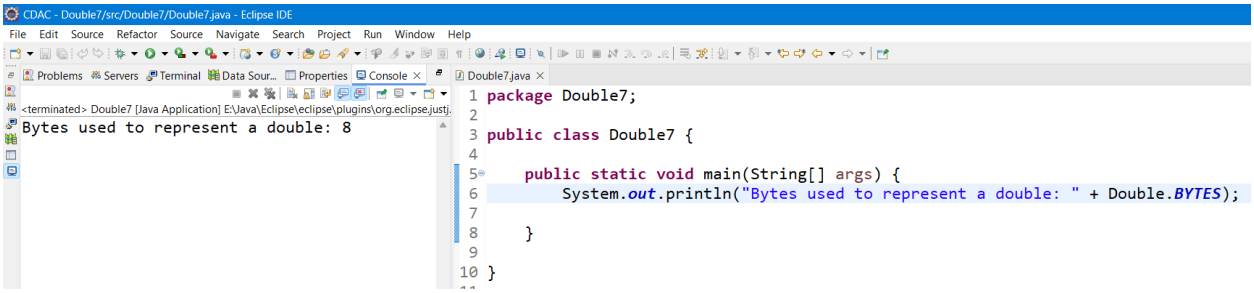
m. Experiment with converting a float value into other primitive types or vice versa and observe the results.



7. Working with java.lang. Double

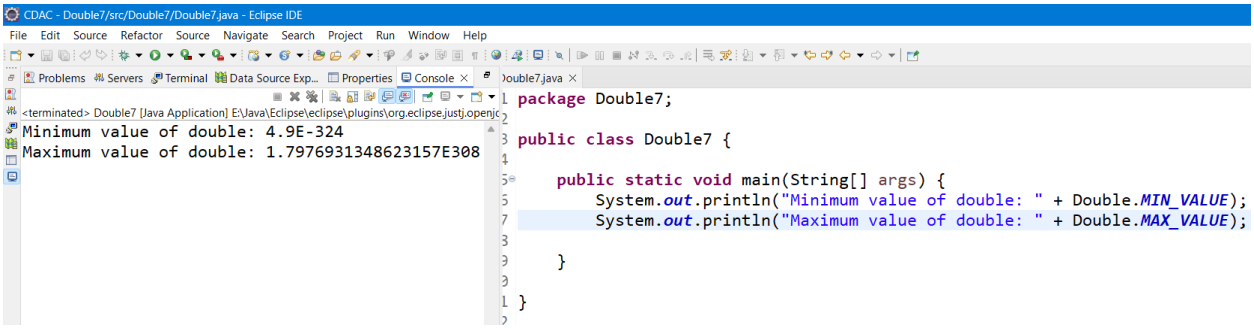
a. Explore the lava API documentation for java.lang. Double and observe its modifiers and super types. (Reading)

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

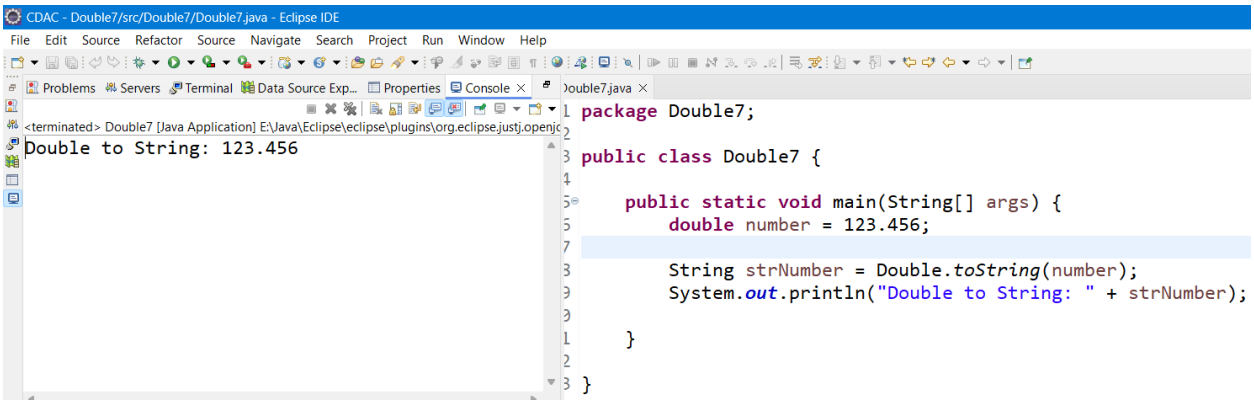


c. Write a program to find the minimum and maximum values of double using the MIN VALUE and MAX VALUE fields. (Hint: Use Double.MIN VALUE and

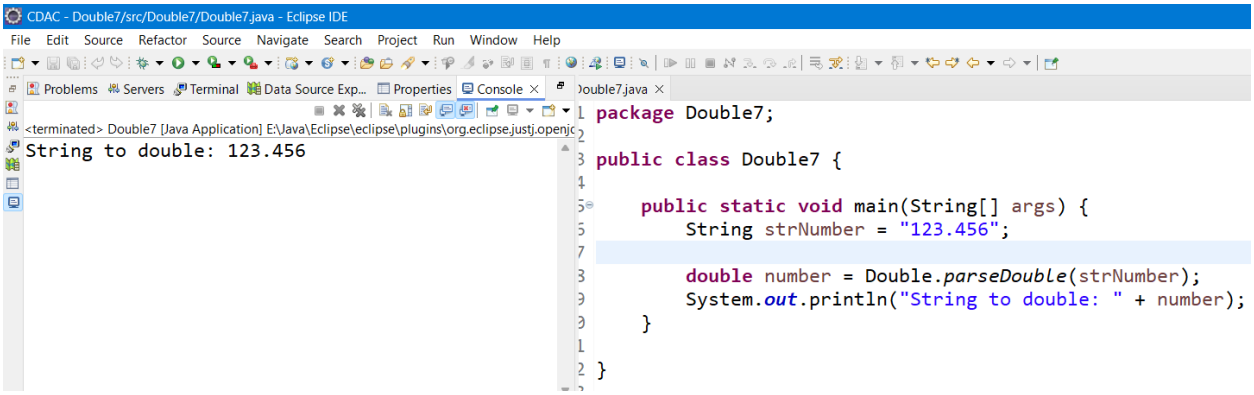
Double MAX VALUE).



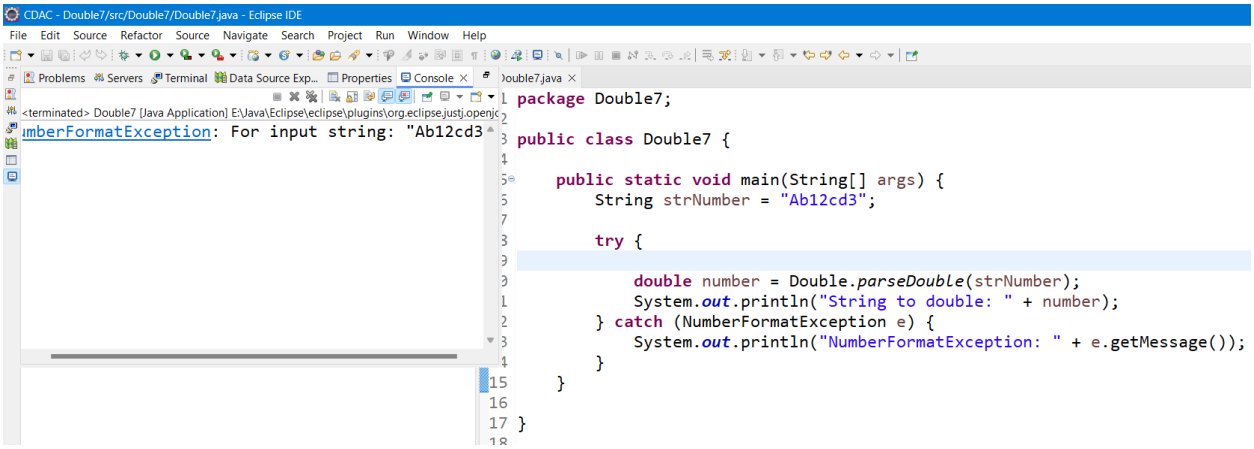
d. Declare a method-local variable number of type double with some value and convert it to a String using the tostring method. (Hint: Use Double.toString(double)).



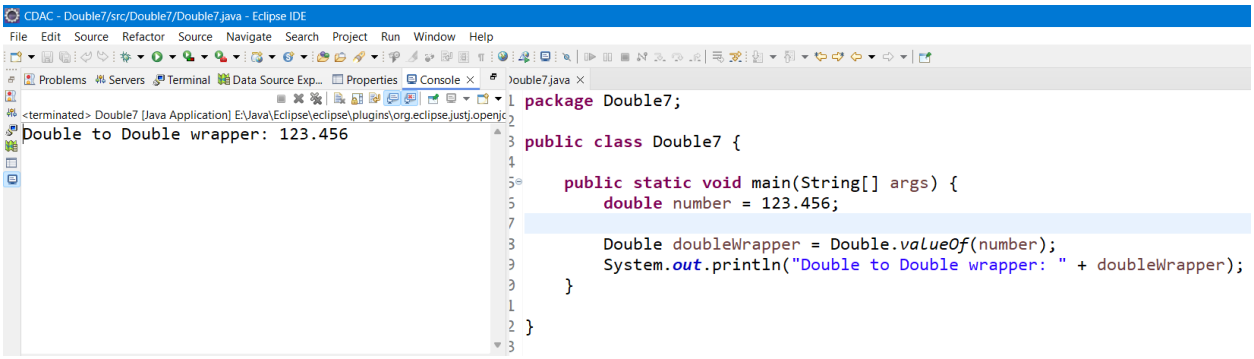
e. Declare a method-local variable strNumber of type string with some value and convert it to a double value using the pazseDouble method. (Hint: Use Double.parseDouble(String)).



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).

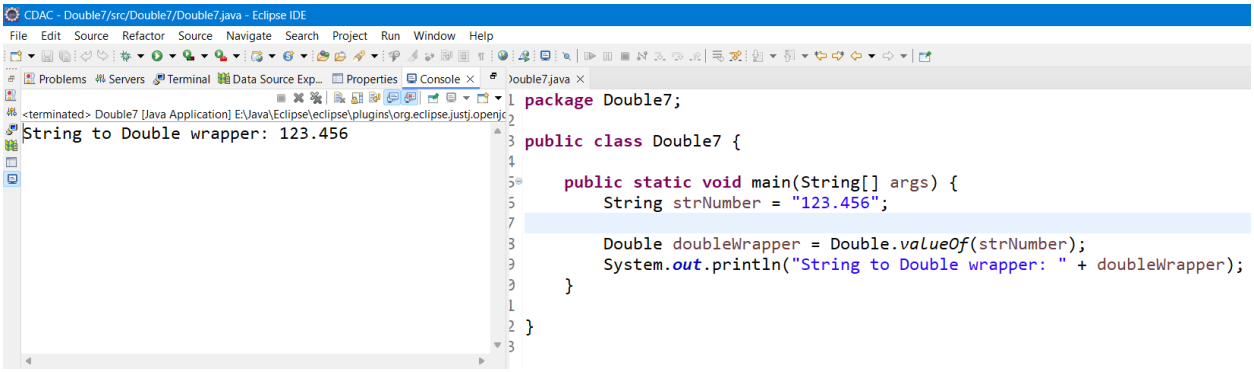


g. Declare a method-local variable number of type double with some value and convert it to the corresponding wrapper class using Double.valueOf(). (Hint: Use Double.valueOf(double)).

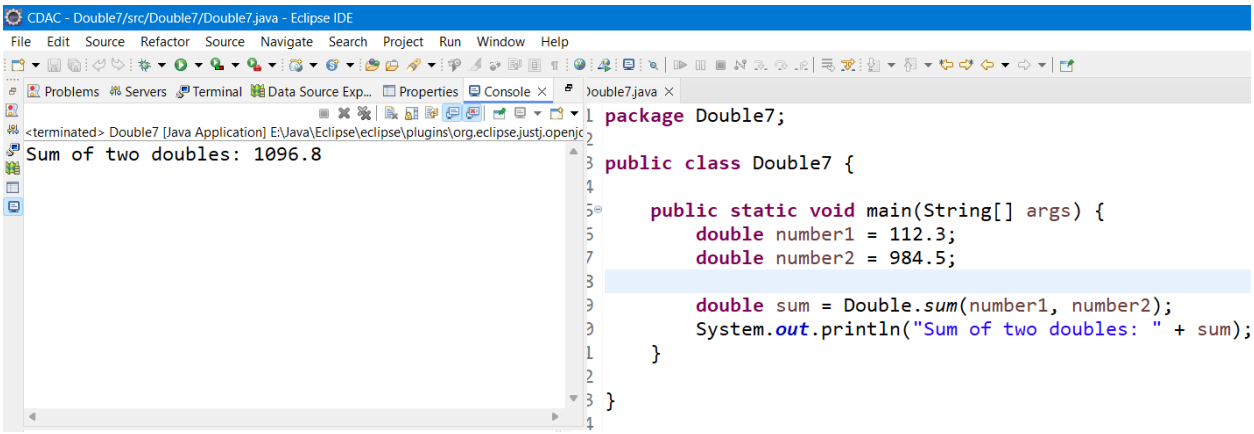


h. Declare a method-local variable strNumber of type String with some double value and convert it to the corresponding wrapper class using Double.

valueOf(). (Hint: Use Double.valueOf(String)).

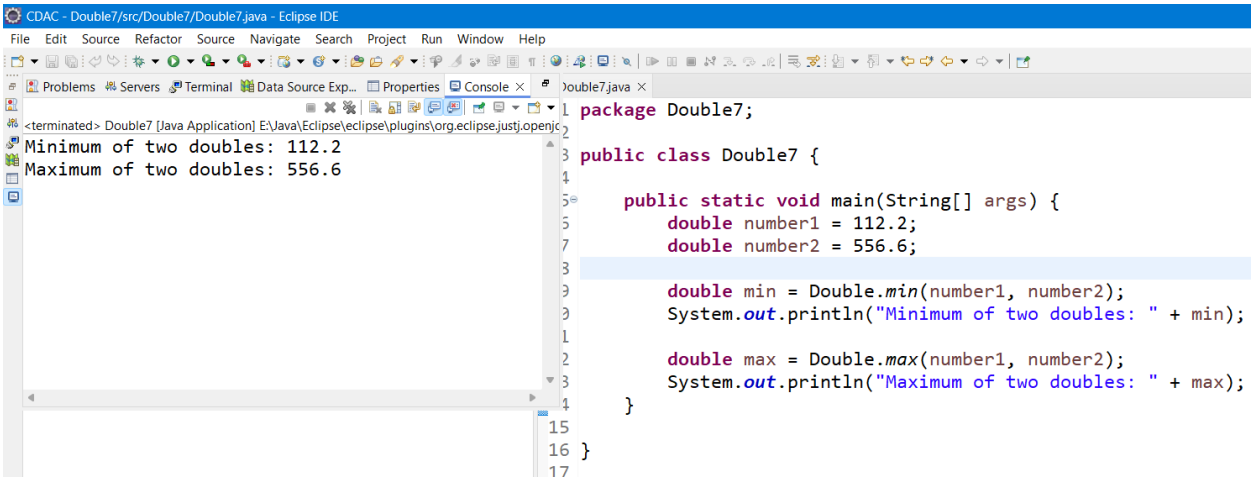


i. Declare two double variables with values 112.3 and 984.5, and add them using a method from the Double class. (Hint: Use Double.sum(double, double)).

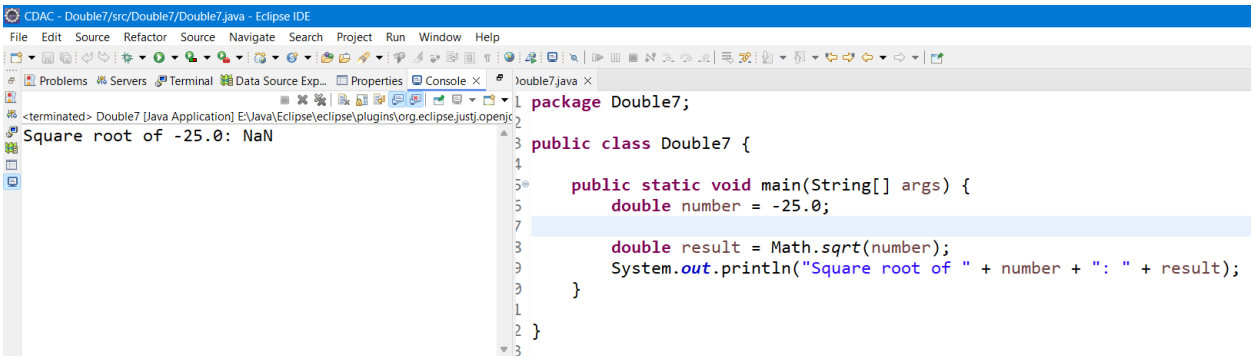


J. Declare two double variables with values 112.2 and 556.6, and find the minimum and maximum values using the Double class. (Hint: Use

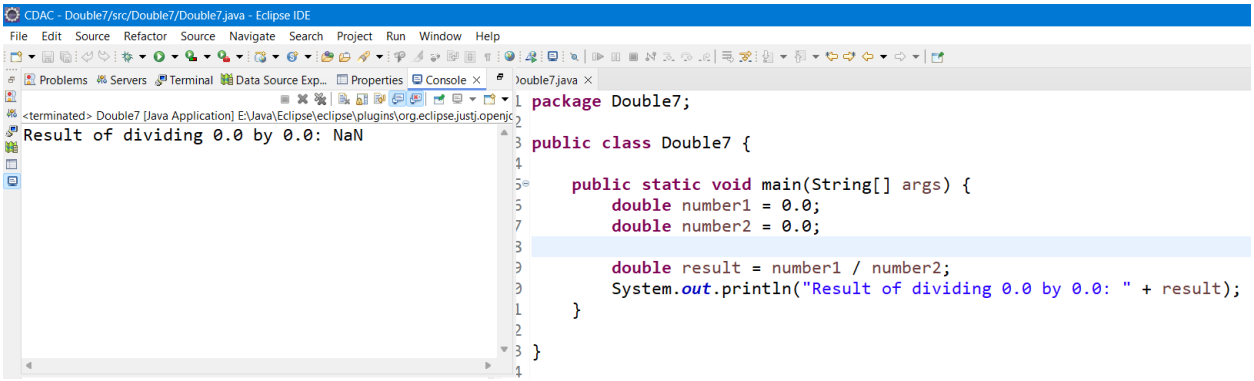
Double.min (double, double) and Double.max (double, double)).



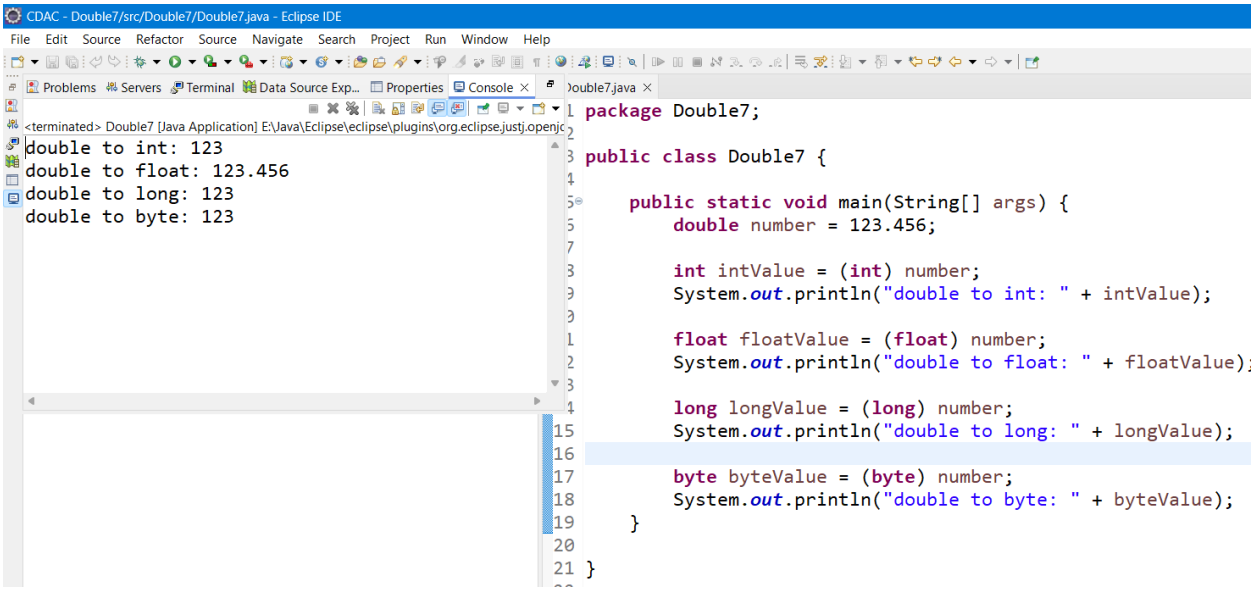
k. Declare a double variable with the value -25.0. Find the square root of this value. (Hint: Use Math.sqrt() method).



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.

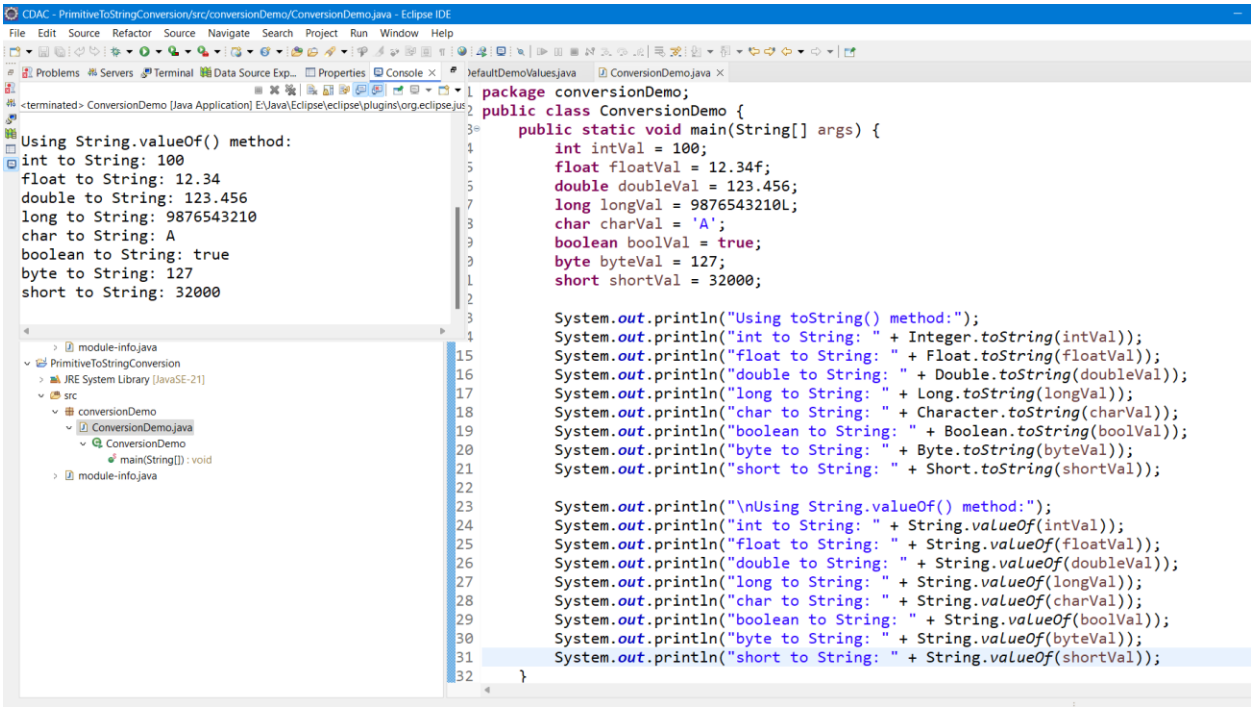


8. Conversion between Primitive Types and Strings

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 value of method of the string class. (e.g., String.valueOf()).



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).

```
package org.example;
public class DefaultValuesDemo {

 // Instance variables (fields)
 int intVal;
 float floatVal;
 double doubleVal;
 long longVal;
 char charVal;
 boolean boolVal;
 byte byteVal;
 short shortVal;
```

```
 static int staticIntVal;
 static float staticFloatVal;
 static double staticDoubleVal;
 static long staticLongVal;
 static char staticCharVal;
 static boolean staticBoolVal;
 static byte staticByteVal;
 static short staticShortVal;

 public static void main(String[] args) {

 DefaultValuesDemo instance = new DefaultValuesDemo();

 System.out.println("Default values of instance variables:");
 System.out.println("int: " + instance.intVal);
 System.out.println("float: " + instance.floatVal);
 System.out.println("double: " + instance.doubleVal);
 System.out.println("long: " + instance.longVal);
 System.out.println("char: [" + instance.charVal + "]");
 System.out.println("boolean: " + instance.boolVal);
 System.out.println("byte: " + instance.byteVal);
 System.out.println("short: " + instance.shortVal);

 System.out.println("\nDefault values of static variables:");
 System.out.println("static int: " + staticIntVal);
 System.out.println("static float: " + staticFloatVal);
 System.out.println("static double: " + staticDoubleVal);
 System.out.println("static long: " + staticLongVal);
 System.out.println("static char: [" + staticCharVal + "]");
 System.out.println("static boolean: " + staticBoolVal);
 System.out.println("static byte: " + staticByteVal);
 System.out.println("static short: " + staticShortVal);
}
}
```

Output:

```
byte: 0
```

```
short: 0

Default values of static variables:
static int: 0
static float: 0.0
static double: 0.0
static long: 0
static char: [
```

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).

```
package commandArthimetic10;
public class CommandArthimetic10 {
public static void main(String[] args) {
if (args.length < 3) {
System.out.println("Please provide two integers and an operator (+, -,
*, /).");
return;
}
try {
int num1 = Integer.parseInt(args[0]);
int num2 = Integer.parseInt(args[1]);
String operator = args[2];
int result = 0;
switch (operator) {
case "+":
result = num1 + num2;
break;
case "-":
result = num1 - num2;
break;
```

```
case "*":
result = num1 * num2;
break;
case "/":
if (num2 != 0) {
result = num1 / num2;
} else {
System.out.println("Division by zero is not allowed.");
return;
}
break;
default:
System.out.println("Invalid operator! Please use +, -, *, or /.");
return;
}
System.out.println("Result: " + result);
} catch (NumberFormatException e) {
System.out.println("Please enter valid integers.");
}
}
}
```

```
Output: Command-Line Input: 12 3 +
Result: 15
```