**Note**: Consider the following before starting the assignment:

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

#### ****1. Working with**** java.lang.Boolean

**a.** Explore the [Java API documentation for java.lang.Boolean](https://docs.oracle.com/javase/8/docs/api/java/lang/Boolean.html) and observe its modifiers and super types.

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

public class A1\_boolean {

        public static void main(String[] args) {

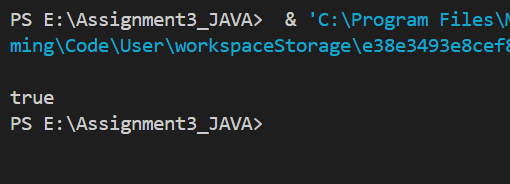
            boolean status = true ;

            String stringstr = Boolean.toString(status);

            System.out.println(stringstr);

        }

    }



**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)).

public class C1\_boolean {

    public static void main(String[] args) {

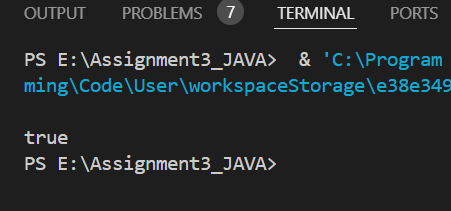
        String strStatus ="true";

        Boolean.parseBoolean(strStatus);

        System.out.println(strStatus);

     }

   }



**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").

public class D1\_boolean {

    public static void main(String[] args) {

        String strStatus = "1";

        boolean b1 = Boolean.parseBoolean(strStatus);

        System.out.println(b1);

    }}

A screenshot of a computer screen

Description automatically generated

**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)).

public class E1\_boolean {

    public static void main (String [] args){

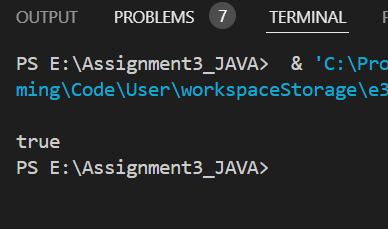
        boolean status = true;

        Boolean.valueOf(status);

        System.out.println(status);

    }

}



**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)).

public class F1\_boolean {

    public static void main(String[] args) {

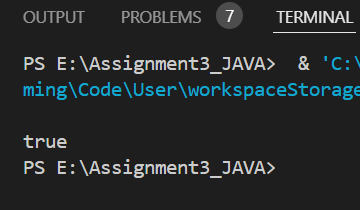
        boolean status = true;

        boolean ab =Boolean.valueOf(status);

        System.out.println(ab);

    }

}



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

public class G1\_boolean {

        public static void main(String args [])

        {

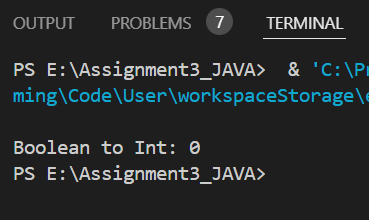
        boolean status=false;

        int strbool = (status) ? 1 :0;

        System.out.println("Boolean to Int: "+strbool);

        }

        }



#### ****2. Working with**** java.lang.Byte

**a.** Explore the [Java API documentation for java.lang.Byte](https://docs.oracle.com/javase/8/docs/api/java/lang/Byte.html) and observe its modifiers and super types.

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

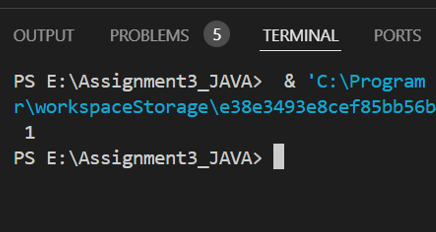
public class B2\_byte {

    public static void main(String[] args) {

        System.out.println(" " + 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).

public class B3\_byte {

    public static void main(String[] args) {

        System.out.println(" " + Byte.MIN\_VALUE );

        System.out.println(" " + Byte.MAX\_VALUE);

    }

}

A screen shot of a computer

Description automatically generated

**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)).

public class B4\_byte {

    public static void main(String [] args){

        byte b = 20;

        String s = Byte.toString(b);

        System.out.println(s);

    }

}

A black screen with blue text

Description automatically generated

**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)).

public class B5\_byte {

    public static void main(String[] args) {

        String strNumber = "25";

        byte b1 = Byte.parseByte(strNumber);

        System.out.println(b1);

    }

}

A screenshot of a computer

Description automatically generated

**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).

public class B6\_byte {

    public static void main(String[] args) {

        String strNumber = "Ab12Cd3";

        byte b1 = Byte.parseByte(strNumber);

        System.out.println("b1");

    }

}

A screen shot of a computer program

Description automatically generated

**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)).

public class B7\_byte {

    public static void main(String[] args) {

     byte number = 24;

     byte ab = Byte.valueOf(number);

     System.out.println(number);

}

}

A black screen with blue text

Description automatically generated

**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)).

public class ByteH {

public static void main(String[] args) {

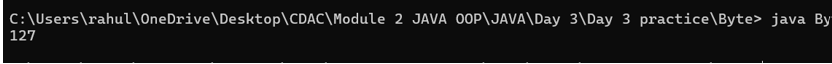
String strNumber = "127";

Byte byteObject = Byte.valueOf(strNumber);

System.out.println("Byte object from string: " + byteObject);

}

}



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

public class B8\_byte {

    public static void main(String[] args) {

        byte number = 10;

        int intValue = number;

        short shortValue = number;

        long longValue = number;

        float floatValue = number;

        double doubleValue = number;

        System.out.println("Byte value as int: " + intValue);

        System.out.println("Byte value as short: " + shortValue);

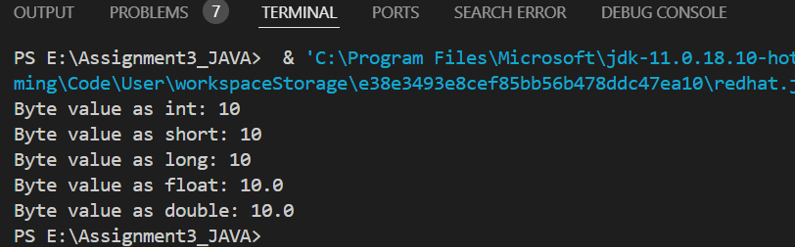
        System.out.println("Byte value as long: " + longValue);

        System.out.println("Byte value as float: " + floatValue);

        System.out.println("Byte value as double: " + doubleValue);

    }

}



#### ****3. Working with**** java.lang.Short

**a.** Explore the [Java API documentation for java.lang.Short](https://docs.oracle.com/javase/8/docs/api/java/lang/Short.html) and observe its modifiers and super types.

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

public class ShortBytes {

    public static void main(String[] args) {

        System.out.println("Bytes used to represent " + Short.BYTES);

    }

}

A screen shot of a computer

Description automatically generated

**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).

public class ShortBytesC {

        public static void main(String[] args) {

            System.out.println("Minimum short value: " + Short.MIN\_VALUE);

            System.out.println("Maximum short value: " + Short.MAX\_VALUE);

        }

    }

A screen shot of a computer

Description automatically generated

**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)).

public static void main(String[] args) {

short number = 32000;

String strNumber = Short.toString(number);

System.out.println("String representation of short value: " + strNumber);

}

A screen shot of a computer

Description automatically generated

**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)).

public class ShortByteE {

    public static void main(String[] args) {

        String strNumber = "30";

        short number = Short.parseShort(strNumber);

        System.out.println("Short value from string " +number);

    }

}

A screenshot of a computer

Description automatically generated

**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).

public class ShortByteF {

    public static void main(String[] args) {

                String strNumber = "Ab12cd3";

                short number = Short.parseShort(strNumber);

                System.out.println("Short value from string: " + number);

            }

        }

A screen shot of a computer

Description automatically generated

**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)).

public class ShortByteG {

   public static void main(String[] args) {

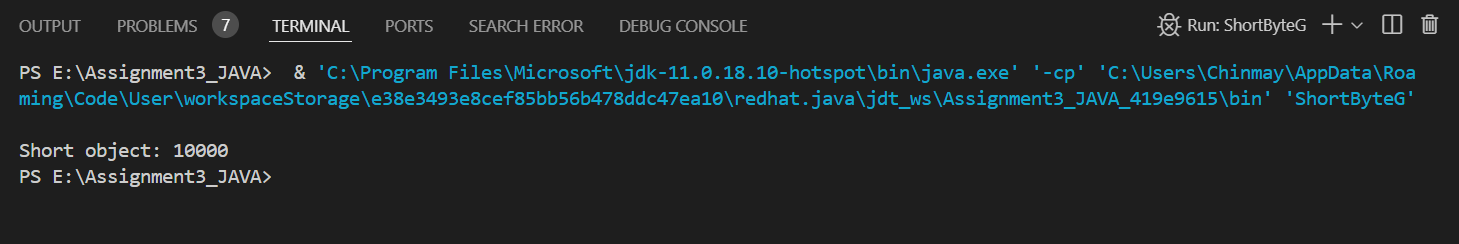
            short number = 10000;

            Short wrapper = Short.valueOf(number);

            System.out.println("Short object: " + wrapper);

        }

    }



**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)).

public class ShortByteH {

  public static void main(String[] args) {

    String strNumber = "32767";

        Short shortval = Short.valueOf(strNumber);

        System.out.println("Short object from string: " + shortval);

}

}

A computer screen with blue text

Description automatically generated

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

public class ShortByteI {

public static void main(String[] args) {

short number = 100;

int intValue = number;

byte byteValue = (byte) number; // Casting needed

long longValue = number;

float floatValue = number;

double doubleValue = number;

System.out.println("Short value as int: " + intValue);

System.out.println("Short value as byte (with casting): " + byteValue);

System.out.println("Short value as long: " + longValue);

System.out.println("Short value as float: " + floatValue);

System.out.println("Short value as double: " + doubleValue);

}

}

A screen shot of a computer error

Description automatically generated

#### ****4. Working with**** java.lang.Integer

**a.** Explore the [Java API documentation for java.lang.Integer](https://docs.oracle.com/javase/8/docs/api/java/lang/Integer.html) 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).

**program:**

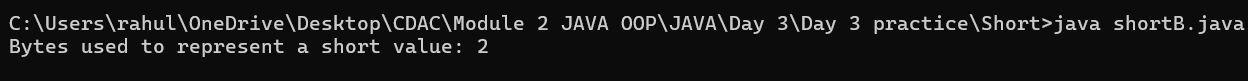
public class ShortB {

public static void main(String[] args) {

System.out.println("Bytes used to represent a short value: " + Short.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).

**Program:**

public class ShortC{

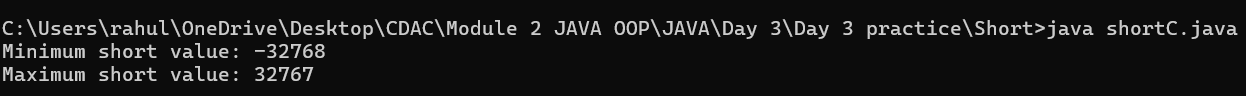
public static void main(String[] args) {

System.out.println("Minimum short value: " + Short.MIN\_VALUE);

System.out.println("Maximum short value: " + Short.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)).

**Program:**

**public class IntegerExample {**

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

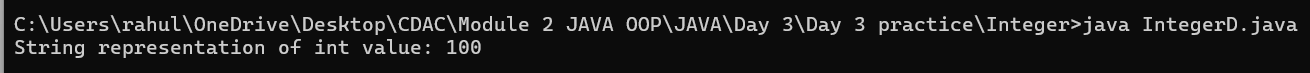
**int number = 100;**

**String strNumber = Integer.toString(number);**

**System.out.println("String representation of int value: " + strNumber);**

**}**

**}**



**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)).

**Program:**

**public class IntegerExample {**

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

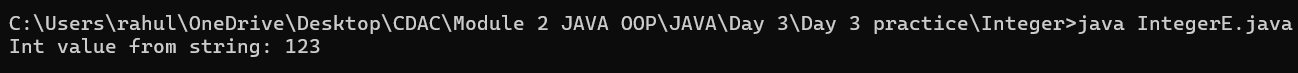
**String strNumber = "123";**

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

**System.out.println("Int value from string: " + number);**

**}**

**}**



**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).

**Program:**

public class ShortF {

public static void main(String[] args) {

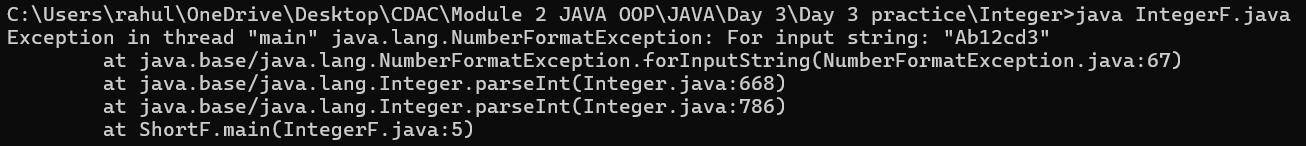
String strNumber = "Ab12cd3";

int number = Integer.parseInt(strNumber);

System.out.println("int value from string: " + number);

}

}



**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)).

**Program:**

**public class IntegerG{**

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

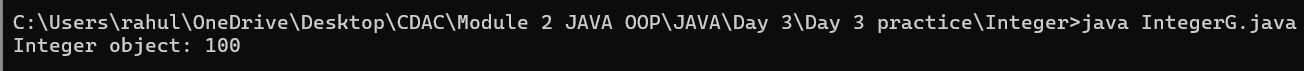
**int number = 100;**

**Integer wrapperObject = Integer.valueOf(number);**

**System.out.println("Integer object: " + wrapperObject);**

**}**

**}**



**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)).

**Program:**

public class IntegerExample {

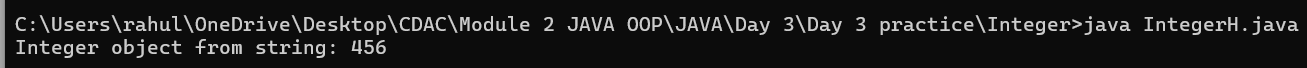
public static void main(String[] args) {

String strNumber = "456";

int wrapperObject = Integer.valueOf(strNumber);

System.out.println("Integer object from string: " + wrapperObject);

}



**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)).

public class IntegerExample {

public static void main(String[] args) {

int a = 10;

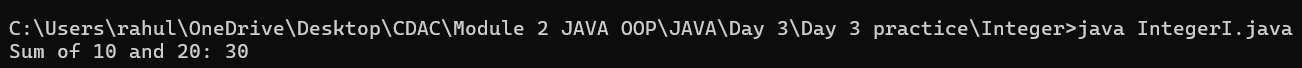
int b = 20;

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

System.out.println("Sum of 10 and 20: " + sum);

}

}

****

**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)).

**Program:**

**public class IntegerJ{**

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

**int a = 10;**

**int b = 20;**

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

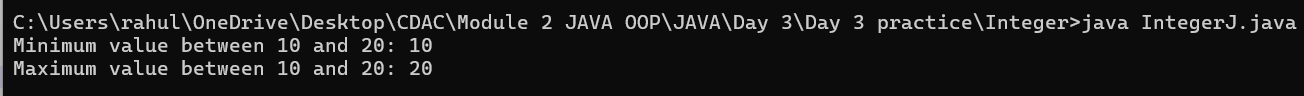
**int maxValue = Integer.max(a, b);**

**System.out.println("Minimum value between 10 and 20: " + minValue);**

**System.out.println("Maximum value between 10 and 20: " + maxValue);**

**}**

**}**



**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)).

**Program:**

public class IntegerExample {

public static void main(String[] args) {

int number = 7;

String binary = Integer.toBinaryString(number);

String octal = Integer.toOctalString(number);

String hex = Integer.toHexString(number);

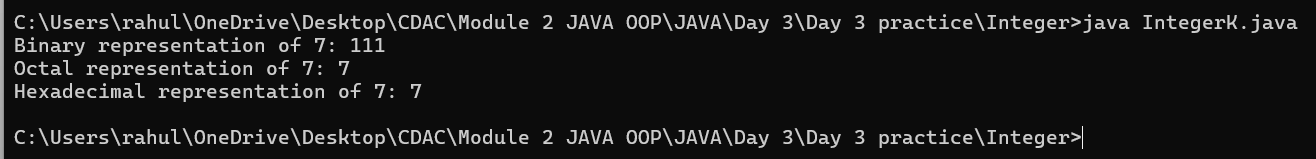
System.out.println("Binary representation of 7: " + binary);

System.out.println("Octal representation of 7: " + octal);

System.out.println("Hexadecimal representation of 7: " + hex);

}

}



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

Program:

public class IntegerExample {

public static void main(String[] args) {

int number = 45 ;

byte byteValue = (byte) number; // Explicit casting needed

short shortValue = (short) number; // Explicit casting

long longValue = number;

float floatValue = number;

double doubleValue = number;

System.out.println("Int value as byte : " + byteValue); // with casting

System.out.println("Int value as short : " + shortValue); //with casting

System.out.println("Int value as long: " + longValue);

System.out.println("Int value as float: " + floatValue);

System.out.println("Int value as double: " + doubleValue);

}

}

A black screen with white text

Description automatically generated

#### ****5. Working with**** java.lang.Long

**a.** Explore the [Java API documentation for java.lang.Long](https://docs.oracle.com/javase/8/docs/api/java/lang/Long.html) and observe its modifiers and super types.

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

**Program**

public class Main {

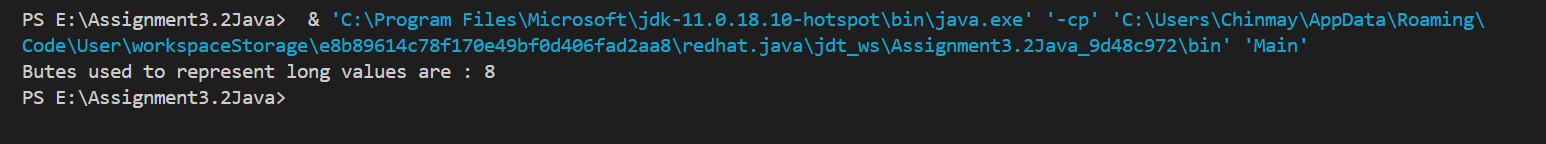
    public static void main(String[] args) {

        System.out.println("Butes used to represent long values are : " + Long.BYTES);

    }

}

Outpt



**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).

**Program**

public class Q5c {

    public static void main(String[] args) {

        System.out.println("Minumum value of long is : " + Long.MIN\_VALUE);

        System.out.println("Minumum value of long is : " + Long.MAX\_VALUE);

    }

}

Output

A computer screen with blue text

Description automatically generated

**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)).

**Program**

public class Q5d {

    public static void main(String[] args) {

        long a  = 1500000L;

        String str = Long.toString(a);

        System.out.println("String representation of long value: " + a);

    }

}

A computer screen with numbers and letters

Description automatically generated

**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)).

**Input**

public class Q5f {

    public static void main(String[] args) {

        String strNumber = "12334";

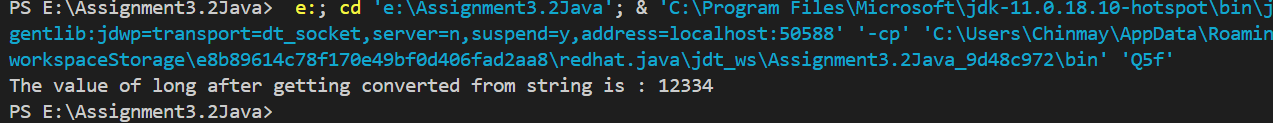
        Long num = Long.parseLong(strNumber);

        System.out.println("The value of long after getting converted from string is : "+ num);

    }

}

Output:



**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).

**Program:**

public class Q5e {

    public static void main(String[] args) {

        String strNumber = "Ab12Cd3";

        Long number = Long.parseLong(strNumber);

        System.out.println("The value of long from string is: "+ strNumber);

    }

}

A computer screen shot of a program code

Description automatically generated

**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)).

**Program:**

public class Q5g {

    public static void main(String[] args) {

        long num = 1500000L;

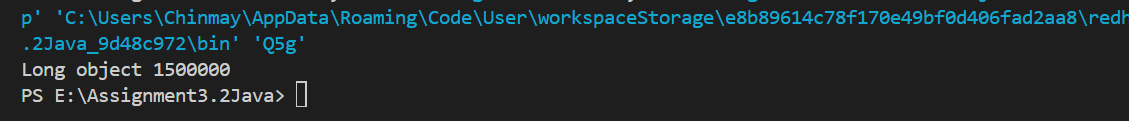
        Long wrapperobj = Long.valueOf(num);

        System.out.println("Long object "+ wrapperobj);

    }

}

Output:



**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)).

**Program:**

public class Q5g {

    public static void main(String[] args) {

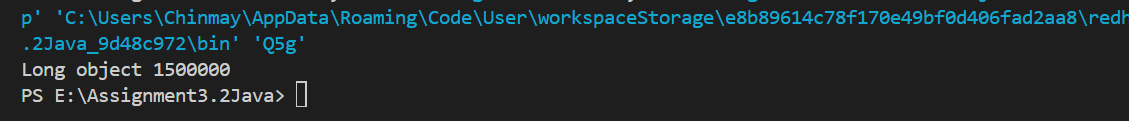
        long num = 1500000L;

        Long wrapperobj = Long.valueOf(num);

        System.out.println("Long object "+ wrapperobj);

    }

}



1. 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)).

Program:

public class Q5i {

    public static void main(String[] args) {

        long a = 1123L;

        long b = 9845L;

        long c = Long.sum(a, b);

        System.out.println(c);

    }

}

A computer screen with blue text

Description automatically generated

**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)).

**Program**

public class Q5j {

    public static void main(String[] args) {

        long a = 1122L;

        long b = 5566L;

        long c = Long.min(a, b);

        System.out.println("The minimum of two numbers is: " + c);

    }

}

Output:

A computer screen with blue text

Description automatically generated

**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](https://docs.oracle.com/javase/8/docs/api/java/lang/Float.html) and observe its modifiers and super types.

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

public class Q6b{

public static void main(String args[])

{

System.out.println(" "+Float.BYTES);

}

}

A black background with white text

Description automatically generated

**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).

public class Q6c{

public static void main(String args[])

{

System.out.println("Minimum value is "+Float.MIN\_VALUE);

System.out.println("MAximum value is "+Float.MAX\_VALUE);

}

}

A black screen with white text

Description automatically generated

**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)).

public class Q6d{

public static void main(String args[])

{

float number=12345f;

String str=Float.toString(number);

System.out.println("Value in String is : "+str );

}

}

Output:

A black screen with white text

Description automatically generated

**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)).

public class Q6e{

public static void main(String[] args)

{

String strNumber = "1234";

float number = Float.parseFloat(strNumber);

System.out.println("Value in Long is : "+number );

}

}

A black screen with white text

Description automatically generated

**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).

**public class Q6f{**

**public static void main(String args[])**

**{**

**String strNumber = "Ab12Cd3";**

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

**System.out.println("Value in Float is : "+number );**

**}**

**}**

A screen shot of a computer code

Description automatically generated

**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)).

**public class Q6g{**

**public static void main(String args[])**

**{**

**float number=2f;**

**float num=Float.valueOf(number);**

**System.out.println("Value in Float is : "+num );**

**}**

**}**

A black background with white text

Description automatically generated

**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)).

public class Q6h{

public static void main(String args[])

{

String number="258745";

float num=Float.valueOf(number);

System.out.println("Value in Float is : "+num );

}

}

A screen shot of a computer code

Description automatically generated

1. 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)).

public class Q6i{

public static void main(String args[])

{

float num1=112.3f;

float num2=984.5f;

System.out.println("Sum is :"+Float.sum(num1,num2));

}

}

A black background with white text

Description automatically generated

**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)).

public class Q6j{

public static void main(String args[])

{

float num1=112.2f;

float num2=556.6f;

System.out.println("Maximum value is :"+Float.max(num1,num2));

System.out.println("Minimum value is :"+Float.min(num1,num2));

}

}

A black screen with white text

Description automatically generated

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

**public class Q6k{**

**public static void main(String args[])**

**{**

**float num1=-25.0f;**

**float result=(float)Math.sqrt(num1);**

**System.out.println("Sqaure root is :"+result);**

**}**

**}**

A black screen with white text

Description automatically generated

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

**public class Q6l{**

**public static void main(String args[])**

**{**

**float num1=0.0f;**

**float num2=0.0f;**

**float result=num1/num2;**

**System.out.println("Result is :"+result);**

**}**

**}**

A black screen with white text

Description automatically generated

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

#### ****7. Working with**** java.lang.Double

**a.** Explore the [Java API documentation for java.lang.Double](https://docs.oracle.com/javase/8/docs/api/java/lang/Double.html) and observe its modifiers and super types.

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

public class Q7b{

public static void main(String args[])

{

System.out.println(" "+Double.BYTES);

}

}

A black background with white text

Description automatically generated

**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).

public class Q7c{

public static void main(String args[])

{

System.out.println("Minimum value is "+Double.MIN\_VALUE);

System.out.println("MAximum value is "+Double.MAX\_VALUE);

}

}

A black screen with white text

Description automatically generated

**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)).

public class Q7d{

public static void main(String args[])

{

double number=12d;

String str=Double.toString(number);

System.out.println("Value in String is : "+str );

}

}

A black screen with white text

Description automatically generated

**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)).

**public class Q7e{**

**public static void main(String[] args)**

**{**

**String strNumber = "1234";**

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

**System.out.println("Value in Long is : "+number );**

**}**

**}**

A black background with white text

Description automatically generated

**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).

public class Q7f{

public static void main(String args[])

{

String strNumber = "Ab12Cd3";

double number = Double.parseDouble (strNumber);

System.out.println("Value in double is : "+number );

}

}

A computer screen shot of a program code

Description automatically generated

**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)).

public class Q7g{

public static void main(String args[])

{

double number=2d;

double num=Double.valueOf(number);

System.out.println("Value in double is : "+num );

}

}

A black screen with white text

Description automatically generated

**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)).

public class Q7h{

public static void main(String args[])

{

String number="258745";

double num=Double.valueOf(number);

System.out.println("Value in double is : "+num );

}

}

A black background with white text

Description automatically generated

1. 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)).

public class Q7i{

public static void main(String args[])

{

double num1=112.3d;

double num2=984.5d;

System.out.println("Sum is :"+Double.sum(num1,num2));

}

}

A black screen with white text

Description automatically generated

**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)).

public class Q7j{

public static void main(String args[])

{

double num1=112.2d;

double num2=556.6d;

System.out.println("Maximum value is :"+Double.max(num1,num2));

System.out.println("Minimum value is :"+Double.min(num1,num2));

}

}

A black screen with white text

Description automatically generated

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

public class Q7k{

public static void main(String args[])

{

double num1=-25.0d;

double result=Math.sqrt(num1);

System.out.println("Sqaure root is :"+result);

}

}

A black screen with white text

Description automatically generated

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

public class Q7l{

public static void main(String args[])

{

double num1=0.0d;

double num2=0.0d;

double result=num1/num2;

System.out.println("Result is :"+result);

}

}

A black screen with white text

Description automatically generated

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

public class Q7m{

public static void main(String args[])

{

double strnumber=42;

int intnum=Integer.valueOf(strnumber);

System.out.println("Value in Integer is : "+strnumber );

}

}

#### ****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()).

public class problem8  
{  
public static void main(String[]args)  
{  
int a=10;  
String ab=Integer.toString(a);  
String b=String.valueOf(ab);  
System.out.println("Integer to String"+b);  
  
char c='S';  
String cd=Character.toString(c);  
String d=String.valueOf(cd);  
System.out.println("Character to String"+d);  
  
short e=10;  
String ef=Short.toString(e);  
String f=String.valueOf(ef);  
System.out.println("Short to String"+f);  
  
boolean g=true;  
String gh=Boolean.toString(g);  
String h=String.valueOf(gh);  
System.out.println("Boolean to String"+h);  
  
float i=10.3f;  
String ij=Float.toString(i);  
String j=String.valueOf(ij);  
System.out.println("Float to String"+j);  
  
}  
  
  
}

A screenshot of a computer program

Description automatically generated

#### ****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).

public class problem9  
{  
static char a;  
static short b;  
static int c;  
static float d;  
static double e;  
static long f;  
  
public static void main(String[]args)  
{  
  
System.out.println("Default value of Character: "+a);  
System.out.println("Default value of Short: "+b);  
System.out.println("Default value of Integer: "+c);  
System.out.println("Default value of Float: "+d);  
System.out.println("Default value of Double: "+e);  
System.out.println("Default value of Long: "+f);  
  
}  
  
  
}

A screen shot of a computer

Description automatically generated

#### ****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).

public class comline  
{  
public static void main(String args[])  
{  
  
int numa=Integer.parseInt(args[0]);  
int numb=Integer.parseInt(args[1]);  
//char c=Character.parseChar(args[2]);  
char ch=args[2].charAt(0);  
System.out.println("NUMBER 1 ="+numa);  
System.out.println("NUMBER 2 ="+numb);  
  
switch(ch)  
{  
case '+':  
int add=numa+numb;  
System.out.println("Addition of "+numa+"&"+numb+"is :"+add);  
break;  
case '-':  
int sub=numa-numb;  
System.out.println("Substraction of "+numa+"&"+numb+"is :"+sub);  
break;  
case '\*':  
int mul=numa\*numb;  
System.out.println("Multiplication of "+numa+"&"+numb+"is :"+mul);  
break;  
case '/':  
int div=numa/numb;  
System.out.println("Division of "+numa+"&"+numb+"is :"+div);  
break;  
default:  
System.out.println("Invalid input");  
}  
  
  
  
  
}  
}