

1. Working with java.lang.Boolean

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)`).

Ans.

```
class program{

    public static void main(String [] args){

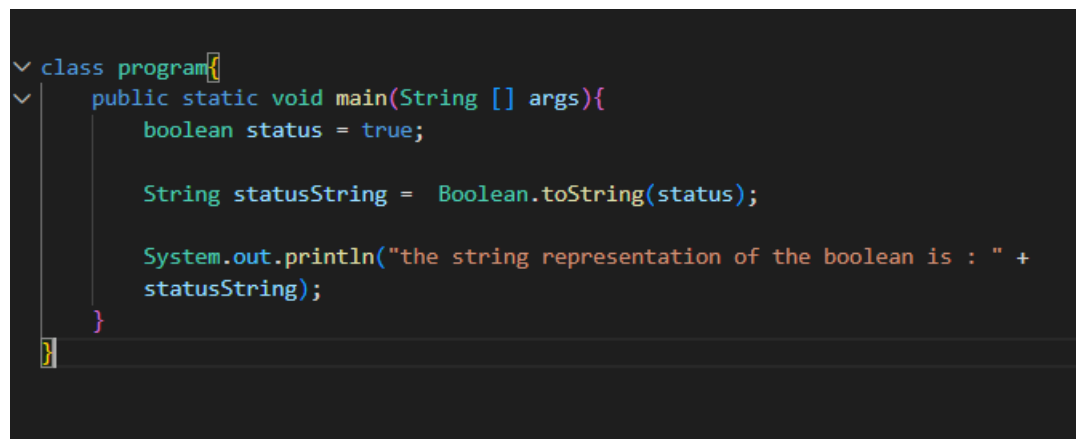
        boolean status = true;


        String statusString = Boolean.toString(status);


        System.out.println("the string representation of the boolean is : " + statusString);

    }

}
```

A screenshot of a code editor with a dark background. The code is written in Java and is color-coded. It shows a class named 'program' with a 'main' method. Inside the 'main' method, a boolean variable 'status' is declared and set to 'true'. Then, a String variable 'statusString' is declared and assigned the value of 'Boolean.toString(status)'. Finally, 'System.out.println' is used to print the string representation of the boolean, which is 'true'.

```
class program{
    public static void main(String [] args){
        boolean status = true;

        String statusString = Boolean.toString(status);

        System.out.println("the string representation of the boolean is : " +
            statusString);
    }
}
```

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)`).

Ans.

```
class program_c{

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

```

String strStatus = "true";

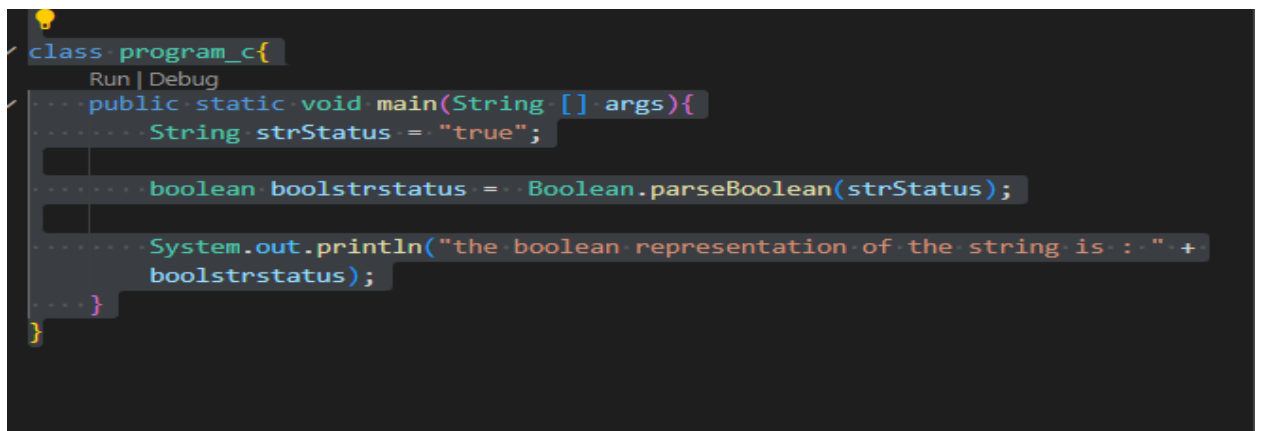
boolean boolstrstatus = Boolean.parseBoolean(strStatus);

System.out.println("the boolean representation of the string is : " + boolstrstatus);

}

}

```



```

class program_c {
    Run | Debug
    public static void main(String [] args) {
        String strStatus = "true";
        boolean boolstrstatus = Boolean.parseBoolean(strStatus);
        System.out.println("the boolean representation of the string is : " +
            boolstrstatus);
    }
}

```

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"`).

```

Ans. class program_d{

    public static void main(String [] args){

        String strStatus = "1"; // or "0"

        boolean status = "1".equals(strStatus);

        System.out.println("the boolean value is : " + status);

    }

}

```

```
ion1 > J d.java > program_d
class program_d{
    Run | Debug
    public static void main(String [] args){
        String strStatus = "1"; // or "0"

        boolean status = "1".equals(strStatus);

        System.out.println("the boolean value is : " + status);
    }
}
```

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)`).

Ans. class `program_e`{

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

```
        boolean status = true;
```

```
        boolean statusWrapper = Boolean.valueOf(status);
```

```
        System.out.println("the boolean value is : " + statusWrapper);
```

```
    }
```

```
}
```

```
class program_e{
    Run | Debug
    public static void main(String [] args){
        boolean status = true;

        boolean statusWrapper = Boolean.valueOf(status);

        System.out.println("the boolean value is : " + statusWrapper);
    }
}
```

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)`).

Ans. class `program_f`{

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

```
        String strStatus = "true";
```

```
        boolean statusWrapper = Boolean.valueOf(strStatus);
```

```
        System.out.println("the boolean value is : " + statusWrapper);
```

```
    }
```

```
}
```

A screenshot of an IDE window showing the code for `program_f.java`. The code is as follows:

```
class program_f {  
    public static void main(String [] args){  
        String strStatus = "true";  
        boolean statusWrapper = Boolean.valueOf(strStatus);  
        System.out.println("the boolean value is : " + statusWrapper);  
    }  
}
```

The IDE has a dark theme. The code is color-coded: `class` is blue, `public static void` is blue, `main` is blue, `String` is blue, `args` is blue, `String` is blue, `strStatus` is blue, `"true"` is red, `boolean` is blue, `statusWrapper` is blue, `Boolean.valueOf` is blue, `strStatus` is blue, `System.out.println` is blue, `"the boolean value is : "` is red, `statusWrapper` is blue, and `}` is blue. There is a lightbulb icon on the left side of the code editor.

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

Ans. class `program_g`{

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

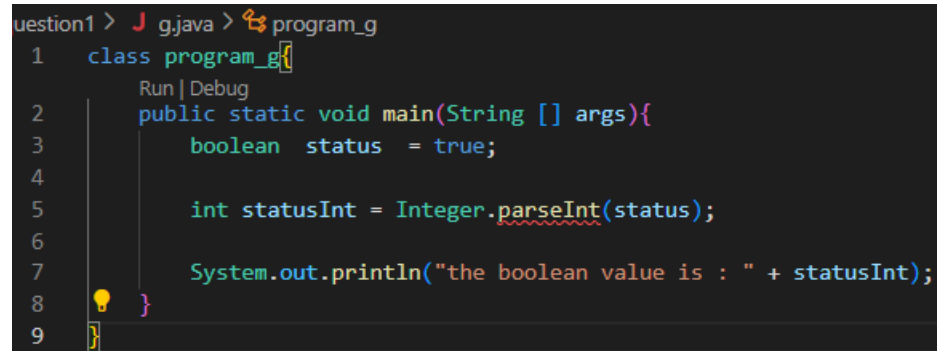
```
        boolean status = true;
```

```
        int statusInt = Integer.parseInt(status);
```

```

        System.out.println("the boolean value is : " + statusInt);
    }
}

```



```

question1 > J g.java > program_g
1  class program_g{
    Run | Debug
2      public static void main(String [] args){
3          boolean status = true;
4
5          int statusInt = Integer.parseInt(status);
6
7          System.out.println("the boolean value is : " + statusInt);
8      }
9  }

```

2. Working with java.lang.Byte

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

Ans. class `byteSizeTest`{

```

    public static void main(String [] args){

```

```

        int byteSize = Byte.BYTES;

```

```

        System.out.println("The number of bytes used to represent a byte value is : " + byteSize);

```

```

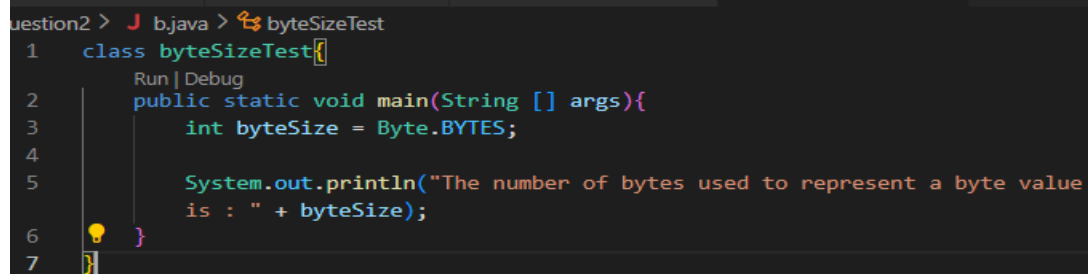
    }

```

```

}

```



```

question2 > J b.java > byteSizeTest
1  class byteSizeTest{
    Run | Debug
2      public static void main(String [] args){
3          int byteSize = Byte.BYTES;
4
5          System.out.println("The number of bytes used to represent a byte value
is : " + byteSize);
6      }
7  }

```

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).

Ans. class min_max_byte{

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

```
    int maxi = Byte.MAX_VALUE;
```

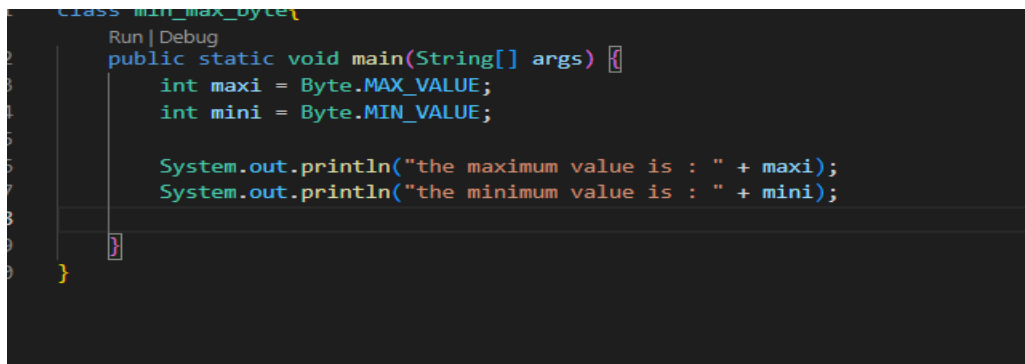
```
    int mini = Byte.MIN_VALUE;
```

```
    System.out.println("the maximum value is : " + maxi);
```

```
    System.out.println("the minimum value is : " + mini);
```

```
}
```

```
}
```

A screenshot of an IDE window showing a Java class named min_max_byte. The code inside the main method initializes two integer variables, maxi and mini, with Byte.MAX_VALUE and Byte.MIN_VALUE respectively. It then prints these values to the console. The IDE interface includes a 'Run | Debug' button at the top and a line number margin on the left.

```
class min_max_byte{  
    public static void main(String[] args) {  
        int maxi = Byte.MAX_VALUE;  
        int mini = Byte.MIN_VALUE;  
  
        System.out.println("the maximum value is : " + maxi);  
        System.out.println("the minimum value is : " + mini);  
    }  
}
```

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)).

Ans. class program{

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

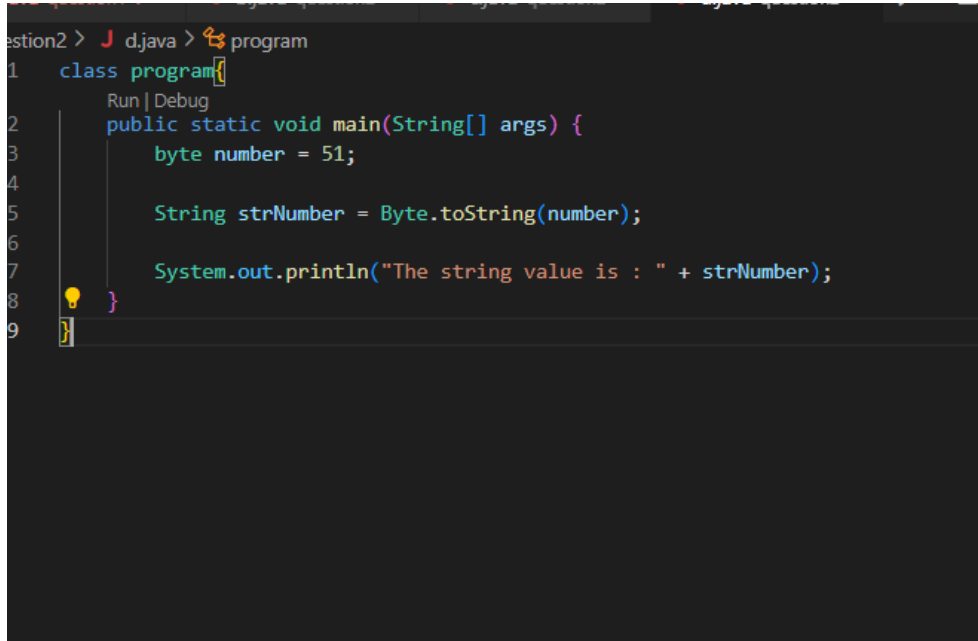
```
    byte number = 51;
```

```
    String strNumber = Byte.toString(number);
```

```

        System.out.println("The string value is : " + strNumber);
    }
}

```



The screenshot shows an IDE window with a Java file named 'program'. The code inside the 'main' method of the 'program' class is as follows:

```

1  class program{
    Run | Debug
2      public static void main(String[] args) {
3          byte number = 51;
4
5          String strNumber = Byte.toString(number);
6
7          System.out.println("The string value is : " + strNumber);
8      }
9  }

```

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)`).

Ans. class `program_e`{

```

    public static void main(String[] args) {

```

```

        String strNumber = "15";

```

```

        byte byteStrNumber = Byte.parseByte(strNumber);

```

```

        System.out.println("the byte value for the strNumber is : "+byteStrNumber);

```

```

    }
}

```

```
1 class program_e {
2     Run | Debug
3     public static void main(String[] args) {
4         String strNumber = "15";
5
6         byte byteStrNumber = Byte.parseByte(strNumber);
7
8         System.out.println("the byte value for the strNumber is : "
9             + byteStrNumber);
10    }
11 }
```

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`).

Ans.

```
class program_f {

    public static void main(String[] args) {

        String strNumber = "Ab12Cd3";

        try{

            byte byteStrNumber = Byte.parseByte(strNumber);

            System.out.println("The byte value is : " + byteStrNumber);

        }

        catch(NumberFormatException e){

            System.out.println("Error: the string is not a valid byte value.");

        }

    }

}
```



```
1  class program_f{
2      Run | Debug
3      public static void main(String[] args) {
4          String strNumber = "Ab12Cd3";
5
6          try{
7              byte byteStrNumber = Byte.parseByte(strNumber);
8              System.out.println("The byte value is : " + byteStrNumber);
9          }
10         catch(NumberFormatException e){
11
12             System.out.println(x:"Error: the string is not a valid byte value.
13             ");
14         }
15     }
16 }
```

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)`).

Ans. class progrma_g{

public static void main(String[] args) {

byte number = 51;

Byte byteObject = Byte.valueOf(number);

System.out.println("The byte object is: " + byteObject);

}

}

```
question2 > J g.java > progrma_g
1  class progrma_g{
2      Run | Debug
3      public static void main(String[] args) {
4          byte number = 51;
5
6          Byte byteObject = Byte.valueOf(number);
7
8          System.out.println("The byte object is: " + byteObject);
9      }
10 }
```

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)`).

Ans. class `program_h`{

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

```
    String strNumber = "300";
```

```
    try {
```

```
        Byte byteObject = Byte.valueOf(strNumber);
```

```
        System.out.println("The Byte object is : "+ byteObject);
```

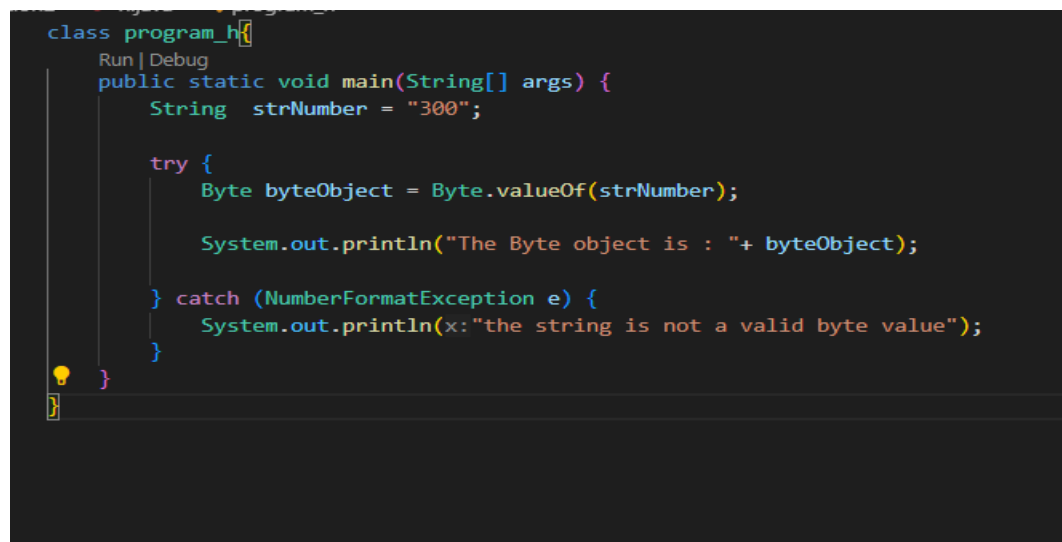
```
    } catch (NumberFormatException e) {
```

```
        System.out.println("the string is not a valid byte value");
```

```
    }
```

```
}
```

```
}
```

A screenshot of an IDE window showing the Java code for the `program_h` class. The code is as follows:

```
class program_h {  
    Run | Debug  
    public static void main(String[] args) {  
        String strNumber = "300";  
  
        try {  
            Byte byteObject = Byte.valueOf(strNumber);  
  
            System.out.println("The Byte object is : "+ byteObject);  
        } catch (NumberFormatException e) {  
            System.out.println(x:"the string is not a valid byte value");  
        }  
    }  
}
```

The code is displayed with syntax highlighting. The IDE interface includes a "Run | Debug" button at the top left of the code editor.

3. Working with java.lang.Short

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

Ans. package question3;

```
class program_b{

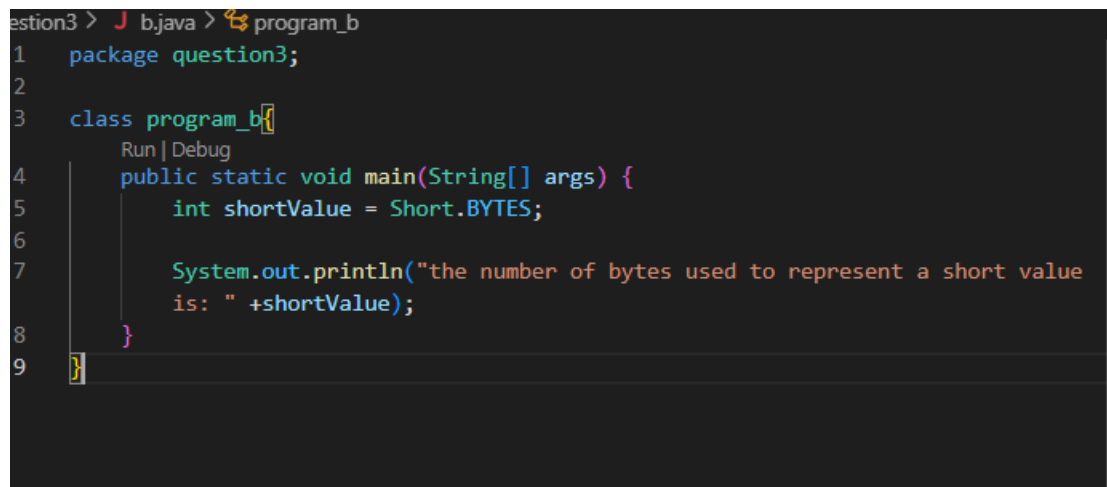
    public static void main(String[] args) {

        int shortValue = Short.BYTES;

        System.out.println("the number of bytes used to represent a short value is: " +shortValue);

    }

}
```



```
question3 > J b.java > program_b
1  package question3;
2
3  class program_b{
4      public static void main(String[] args) {
5          int shortValue = Short.BYTES;
6
7          System.out.println("the number of bytes used to represent a short value
8          is: " +shortValue);
9      }
10 }
```

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).

Ans.

```
class program_c{

    public static void main(String[] args) {

        int maxi = Short.MAX_VALUE;

        int mini = Short.MIN_VALUE;
```

```

        System.out.println(maxi);

        System.out.println(mini);
    }
}

```



```

1
2
3 class program_c{
4     Run | Debug
5     public static void main(String[] args) {
6         int maxi = Short.MAX_VALUE;
7         int mini = Short.MIN_VALUE;
8
9         System.out.println(maxi);
10        System.out.println(mini);
11    }
}

```

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)`).

Ans. package question3;

```

class program_d{

    public static void main(String[] args) {

        short number = 123;

        String strNumber = Short.toString(number);

        System.out.println(strNumber);

    }

}

```

```
package question3;

class program_d{
    Run | Debug
    public static void main(String[] args) {
        short number = 123;

        String strNumber = Short.toString(number);

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

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)`).

Ans. class program_e{

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

        String strNumber = "12345";

        try {

            short number = Short.parseShort(strNumber);

            System.out.println("The short value is : " + number);

        } catch (Exception e) {

            System.out.println("Error : The string is not a valid short value ");

        }

    }

}
```

```
class program_e{
    Run | Debug
    public static void main(String[] args) {
        String strNumber = "12345";

        try {
            short number = Short.parseShort(strNumber);
            System.out.println("The short value is : " +number);
        } catch (Exception e) {
            System.out.println(x:"Error : The string is not a valid short value ");
        }
    }
}
```

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`).

Ans. class `program_f`{

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

`String strNumber` = `"Ab12Cd3"`;

try{

`short number` = `Short.parseShort(strNumber)`;

`System.out.println`("the short value is : "+`number`);

}

catch(`NumberFormatException e`){

`System.out.println`("Error: the string is not a valid short value");

}

}

}

```
stion3 > J f.java > program_f
1 class program_f{
  Run | Debug
2   public static void main(String[] args) {
3       String strNumber = "Ab12Cd3";
4
5       try{
6           short number = Short.parseShort(strNumber);
7
8           System.out.println("the short value is : "+number);
9       }
10      catch(NumberFormatException e){
11          System.out.println(x:"Error: the string is not a valid short value");
12      }
13  }
14  }
```

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)`).

Ans. class `program_g`{

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

`short number = 1213;`

`Short shortObj = Short.valueOf(number);`

`System.out.println("the short object is : " +shortObj);`

`}`

`}`

```
uestions3 > J g.java > program_g
1 class program_g{
  Run | Debug
2   public static void main(String[] args) {
3       short number = 1213;
4       Short shortObj = Short.valueOf(number);
5
6       System.out.println("the short object is : " +shortObj);
7   }
8  }
9  }
```

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)`).

Ans. class `program_h`{

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

```
    String strNumber = "123457980";
```

```
    try{
```

```
        Short shortobj = Short.valueOf(strNumber);
```

```
        System.out.println("the short object is : " +shortobj);
```

```
    }
```

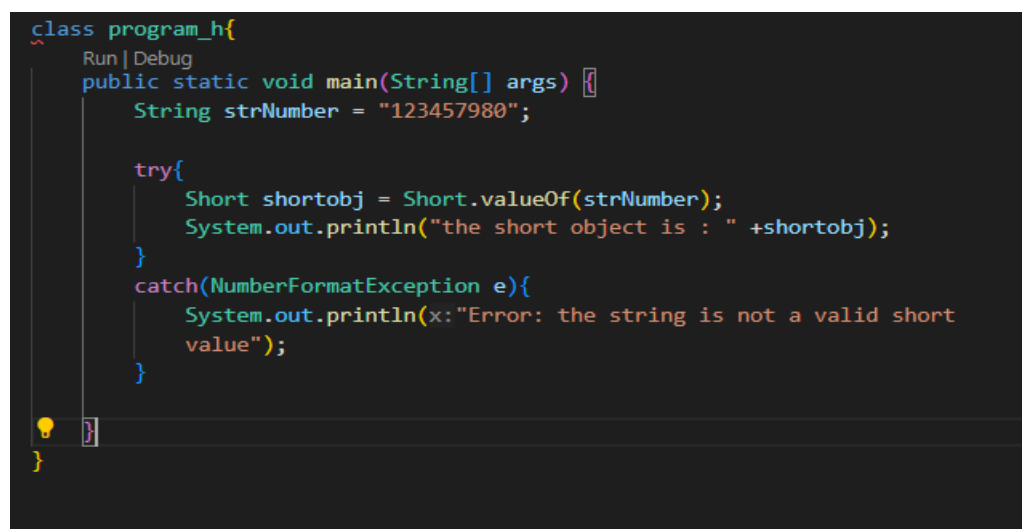
```
    catch(NumberFormatException e){
```

```
        System.out.println("Error: the string is not a valid short value");
```

```
    }
```

```
}
```

```
}
```



```
class program_h{
    Run | Debug
    public static void main(String[] args) {
        String strNumber = "123457980";

        try{
            Short shortobj = Short.valueOf(strNumber);
            System.out.println("the short object is : " +shortobj);
        }
        catch(NumberFormatException e){
            System.out.println(x:"Error: the string is not a valid short
            value");
        }
    }
}
```


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`).

Ans. class `program_b`{

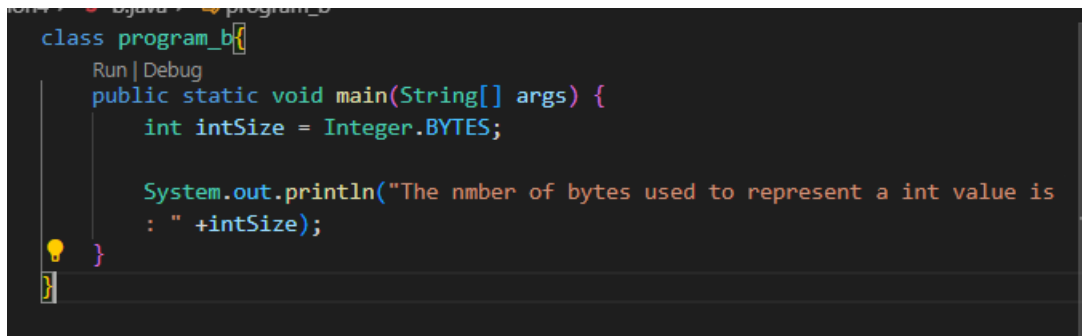
```
public static void main(String[] args) {
```

```
    int intSize = Integer.BYTES;
```

```
    System.out.println("The number of bytes used to represent a int value is : " +intSize);
```

```
}
```

```
}
```



```
class program_b{
    public static void main(String[] args) {
        int intSize = Integer.BYTES;

        System.out.println("The number of bytes used to represent a int value is
        : " +intSize);
    }
}
```

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`).

Ans. class `program_c`{

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

```
    int minValue = Integer.MIN_VALUE;
```

```
    int maxValue = Integer.MAX_VALUE;
```

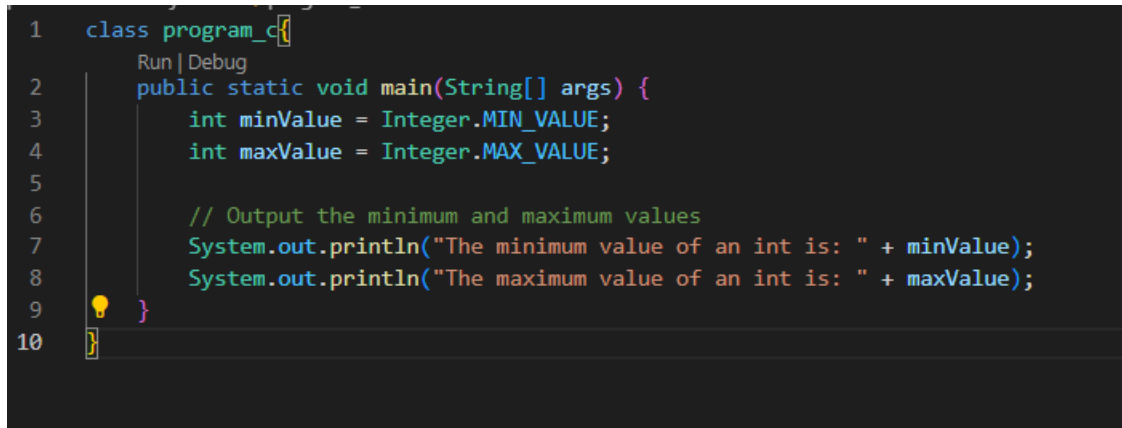
```
    // Output the minimum and maximum values
```

```
    System.out.println("The minimum value of an int is: " + minValue);
```

```

        System.out.println("The maximum value of an int is: " + maxValue);
    }
}

```



```

1  class program_c{
    Run | Debug
2      public static void main(String[] args) {
3          int minValue = Integer.MIN_VALUE;
4          int maxValue = Integer.MAX_VALUE;
5
6          // Output the minimum and maximum values
7          System.out.println("The minimum value of an int is: " + minValue);
8          System.out.println("The maximum value of an int is: " + maxValue);
9      }
10 }

```

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)`).

Ans. class `program_d`{

```

    public static void main(String[] args) {

```

```

        int number = 12345;

```

```

        // Convert the int value to a String using Integer.toString(int)

```

```

        String numberString = Integer.toString(number);

```

```

        // Output the resulting String

```

```

        System.out.println("The int value as a String is: " + numberString);

```

```

    }

```

```

}

```

```
Question4 > J d.java > program_d
1  class program_d{
    Run | Debug
2      public static void main(String[] args) {
3          int number = 12345;
4
5          // Convert the int value to a String using Integer.toString(int)
6          String numberString = Integer.toString(number);
7
8          // Output the resulting String
9          System.out.println("The int value as a String is: " + numberString);
10     }
11 }
```

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)`).

Ans. class `program_e`{

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

```
        String strNumber = "12345";
```

```
        // Convert the String to an int value using Integer.parseInt(String)
```

```
        try {
```

```
            int number = Integer.parseInt(strNumber);
```

```
            System.out.println("The int value is: " + number);
```

```
        } catch (NumberFormatException e) {
```

```
            System.out.println("Error: The String \"" + strNumber + "\" is not a valid int value.");
```

```
        }
```

```
    }
```

```
}
```

```
question4 > J e.java > program_e
1  class program_e{
    Run | Debug
2  public static void main(String[] args) {
3      String strNumber = "12345";
4
5      // Convert the String to an int value using Integer.parseInt(String)
6      try {
7          int number = Integer.parseInt(strNumber);
8          System.out.println("The int value is: " + number);
9      } catch (NumberFormatException e) {
10         System.out.println("Error: The String \"" + strNumber + "\" is not
            a valid int value.");
11     }
12 }
13 }
```

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`).

Ans. class `program_f`{

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

`String strNumber` = "Ab12Cd3";

 // Attempt to convert the `String` to an `int` value

 try {

 int `number` = `Integer.parseInt`(`strNumber`);

`System.out.println`("The int value is: " + `number`);

 } catch (`NumberFormatException` `e`) {

`System.out.println`("Error: The String \"" + `strNumber` + "\" is not a valid int value.");

 }

}

}

```

n4 > J f.java > program_f
class program_f{
  Run | Debug
  public static void main(String[] args) {
    String strNumber = "Ab12Cd3";

    // Attempt to convert the String to an int value
    try {
      int number = Integer.parseInt(strNumber);
      System.out.println("The int value is: " + number);
    } catch (NumberFormatException e) {
      System.out.println("Error: The String \"" + strNumber + "\" is not
        a valid int value.");
    }
  }
}

```

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)`).

Ans. class `program_g`{

```

  public static void main(String[] args) {

```

```

    int number = 12345;

```

```

    // Convert the int value to an Integer object using Integer.valueOf(int)

```

```

    Integer integerObject = Integer.valueOf(number);

```

```

    // Output the resulting Integer object

```

```

    System.out.println("The Integer object is: " + integerObject);

```

```

  }

```

```

}

```

```

1  class program_g{
    Run | Debug
2  public static void main(String[] args) {
3      int number = 12345;
4
5      // Convert the int value to an Integer object using Integer.valueOf(int)
6      Integer integerObject = Integer.valueOf(number);
7
8      // Output the resulting Integer object
9      System.out.println("The Integer object is: " + integerObject);
0  }
1  }

```

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)`).

Ans. class `program_h`{

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

```
    String strNumber = "12345";
```

```
    // Convert the String to an Integer object using Integer.valueOf(String)
```

```
    try {
```

```
        Integer integerObject = Integer.valueOf(strNumber);
```

```
        System.out.println("The Integer object is: " + integerObject);
```

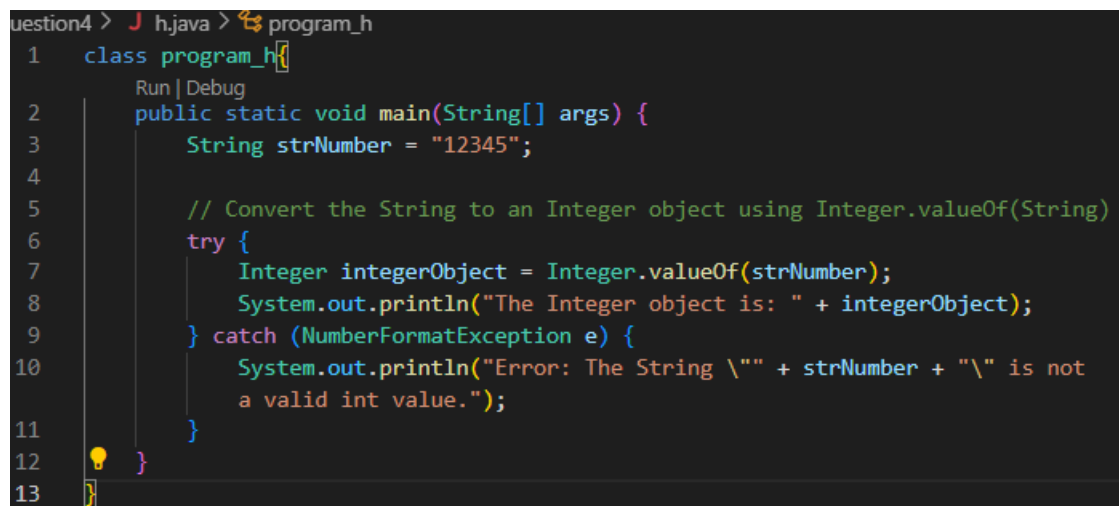
```
    } catch (NumberFormatException e) {
```

```
        System.out.println("Error: The String \"" + strNumber + "\" is not a valid int value.");
```

```
    }
```

```
}
```

```
}
```



```
question4 > h.java > program_h
1  class program_h {
2      Run | Debug
3      public static void main(String[] args) {
4          String strNumber = "12345";
5
6          // Convert the String to an Integer object using Integer.valueOf(String)
7          try {
8              Integer integerObject = Integer.valueOf(strNumber);
9              System.out.println("The Integer object is: " + integerObject);
10         } catch (NumberFormatException e) {
11             System.out.println("Error: The String \"" + strNumber + "\" is not
12             a valid int value.");
13         }
14     }
15 }
```

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)`).

Ans. class `program_i`{

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

    int a = 10;

    int b = 20;


    // Add the two integers using Integer.sum(int, int)

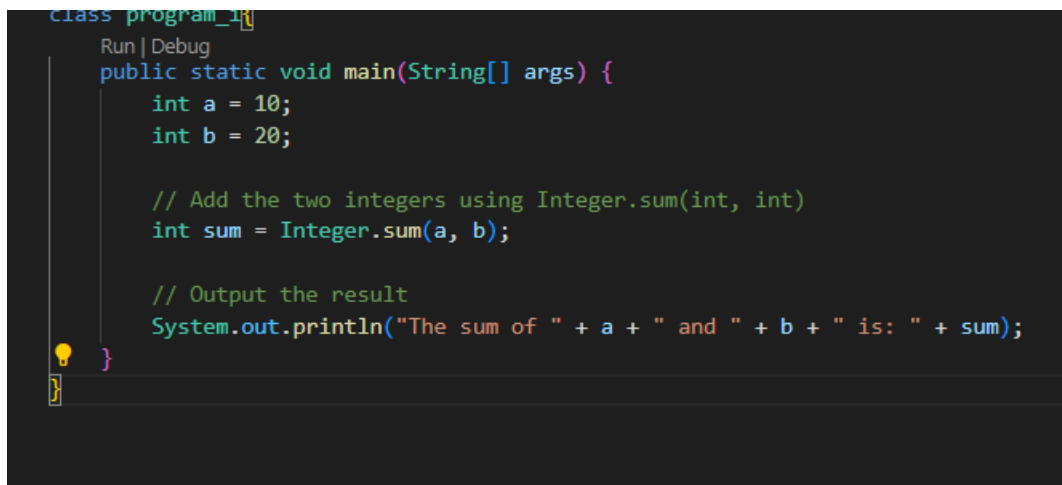
    int sum = Integer.sum(a, b);


    // Output the result

    System.out.println("The sum of " + a + " and " + b + " is: " + sum);

}

}
```

A screenshot of an IDE window titled 'Class program_i'. The code is displayed in a dark-themed editor with syntax highlighting. It shows the same Java code as the previous block: a public static void main method that declares two integers, a and b, with values 10 and 20 respectively. It then uses Integer.sum(a, b) to calculate the sum and prints the result. The code is enclosed in curly braces. On the left side of the editor, there is a lightbulb icon and a yellow bracket icon. The top of the window has a 'Run | Debug' button.

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)`).

Ans. class `program_j`{

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

    int a = 10;

    int b = 20;


    // Find the minimum and maximum values using Integer.min(int, int) and Integer.max(int, int)

    int min = Integer.min(a, b);

    int max = Integer.max(a, b);

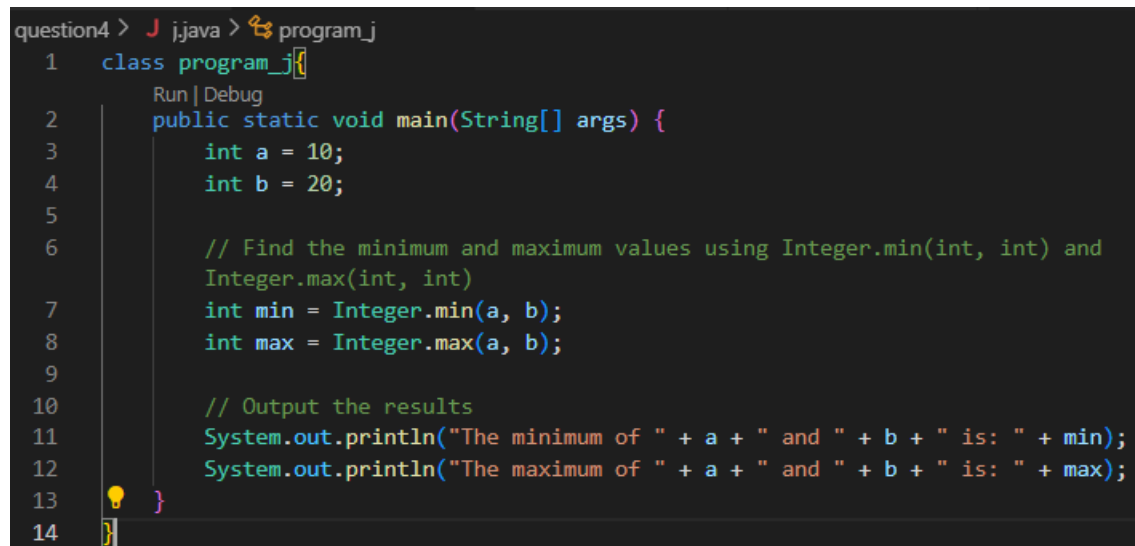

    // Output the results

    System.out.println("The minimum of " + a + " and " + b + " is: " + min);

    System.out.println("The maximum of " + a + " and " + b + " is: " + max);

}

}
```



```
question4 > J java > program_j
1  class program_j {
2      Run | Debug
3      public static void main(String[] args) {
4          int a = 10;
5          int b = 20;
6
7          // Find the minimum and maximum values using Integer.min(int, int) and
8          Integer.max(int, int)
9          int min = Integer.min(a, b);
10         int max = Integer.max(a, b);
11
12         // Output the results
13         System.out.println("The minimum of " + a + " and " + b + " is: " + min);
14         System.out.println("The maximum of " + a + " and " + b + " is: " + max);
15     }
16 }
```


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.toString(int)`, `Integer.toOctalString(int)`, and `Integer.toHexString(int)`).

Ans. class program_k{

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

    int number = 7;

    // Convert the integer to binary, octal, and hexadecimal strings

    String binaryString = Integer.toBinaryString(number);

    String octalString = Integer.toOctalString(number);

    String hexString = Integer.toHexString(number);

    // Output the results

    System.out.println("The binary representation of " + number + " is: " + binaryString);

    System.out.println("The octal representation of " + number + " is: " + octalString);

    System.out.println("The hexadecimal representation of " + number + " is: " + hexString);

}
```

```

question4 > J k.java > program_k
1  class program_k{
    Run | Debug
2  public static void main(String[] args) {
3      int number = 7;
4
5      // Convert the integer to binary, octal, and hexadecimal strings
6      String binaryString = Integer.toBinaryString(number);
7      String octalString = Integer.toOctalString(number);
8      String hexString = Integer.toHexString(number);
9
10     // Output the results
11     System.out.println("The binary representation of " + number + " is: " +
        binaryString);
12     System.out.println("The octal representation of " + number + " is: " +
        octalString);
13     System.out.println("The hexadecimal representation of " + number + "
        is: " + hexString);
14 }
15 }

```

5. Working with java.lang.Long

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

Ans. class program_b{

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

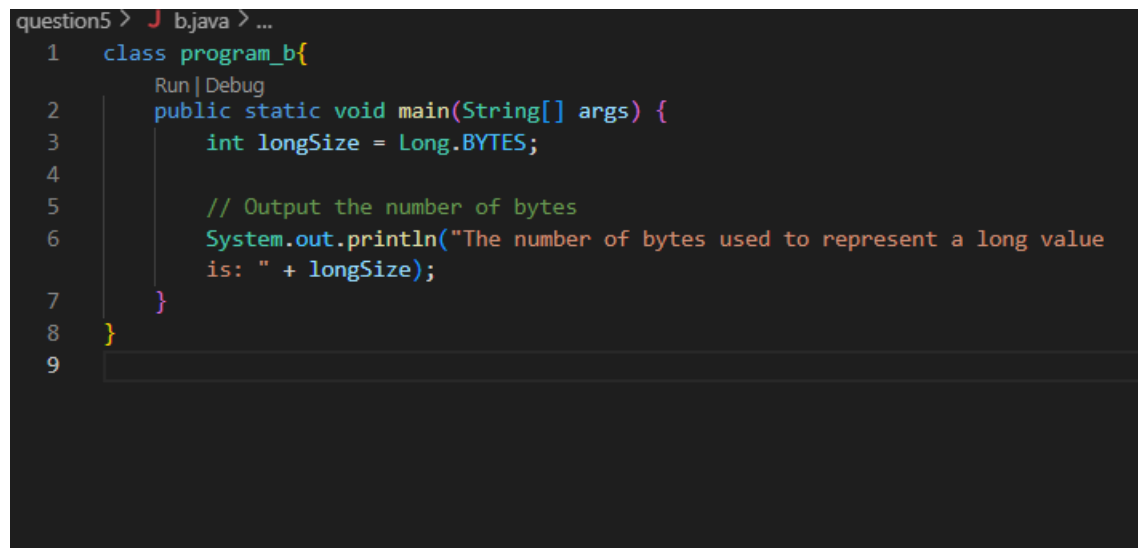
```
    int longSize = Long.BYTES;
```

```
    // Output the number of bytes
```

```
    System.out.println("The number of bytes used to represent a long value is: " + longSize);
```

```
}
```

```
}
```



```
question5 > J b.java > ...
1  class program_b{
2      public static void main(String[] args) {
3          int longSize = Long.BYTES;
4
5          // Output the number of bytes
6          System.out.println("The number of bytes used to represent a long value
7              is: " + longSize);
8      }
9  }
```

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).

Ans. class program_c{

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

```
    long minValue = Long.MIN_VALUE;
```

```

        long maxValue = Long.MAX_VALUE;

        // Output the minimum and maximum values

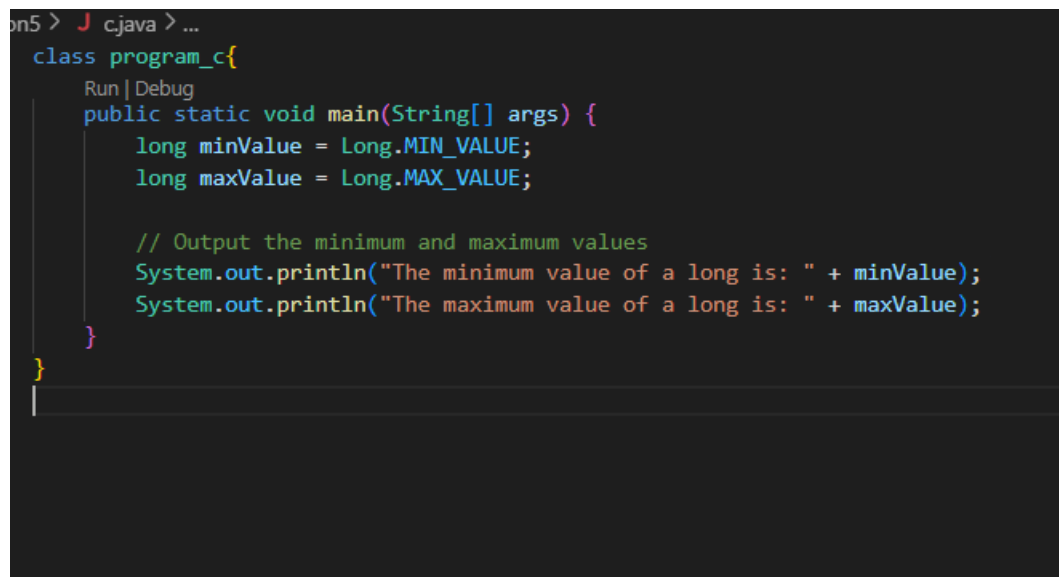
        System.out.println("The minimum value of a long is: " + minValue);

        System.out.println("The maximum value of a long is: " + maxValue);

    }

}

```



```

on5 > J c.java > ...
class program_c{
    Run | Debug
    public static void main(String[] args) {
        long minValue = Long.MIN_VALUE;
        long maxValue = Long.MAX_VALUE;

        // Output the minimum and maximum values
        System.out.println("The minimum value of a long is: " + minValue);
        System.out.println("The maximum value of a long is: " + maxValue);
    }
}

```

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)`).

Ans. class `program_d`{

```

    public static void main(String[] args) {

        long number = 123456789L;

        // Convert the long value to a String using Long.toString(long)

        String numberString = Long.toString(number);
    }
}

```

```

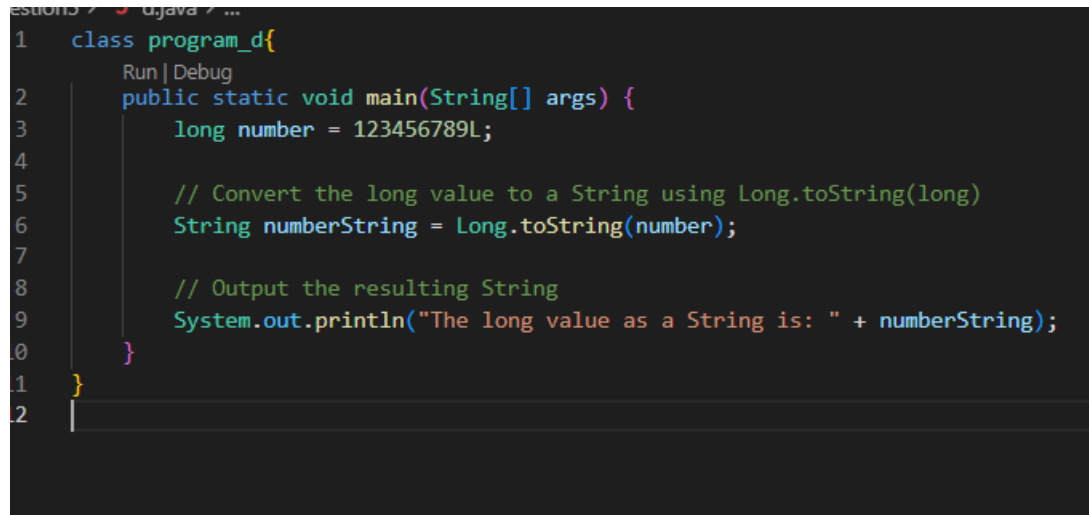
        // Output the resulting String

        System.out.println("The long value as a String is: " + numberString);

    }

}

```



```

1  class program_d{
2      public static void main(String[] args) {
3          long number = 123456789L;
4
5          // Convert the long value to a String using Long.toString(long)
6          String numberString = Long.toString(number);
7
8          // Output the resulting String
9          System.out.println("The long value as a String is: " + numberString);
10     }
11 }
12

```

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)`).

Ans. class `program_e`{

```

    public static void main(String[] args) {

```

```

        String strNumber = "123456789";

```

```

        // Convert the String to a long value using Long.parseLong(String)

```

```

        try {

```

```

            long number = Long.parseLong(strNumber);

```

```

            System.out.println("The long value is: " + number);

```

```

        } catch (NumberFormatException e) {

```

```

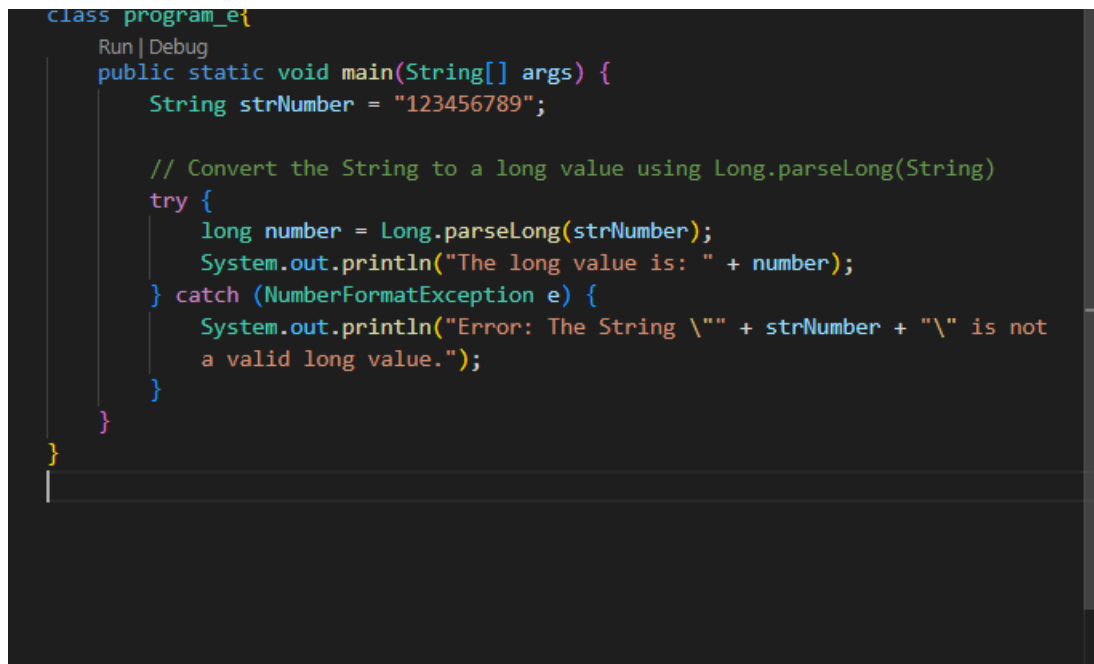
            System.out.println("Error: The String \"" + strNumber + "\" is not a valid long value.");

```

```

    }
}
}

```



```

class program_e{
    Run | Debug
    public static void main(String[] args) {
        String strNumber = "123456789";

        // Convert the String to a long value using Long.parseLong(String)
        try {
            long number = Long.parseLong(strNumber);
            System.out.println("The long value is: " + number);
        } catch (NumberFormatException e) {
            System.out.println("Error: The String \"" + strNumber + "\" is not
            a valid long value.");
        }
    }
}

```

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`).

Ans. class `program_f`{

```

    public static void main(String[] args) {

        String strNumber = "Ab12Cd3";

        // Attempt to convert the String to a long value

        try {

            long number = Long.parseLong(strNumber);

            System.out.println("The long value is: " + number);

        } catch (NumberFormatException e) {

```

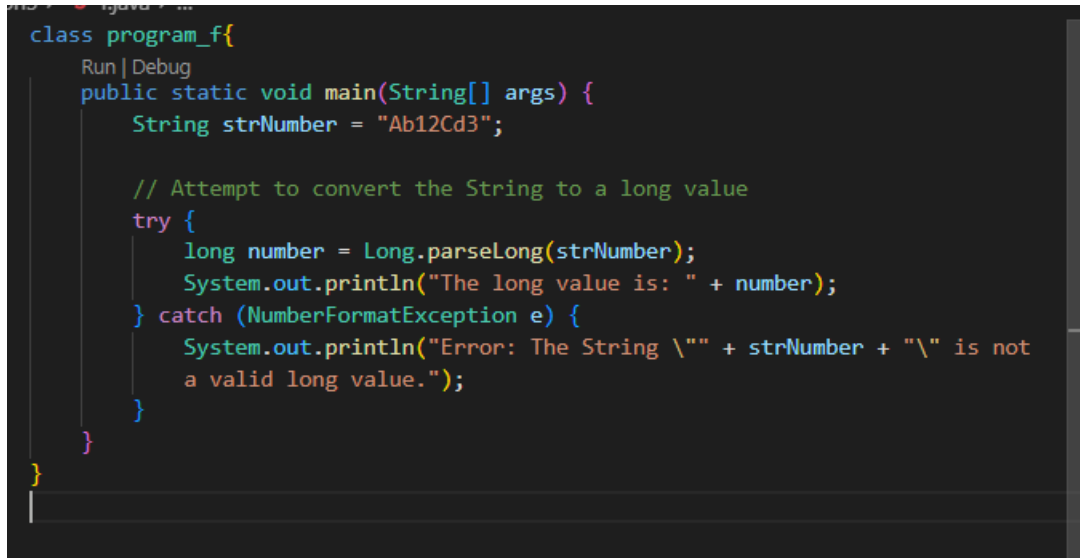
```

        System.out.println("Error: The String \"" + strNumber + "\" is not a valid long value.");
    }

}

}

```



```

class program_f{
    Run | Debug
    public static void main(String[] args) {
        String strNumber = "Ab12Cd3";

        // Attempt to convert the String to a long value
        try {
            long number = Long.parseLong(strNumber);
            System.out.println("The long value is: " + number);
        } catch (NumberFormatException e) {
            System.out.println("Error: The String \"" + strNumber + "\" is not
            a valid long value.");
        }
    }
}

```

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)`).

Ans. class `program_g`{

```

    public static void main(String[] args) {

```

```

        long number = 123456789L;

```

```

        // Convert the long value to a Long object using Long.valueOf(long)

```

```

        Long longObject = Long.valueOf(number);

```

```

        // Output the resulting Long object

```

```

        System.out.println("The Long object is: " + longObject);

```

```

    }

```

```
}
```

```
essions / J gjava / ...
1 class program_g{
  Run | Debug
2 public static void main(String[] args) {
3     long number = 123456789L;
4
5     // Convert the long value to a Long object using Long.valueOf(long)
6     Long longObject = Long.valueOf(number);
7
8     // Output the resulting Long object
9     System.out.println("The Long object is: " + longObject);
0 }
1 }
2
```

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)`).

Ans. class `program_h`{

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

```
    String strNumber = "123456789";
```

```
    // Convert the String to a Long object using Long.valueOf(String)
```

```
    try {
```

```
        Long longObject = Long.valueOf(strNumber);
```

```
        System.out.println("The Long object is: " + longObject);
```

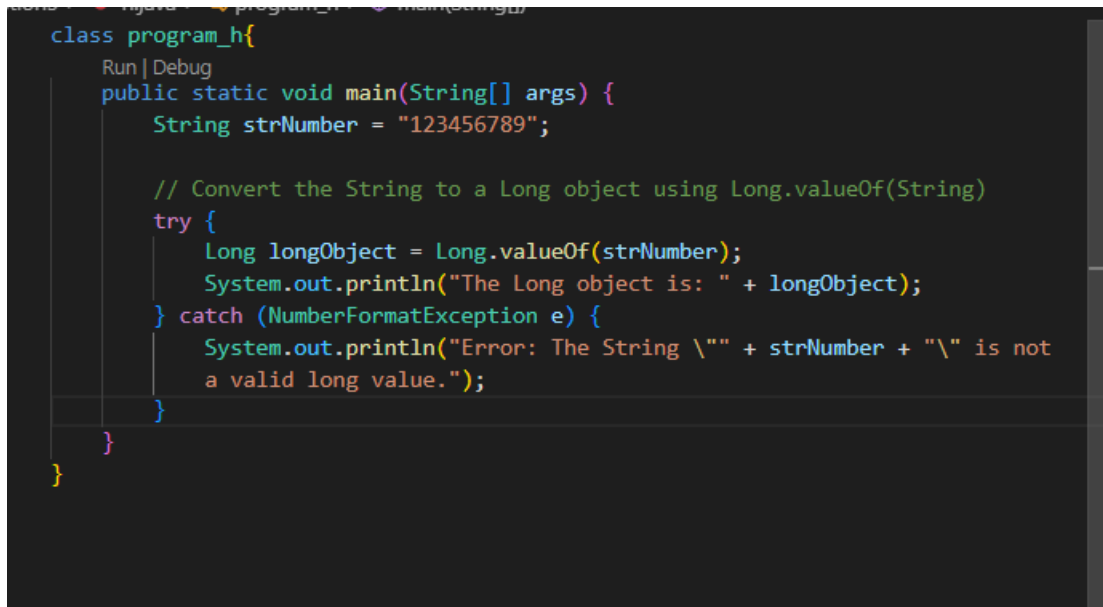
```
    } catch (NumberFormatException e) {
```

```
        System.out.println("Error: The String \"" + strNumber + "\" is not a valid long value.");
```

```
    }
```

```
}
```

```
}
```

A screenshot of an IDE window showing a Java class named 'program_h'. The code defines a 'main' method that takes a 'String[] args' parameter. Inside the method, a 'String strNumber' is initialized with the value '123456789'. A comment indicates the goal is to convert this string to a Long object using 'Long.valueOf(String)'. A 'try' block contains the conversion and a 'println' statement. A 'catch' block for 'NumberFormatException' contains an 'println' statement that outputs an error message if the conversion fails. The code is properly bracketed with curly braces.

```
class program_h{  
    Run | Debug  
    public static void main(String[] args) {  
        String strNumber = "123456789";  
  
        // Convert the String to a Long object using Long.valueOf(String)  
        try {  
            Long longObject = Long.valueOf(strNumber);  
            System.out.println("The Long object is: " + longObject);  
        } catch (NumberFormatException e) {  
            System.out.println("Error: The String \"" + strNumber + "\" is not  
                a valid long value.");  
        }  
    }  
}
```

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)).

Ans. class program_i{

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

```
    long a = 1123L;
```

```
    long b = 9845L;
```

```
    // Add the two long variables using Long.sum(long, long)
```

```
    long sum = Long.sum(a, b);
```

```
    // Output the result
```

```
    System.out.println("The sum of " + a + " and " + b + " is: " + sum);
```

```
}
```


}

```
Question5 > J ijava > program_i
1  class program_i{
2      Run | Debug
3      public static void main(String[] args) {
4          long a = 1123L;
5          long b = 9845L;
6
7          // Add the two long variables using Long.sum(long, long)
8          long sum = Long.sum(a, b);
9
10         // Output the result
11         System.out.println("The sum of " + a + " and " + b + " is: " + sum);
12     }
13 }
```

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)).

Ans. class program_j{

public static void main(String[] args) {

long a = 1122L;

long b = 5566L;

// Find the minimum and maximum values using Long.min(long, long) and Long.max(long, long)

long min = Long.min(a, b);

long max = Long.max(a, b);

// Output the results

System.out.println("The minimum of " + a + " and " + b + " is: " + min);

System.out.println("The maximum of " + a + " and " + b + " is: " + max);

```
}  
  
}
```

```
class program_j{  
    Run | Debug  
    public static void main(String[] args) {  
        long a = 1122L;  
        long b = 5566L;  
  
        // Find the minimum and maximum values using Long.min(long, long) and  
        Long.max(long, long)  
        long min = Long.min(a, b);  
        long max = Long.max(a, b);  
  
        // Output the results  
        System.out.println("The minimum of " + a + " and " + b + " is: " + min);  
        System.out.println("The maximum of " + a + " and " + b + " is: " + max);  
    }  
}
```

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.toString(long)`, `Long.toOctalString(long)`, and `Long.toHexString(long)`).

Ans. class program_k{

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

```
        long number = 7L;
```

```
        // Convert the long variable to binary, octal, and hexadecimal strings
```

```
        String binaryString = Long.toString(number);
```

```
        String octalString = Long.toOctalString(number);
```

```

String hexString = Long.toHexString(number);

// Output the results

System.out.println("The binary representation of " + number + " is: " + binaryString);

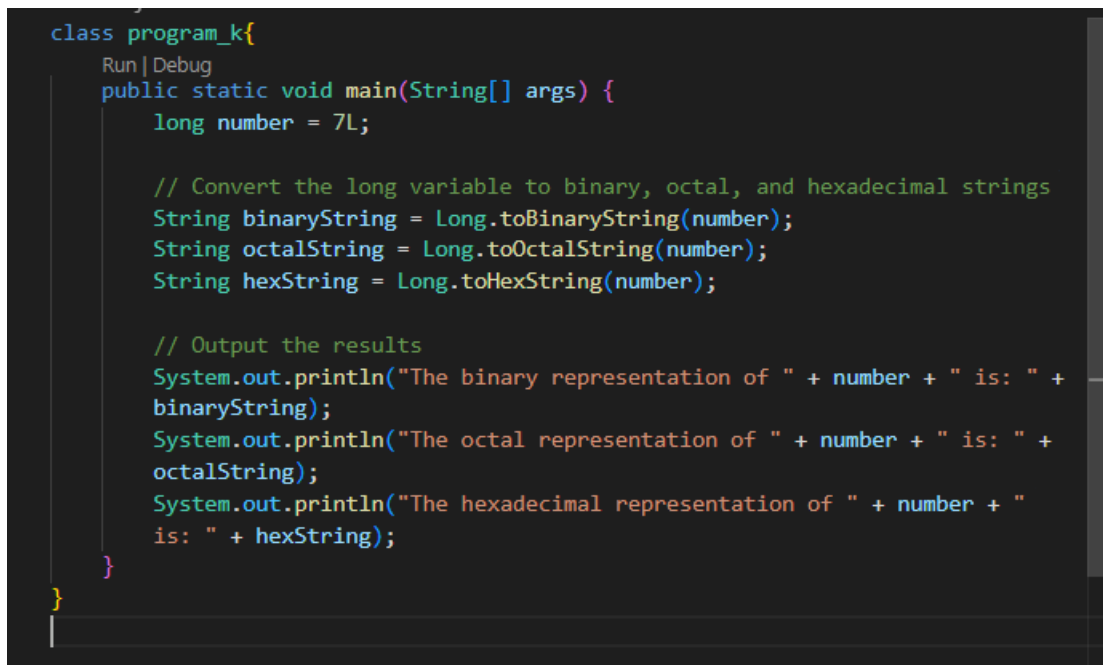
System.out.println("The octal representation of " + number + " is: " + octalString);

System.out.println("The hexadecimal representation of " + number + " is: " + hexString);

}

}

```



```

class program_k{
    Run | Debug
    public static void main(String[] args) {
        long number = 7L;

        // Convert the long variable to binary, octal, and hexadecimal strings
        String binaryString = Long.toBinaryString(number);
        String octalString = Long.toOctalString(number);
        String hexString = Long.toHexString(number);

        // Output the results
        System.out.println("The binary representation of " + number + " is: " +
            binaryString);
        System.out.println("The octal representation of " + number + " is: " +
            octalString);
        System.out.println("The hexadecimal representation of " + number + "
            is: " + hexString);
    }
}

```

6. Working with java.lang.Float

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

Ans. class program_b{

```

public static void main(String[] args) {

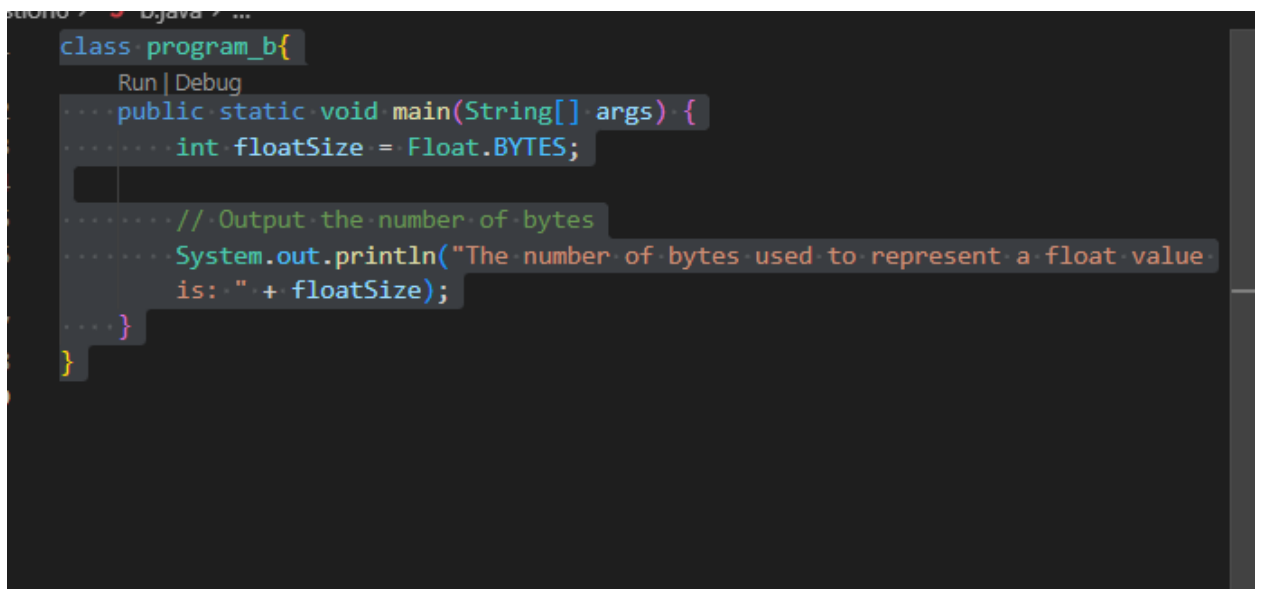
    int floatSize = Float.BYTES;

    // Output the number of bytes

    System.out.println("The number of bytes used to represent a float value is: " + floatSize);

}
}

```



```

class program_b{
    Run | Debug
    public static void main(String[] args) {
        int floatSize = Float.BYTES;

        // Output the number of bytes
        System.out.println("The number of bytes used to represent a float value
        is: " + floatSize);
    }
}

```

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).

Ans. class program_c{

```

public static void main(String[] args) {

    float minValue = Float.MIN_VALUE;

    float maxValue = Float.MAX_VALUE;

    // Output the minimum and maximum values

```

```

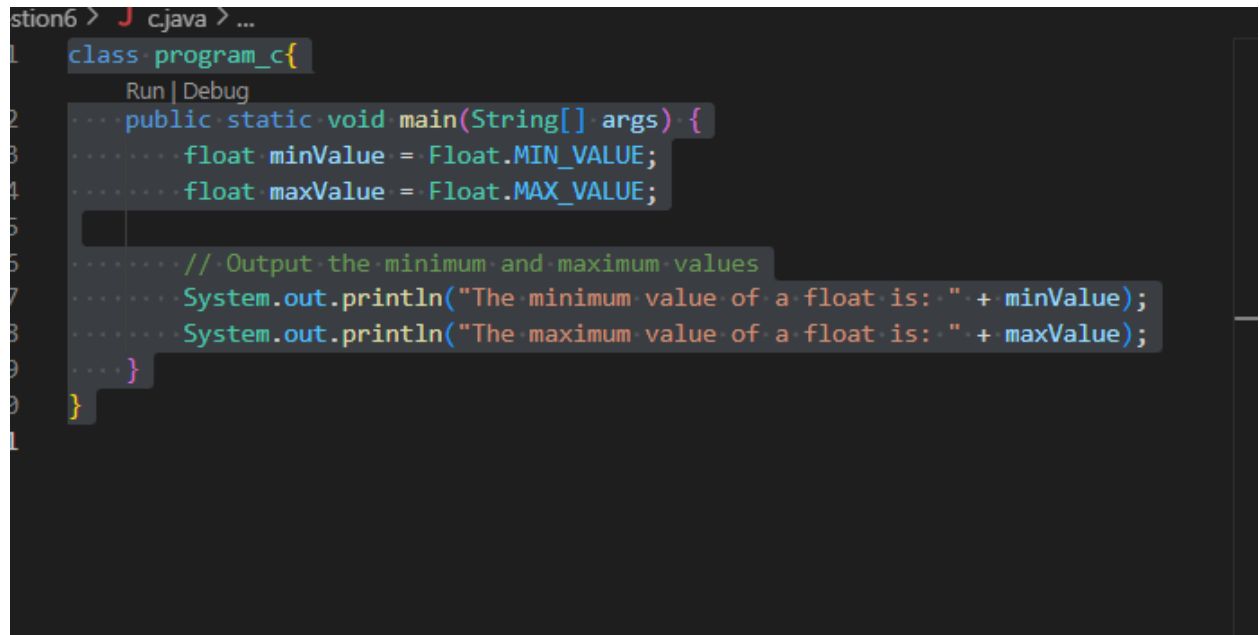
        System.out.println("The minimum value of a float is: " + minValue);

        System.out.println("The maximum value of a float is: " + maxValue);

    }

}

```



```

stion6 > J cjava > ...
1  class program_c{
    Run | Debug
2  ... public static void main(String[] args) {
3  ...     float minValue = Float.MIN_VALUE;
4  ...     float maxValue = Float.MAX_VALUE;
5
6  ...     // Output the minimum and maximum values
7  ...     System.out.println("The minimum value of a float is: " + minValue);
8  ...     System.out.println("The maximum value of a float is: " + maxValue);
9  ... }
10 }

```

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)`).

Ans. class `program_d`{

```

public static void main(String[] args) {

    float number = 123.456f;

    // Convert the float value to a String using Float.toString(float)

    String numberString = Float.toString(number);

    // Output the resulting String

    System.out.println("The float value as a String is: " + numberString);
}

```

```
}  
  
}
```

```
class program_d {  
    Run | Debug  
    public static void main(String[] args) {  
        float number = 123.456f;  
  
        // Convert the float value to a String using Float.toString(float)  
        String numberString = Float.toString(number);  
  
        // Output the resulting String  
        System.out.println("The float value as a String is: " + numberString);  
    }  
}
```

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)`).

Ans. class program_e{

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

```
    String strNumber = "123.456";
```

```
    // Convert the String to a float value using Float.parseFloat(String)
```

```
    try {
```

```
        float number = Float.parseFloat(strNumber);
```

```
        System.out.println("The float value is: " + number);
```

```
    } catch (NumberFormatException e) {
```

```
        System.out.println("Error: The String \"" + strNumber + "\" is not a valid float value.");
```

```
    }
```

```
}  
  
}
```

```
Run | Debug  
class program_e{  
    public static void main(String[] args) {  
        String strNumber = "123.456";  
  
        // Convert the String to a float value using Float.parseFloat(String)  
        try {  
            float number = Float.parseFloat(strNumber);  
            System.out.println("The float value is: " + number);  
        } catch (NumberFormatException e) {  
            System.out.println("Error: The String \"" + strNumber + "\" is not  
                a valid float value.");  
        }  
    }  
}
```

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`).

Ans. `class program_f{`

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

 `String strNumber = "Ab12Cd3";`

 `// Attempt to convert the String to a float value`

 `try {`

 `float number = Float.parseFloat(strNumber);`

 `System.out.println("The float value is: " + number);`

 `} catch (NumberFormatException e) {`

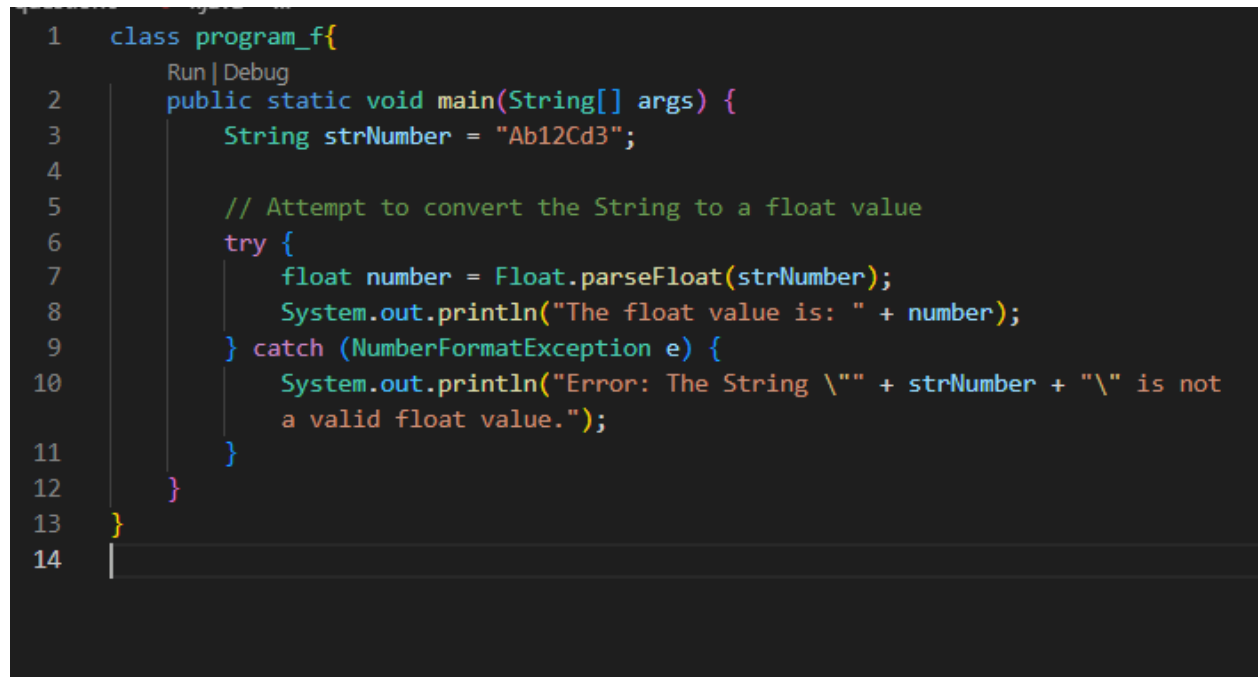
```

        System.out.println("Error: The String \"" + strNumber + "\" is not a valid float value.");
    }

}

}

```



```

1  class program_f{
    Run | Debug
2      public static void main(String[] args) {
3          String strNumber = "Ab12Cd3";
4
5          // Attempt to convert the String to a float value
6          try {
7              float number = Float.parseFloat(strNumber);
8              System.out.println("The float value is: " + number);
9          } catch (NumberFormatException e) {
10             System.out.println("Error: The String \"" + strNumber + "\" is not
                a valid float value.");
11         }
12     }
13 }
14

```

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)`).

Ans. class `program_g`{

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

```
    float number = 123.456f;
```

```
    // Convert the float value to a Float object using Float.valueOf(float)
```

```
    Float floatObject = Float.valueOf(number);
```

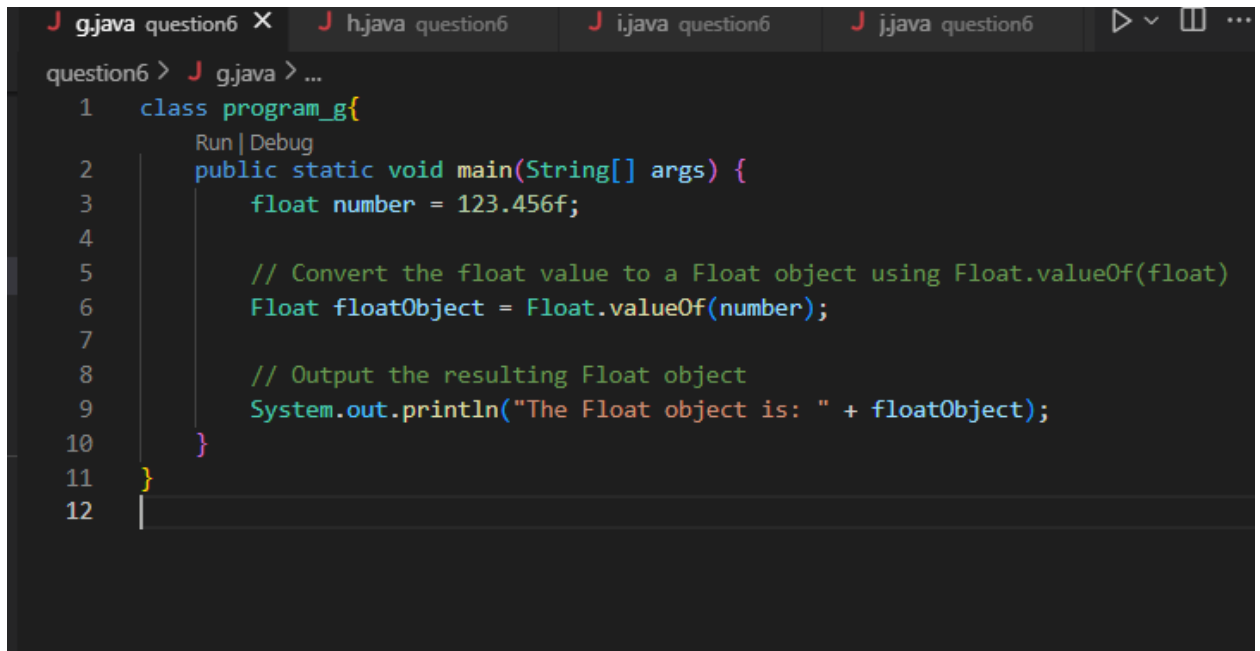
```
    // Output the resulting Float object
```



```

        System.out.println("The Float object is: " + floatObject);
    }
}

```



The screenshot shows an IDE with four tabs: g.java question6, h.java question6, i.java question6, and j.java question6. The active tab is g.java question6. The code in the editor is as follows:

```

question6 > J g.java > ...
1  class program_g{
    Run | Debug
2      public static void main(String[] args) {
3          float number = 123.456f;
4
5          // Convert the float value to a Float object using Float.valueOf(float)
6          Float floatObject = Float.valueOf(number);
7
8          // Output the resulting Float object
9          System.out.println("The Float object is: " + floatObject);
10     }
11 }
12

```

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)`).

Ans. class `program_h`{

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

```
    String strNumber = "123.456";
```

```
    // Convert the String to a Float object using Float.valueOf(String)
```

```
    try {
```

```
        Float floatObject = Float.valueOf(strNumber);
```

```
        System.out.println("The Float object is: " + floatObject);
```

```
    } catch (NumberFormatException e) {
```

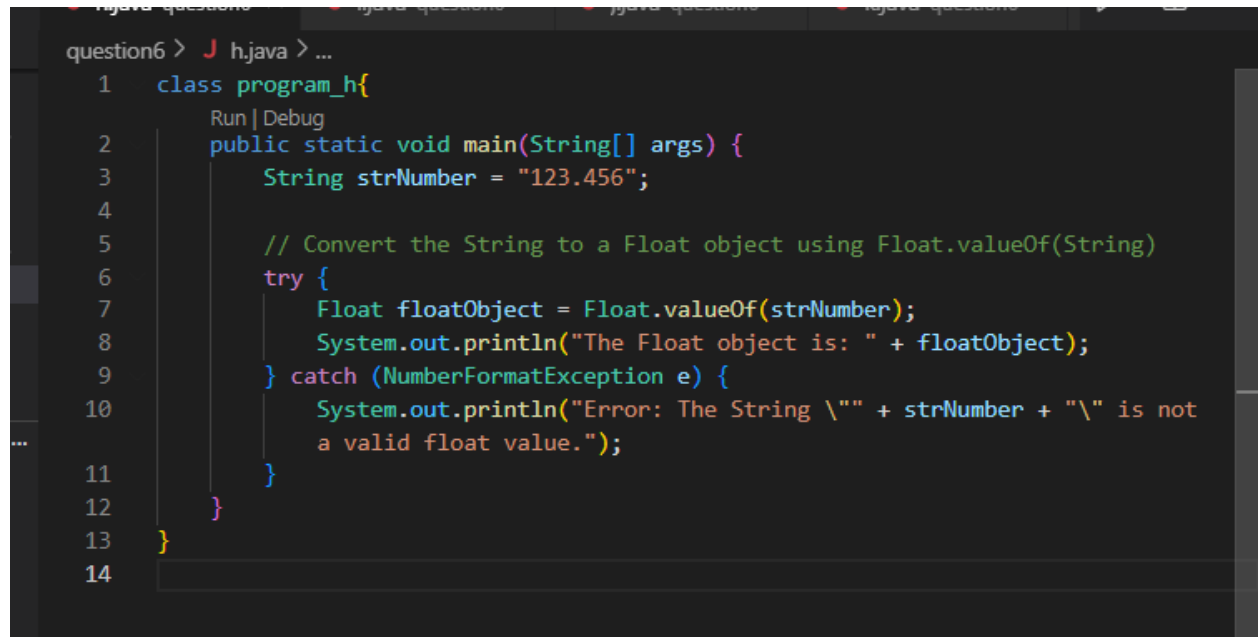
```

        System.out.println("Error: The String \"" + strNumber + "\" is not a valid float value.");
    }

}

}

```



```

question6 > J h.java > ...
1  class program_h{
    Run | Debug
2  public static void main(String[] args) {
3      String strNumber = "123.456";
4
5      // Convert the String to a Float object using Float.valueOf(String)
6      try {
7          Float floatObject = Float.valueOf(strNumber);
8          System.out.println("The Float object is: " + floatObject);
9      } catch (NumberFormatException e) {
10         System.out.println("Error: The String \"" + strNumber + "\" is not
...         a valid float value.");
11     }
12 }
13 }
14

```

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)`).

Ans. class program_i{

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

```
    float a = 112.3f;
```

```
    float b = 984.5f;
```

```
    // Add the two float variables using Float.sum(float, float)
```

```
    float sum = Float.sum(a, b);
```

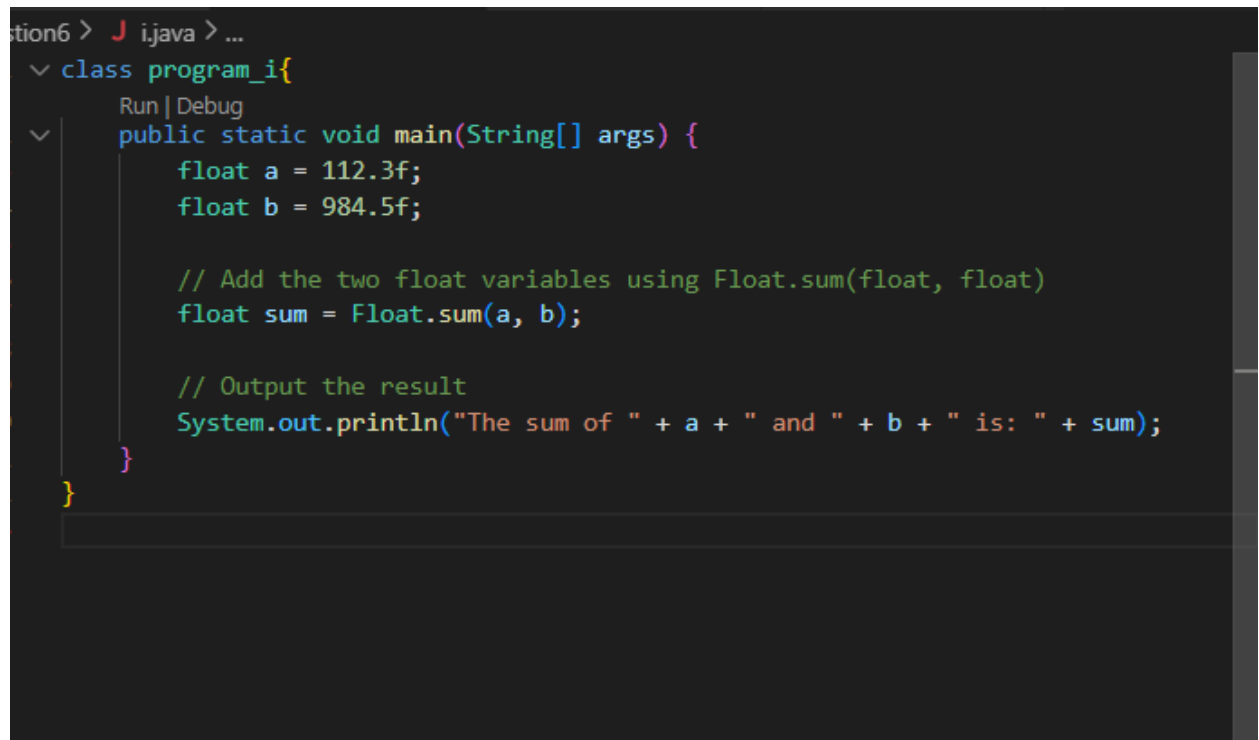
```
    // Output the result

```

```

        System.out.println("The sum of " + a + " and " + b + " is: " + sum);
    }
}

```



```

tion6 > J ijava > ...
  v class program_i{
    Run | Debug
  v public static void main(String[] args) {
      float a = 112.3f;
      float b = 984.5f;

      // Add the two float variables using Float.sum(float, float)
      float sum = Float.sum(a, b);

      // Output the result
      System.out.println("The sum of " + a + " and " + b + " is: " + sum);
    }
  }

```

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)`).

Ans. class program_j{

```

    public static void main(String[] args) {

        float a = 112.2f;

        float b = 556.6f;

        // Find the minimum and maximum values using Float.min(float, float) and Float.max(float, float)

        float min = Float.min(a, b);

        float max = Float.max(a, b);
    }
}

```

```

        // Output the results

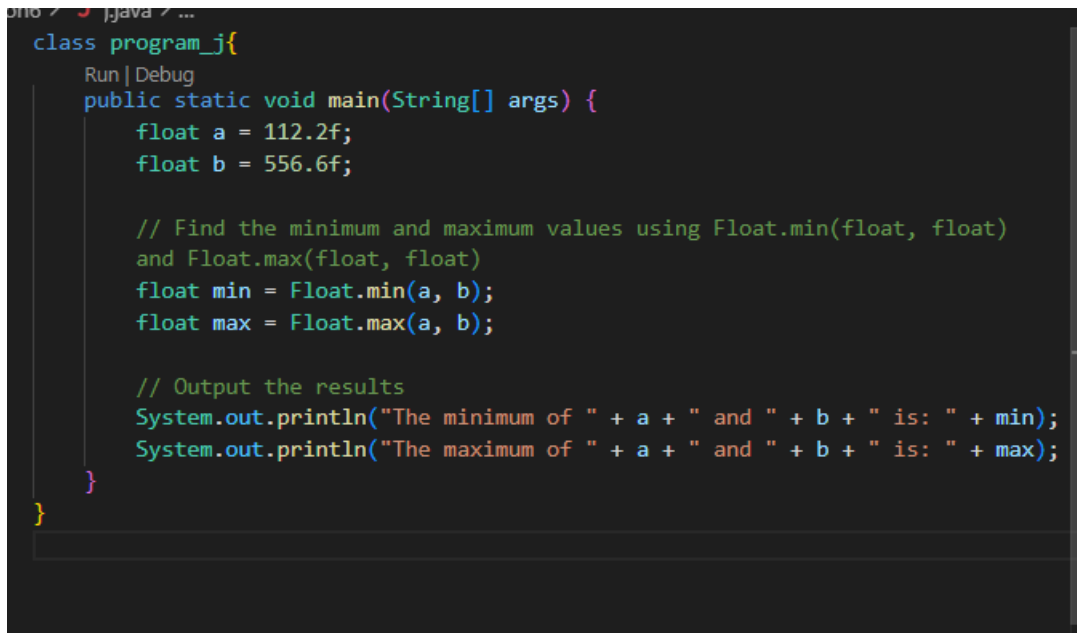
        System.out.println("The minimum of " + a + " and " + b + " is: " + min);

        System.out.println("The maximum of " + a + " and " + b + " is: " + max);

    }

}

```



```

class program_j{
    Run | Debug
    public static void main(String[] args) {
        float a = 112.2f;
        float b = 556.6f;

        // Find the minimum and maximum values using Float.min(float, float)
        // and Float.max(float, float)
        float min = Float.min(a, b);
        float max = Float.max(a, b);

        // Output the results
        System.out.println("The minimum of " + a + " and " + b + " is: " + min);
        System.out.println("The maximum of " + a + " and " + b + " is: " + max);
    }
}

```

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

Ans. `class program_k{`

```

    public static void main(String[] args) {

        float number = -25.0f;

        // Find the square root of the float value

        // Note: sqrt() returns NaN for negative numbers

        double sqrtValue = Math.sqrt(number);

```

```

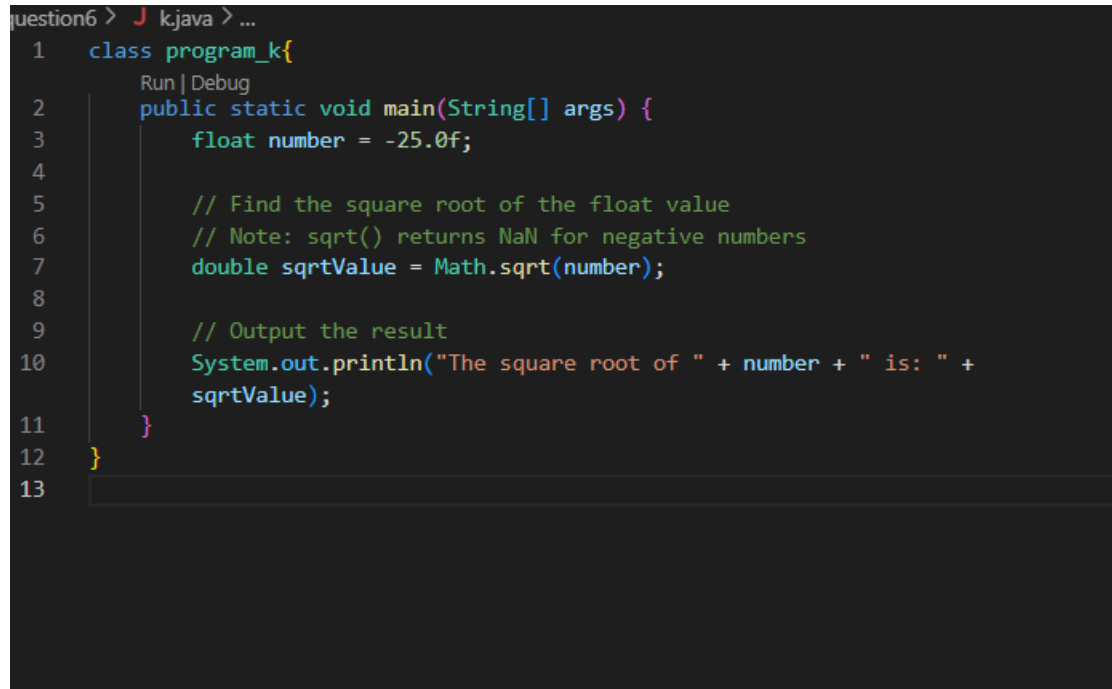
        // Output the result

        System.out.println("The square root of " + number + " is: " + sqrtValue);

    }

}

```



The screenshot shows an IDE window with a Java file named 'program_k.java'. The code is as follows:

```

1  class program_k{
2      public static void main(String[] args) {
3          float number = -25.0f;
4
5          // Find the square root of the float value
6          // Note: sqrt() returns NaN for negative numbers
7          double sqrtValue = Math.sqrt(number);
8
9          // Output the result
10         System.out.println("The square root of " + number + " is: " +
11                             sqrtValue);
12     }
13 }

```

7. Working with java.lang.Double

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

```

Ans. class program_b{

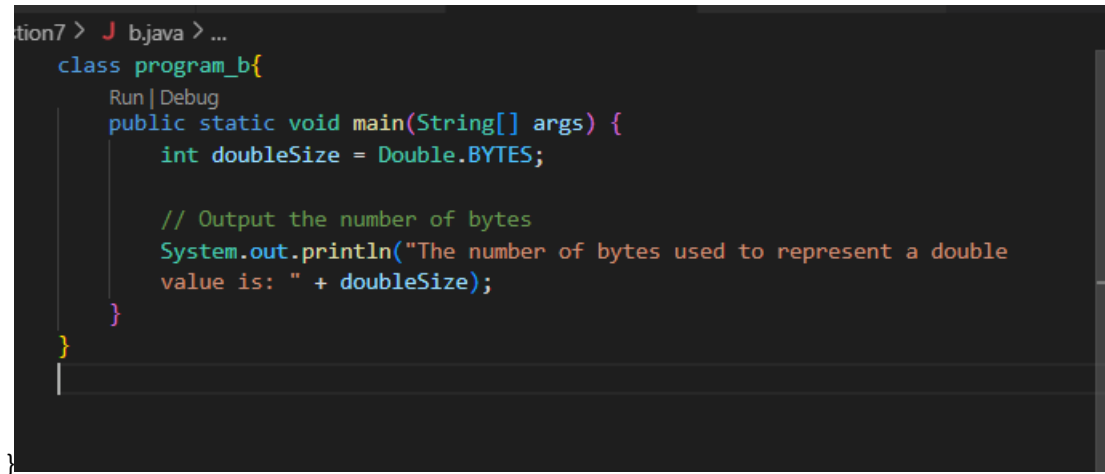
public static void main(String[] args) {

    int doubleSize = Double.BYTES;

    // Output the number of bytes

```

```
System.out.println("The number of bytes used to represent a double value is: " + doubleSize);  
}
```

A screenshot of an IDE window showing a Java program. The code defines a class named 'program_b' with a 'main' method. Inside the 'main' method, it declares an integer variable 'doubleSize' and assigns it the value of 'Double.BYTES'. A comment indicates the purpose is to output the number of bytes. The 'main' method then prints a string that includes the value of 'doubleSize'. The IDE interface includes a 'Run | Debug' button above the code.

```
tion7 > J b.java > ...  
class program_b{  
    Run | Debug  
    public static void main(String[] args) {  
        int doubleSize = Double.BYTES;  
  
        // Output the number of bytes  
        System.out.println("The number of bytes used to represent a double  
        value is: " + doubleSize);  
    }  
}
```

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).

Ans. class program_c{

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

```
    double minValue = Double.MIN_VALUE;
```

```
    double maxValue = Double.MAX_VALUE;
```

```
    // Output the minimum and maximum values
```

```
    System.out.println("The minimum value of a double is: " + minValue);
```

```
    System.out.println("The maximum value of a double is: " + maxValue);
```

```
}
```

```
}
```

```
class program_c{
    Run | Debug
    public static void main(String[] args) {
        double minValue = Double.MIN_VALUE;
        double maxValue = Double.MAX_VALUE;

        // Output the minimum and maximum values
        System.out.println("The minimum value of a double is: " + minValue);
        System.out.println("The maximum value of a double is: " + maxValue);
    }
}
```

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)`).

Ans. class `program_d`{

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

double `number` = 123.456;

// Convert the double value to a `String` using `Double.toString(double)`

`String` `numberString` = `Double.toString(number)`;

// Output the resulting `String`

`System.out.println`("The double value as a `String` is: " + `numberString`);

}

}

```

class program_d{
    Run | Debug
    public static void main(String[] args) {
        double number = 123.456;

        // Convert the double value to a String using Double.toString(double)
        String numberString = Double.toString(number);

        // Output the resulting String
        System.out.println("The double value as a String is: " + numberString);
    }
}

```

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)`).

Ans. class program_e{

```

    public static void main(String[] args) {

```

```

        String strNumber = "123.456";

```

```

        // Convert the String to a double value using Double.parseDouble(String)

```

```

        try {

```

```

            double number = Double.parseDouble(strNumber);

```

```

            System.out.println("The double value is: " + number);

```

```

        } catch (NumberFormatException e) {

```

```

            System.out.println("Error: The String \"" + strNumber + "\" is not a valid double value.");

```

```

        }

```

```

    }

```

```

}

```



```

class program_e{
    Run | Debug
    public static void main(String[] args) {
        String strNumber = "123.456";

        // Convert the String to a double value using Double.parseDouble(String)
        try {
            double number = Double.parseDouble(strNumber);
            System.out.println("The double value is: " + number);
        } catch (NumberFormatException e) {
            System.out.println("Error: The String \"" + strNumber + "\" is not a valid double value.");
        }
    }
}

```

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`).

Ans. class `program_f`{

```

    public static void main(String[] args) {

```

```

        String strNumber = "Ab12Cd3";

```

```

        // Attempt to convert the String to a double value

```

```

        try {

```

```

            double number = Double.parseDouble(strNumber);

```

```

            System.out.println("The double value is: " + number);

```

```

        } catch (NumberFormatException e) {

```

```

            System.out.println("Error: The String \"" + strNumber + "\" is not a valid double value.");

```

```

        }

```

```

    }

```

```

}

```

```

1  class program_f{
2      Run | Debug
3      public static void main(String[] args) {
4          String strNumber = "Ab12Cd3";
5
6          // Attempt to convert the String to a double value
7          try {
8              double number = Double.parseDouble(strNumber);
9              System.out.println("The double value is: " + number);
10         } catch (NumberFormatException e) {
11             System.out.println("Error: The String \"" + strNumber + "\" is not
12             a valid double value.");
13         }
14     }
15 }

```

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)`).

Ans. class `program_g`{

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

```
    // Declare a method-local variable of type double
```

```
    double number = 123.456;
```

```
    // Convert the double value to a Double object using Double.valueOf(double)
```

```
    Double doubleObject = Double.valueOf(number);
```

```
    // Output the resulting Double object
```

```
    System.out.println("The Double object is: " + doubleObject);
```

```
}
```

```
}
```

```
class program_g{
    public static void main(String[] args) {
        // Declare a method-local variable of type double
        double number = 123.456;

        // Convert the double value to a Double object using Double.valueOf
        // (double)
        Double doubleObject = Double.valueOf(number);

        // Output the resulting Double object
        System.out.println("The Double object is: " + doubleObject);
    }
}
```

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)`).

Ans. class `program_h`{

public static void main(String[] args) {

String strNumber = "123.456";

// Convert the String to a Double object using Double.valueOf(String)

try {

Double doubleObject = Double.valueOf(strNumber);

System.out.println("The Double object is: " + doubleObject);

} catch (NumberFormatException e) {

System.out.println("Error: The String \"" + strNumber + "\" is not a valid double value.");

}

}

}

```

1  class program_h{
2      Run | Debug
3      public static void main(String[] args) {
4          String strNumber = "123.456";
5
6          // Convert the String to a Double object using Double.valueOf(String)
7          try {
8              Double doubleObject = Double.valueOf(strNumber);
9              System.out.println("The Double object is: " + doubleObject);
10         } catch (NumberFormatException e) {
11             System.out.println("Error: The String \"" + strNumber + "\" is not
12             a valid double value.");
13         }
14     }
15 }

```

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)`).

Ans.

```

class program_i{

    public static void main(String[] args) {

        double a = 112.3;

        double b = 984.5;

        // Add the two double variables using Double.sum(double, double)

        double sum = Double.sum(a, b);

        // Output the result

        System.out.println("The sum of " + a + " and " + b + " is: " + sum);

    }
}

```

```
}
```

```
class program_i{
    Run | Debug
    public static void main(String[] args) {
        double a = 112.3;
        double b = 984.5;

        // Add the two double variables using Double.sum(double, double)
        double sum = Double.sum(a, b);

        // Output the result
        System.out.println("The sum of " + a + " and " + b + " is: " + sum);
    }
}
```

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)).

Ans. class program_j{

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

```
        double a = 112.2;
```

```
        double b = 556.6;
```

```
        // Find the minimum and maximum values using Double.min(double, double) and
        Double.max(double, double)
```

```
        double min = Double.min(a, b);
```

```
        double max = Double.max(a, b);
```

```
        // Output the results
```

```
        System.out.println("The minimum of " + a + " and " + b + " is: " + min);
```

```
        System.out.println("The maximum of " + a + " and " + b + " is: " + max);
```

```
    }
```

```
}
```

```

class program_j{
    Run | Debug
    public static void main(String[] args) {
        double a = 112.2;
        double b = 556.6;

        // Find the minimum and maximum values using Double.min(double, double)
        // and Double.max(double, double)
        double min = Double.min(a, b);
        double max = Double.max(a, b);

        // Output the results
        System.out.println("The minimum of " + a + " and " + b + " is: " + min);
        System.out.println("The maximum of " + a + " and " + b + " is: " + max);
    }
}

```

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

Ans. class program_k{

```

    public static void main(String[] args) {

```

```

        double number = -25.0;

```

```

        // Find the square root of the double value

```

```

        // Note: sqrt() returns NaN for negative numbers

```

```

        double sqrtValue = Math.sqrt(number);

```

```

        // Output the result

```

```

        System.out.println("The square root of " + number + " is: " + sqrtValue);

```

```

    }

```

```

}

```

```

1  class program_k{
    Run | Debug
2  public static void main(String[] args) {
3      double number = -25.0;
4
5      // Find the square root of the double value
6      // Note: sqrt() returns NaN for negative numbers
7      double sqrtValue = Math.sqrt(number);
8
9      // Output the result
10     System.out.println("The square root of " + number + " is: " +
        sqrtValue);
11 }
12 }
13

```

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()`).

Ans.

```

class PrimitiveToStringConversion {

    public static void main(String[] args) {

        // Initialize variables of each primitive type

        int intValue = 123;

        long longValue = 123456789L;

        float floatValue = 12.34f;

        double doubleValue = 123.456;

        char charValue = 'A';

        short shortValue = 12345;
    }
}

```

```
byte byteValue = 123;

boolean booleanValue = true;


// Convert each primitive type to String using the toString method of the corresponding wrapper class

String intToString = Integer.toString(intValue);

String longToString = Long.toString(longValue);

String floatToString = Float.toString(floatValue);

String doubleToString = Double.toString(doubleValue);

String charToString = Character.toString(charValue);

String shortToString = Short.toString(shortValue);

String byteToString = Byte.toString(byteValue);

String booleanToString = Boolean.toString(booleanValue);


// Convert each primitive type to String using the valueOf method of the String class

String intValueOf = String.valueOf(intValue);

String longValueOf = String.valueOf(longValue);

String floatValueOf = String.valueOf(floatValue);

String doubleValueOf = String.valueOf(doubleValue);

String charValueOf = String.valueOf(charValue);

String shortValueOf = String.valueOf(shortValue);

String byteValueOf = String.valueOf(byteValue);

String booleanValueOf = String.valueOf(booleanValue);


// Output the results

System.out.println("Using toString method:");
```



```
System.out.println("int to String: " + toString());
System.out.println("long to String: " + toString());
System.out.println("float to String: " + toString());
System.out.println("double to String: " + toString());
System.out.println("char to String: " + toString());
System.out.println("short to String: " + toString());
System.out.println("byte to String: " + toString());
System.out.println("boolean to String: " + toString());
```

```
System.out.println("\nUsing valueOf method:");
System.out.println("int to String: " + Integer.valueOf(1));
System.out.println("long to String: " + Long.valueOf(1L));
System.out.println("float to String: " + Float.valueOf(1.0f));
System.out.println("double to String: " + Double.valueOf(1.0));
System.out.println("char to String: " + Character.valueOf('a'));
System.out.println("short to String: " + Short.valueOf((short)1));
System.out.println("byte to String: " + Byte.valueOf((byte)1));
System.out.println("boolean to String: " + Boolean.valueOf(true));
```

```
}
```

```
}
```

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).

```
Ans. class DefaultValuesExample {  
  
    // Instance variables  
  
    int instanceInt;  
  
    long instanceLong;  
  
    float instanceFloat;  
  
    double instanceDouble;  
  
    char instanceChar;  
  
    short instanceShort;  
  
    byte instanceByte;  
  
    boolean instanceBoolean;  
  
  
    // Static variables  
  
    static int staticInt;  
  
    static long staticLong;  
  
    static float staticFloat;  
  
    static double staticDouble;  
  
    static char staticChar;  
  
    static short staticShort;  
  
    static byte staticByte;  
  
    static boolean staticBoolean;  
  
  
    public static void main(String[] args) {  
  
        // Create an instance of the class to check instance variables  
  
        DefaultValuesExample example = new DefaultValuesExample();  
    }  
}
```

```

// Output default values of instance variables

System.out.println("Default values of instance variables:");

System.out.println("int: " + example.instanceInt);

System.out.println("long: " + example.instanceLong);

System.out.println("float: " + example.instanceFloat);

System.out.println("double: " + example.instanceDouble);

System.out.println("char: [" + example.instanceChar + "]"); // char defaults to '\u0000', which is an
empty character

System.out.println("short: " + example.instanceShort);

System.out.println("byte: " + example.instanceByte);

System.out.println("boolean: " + example.instanceBoolean);


// Output default values of static variables

System.out.println("\nDefault values of static variables:");

System.out.println("int: " + DefaultValuesExample.staticInt);

System.out.println("long: " + DefaultValuesExample.staticLong);

System.out.println("float: " + DefaultValuesExample.staticFloat);

System.out.println("double: " + DefaultValuesExample.staticDouble);

System.out.println("char: [" + DefaultValuesExample.staticChar + "]"); // char defaults to '\u0000',
which is an empty character

System.out.println("short: " + DefaultValuesExample.staticShort);

System.out.println("byte: " + DefaultValuesExample.staticByte);

System.out.println("boolean: " + DefaultValuesExample.staticBoolean);

}

}

```

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).

```
Ans. class ArithmeticOperations {
    public static void main(String[] args) {
        // Check if the correct number of arguments are provided
        if (args.length != 3) {
            System.out.println("Usage: java ArithmeticOperations <number1> <number2> <operator>");
            System.out.println("Example: java ArithmeticOperations 10 20 +");
            return;
        }

        // Parse command line arguments
        int number1;
        int number2;
        String operator = args[2];

        try {
            number1 = Integer.parseInt(args[0]);
            number2 = Integer.parseInt(args[1]);
        } catch (NumberFormatException e) {
            System.out.println("Error: The first two arguments must be integers.");
            return;
        }

        // Perform arithmetic operation based on the operator
        double result;
        switch (operator) {
            case "+":
                result = number1 + number2;
                break;
            case "-":
                result = number1 - number2;
                break;
            case "*":
                result = number1 * number2;
                break;
            case "/":
                if (number2 == 0) {
                    System.out.println("Error: Division by zero is not allowed.");
                    return;
                }
                result = (double) number1 / number2;
                break;
            default:
                System.out.println("Error: Unsupported operator. Please use +, -, *, or /.");
                return;
        }
    }
}
```

```
}  
  
// Output the result  
System.out.printf("Result: %d %s %d = %.2f%n", number1, operator, number2, result);  
}  
}
```