Sandeep Sir ‘s Assignment no. 2

public class Assignment3\_Q1{

    public static void main(String args[]){

1. 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.

->It has only one super type i.e. java.lang.Object

**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) ).

->        // boolean status = true;

        // String statusString = Boolean.toString(status);

        // System.out.println(status);

        // System.out.println(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)).

->     // String strString="true";

        // boolean flag=Boolean.parseBoolean("strString");

        // System.out.println(strString);

        // System.out.println(flag);

**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").

->        // String strString="1";

        // boolean flag=Boolean.parseBoolean("strString");

        // System.out.println(strString);

        // System.out.println(flag);

It becomes ‘false’ whatever value we take in string.

**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)).

->        // boolean status=true;

        // Boolean wrapperBoolean=Boolean.valueOf(status);

        // System.out.println(status);

        // System.out.println(wrapperBoolean);

If boolean is false then Boolean is false , if boolean is true then Boolean is true.

**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)).

        ->// String strStatus="true";

        // boolean wrapBool=Boolean.valueOf(strStatus);

        // System.out.println(wrapBool);

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

        ->boolean Cannot be converted to another datatype

        // boolean status=true;

        // int num=(int)status;

        // System.out.println(num);

    }

 }

#### **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.

-> Super type of Byte are **java.lang.Object** and **java.lang.Number**

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

**-**> System.out.println("Number of bytes: "+Byte.BYTES);

Output:

Number of bytes: 1

**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).

-> System.out.println("Max value in byte="+Byte.MAX\_VALUE);

System.out.println("Min value in byte="+Byte.MIN\_VALUE);

Output:

Max value in byte:127

Min value in byte:-128

**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)).

-> byte num=120;

    System.out.println("The value of num in String is: "+**Byte.toString(num)**);

byte value gets converted to String value.

**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)).

-> String strString="121";

    byte num1=Byte.parseByte(strString);

    System.out.println("num1="+num1);

**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).

-> String strNumber="Ab12Cd3";

    byte num2=Byte.parseByte("strNumber");

Output:

It throws “NumberFormatException”.

It is because we are trying to convert String type character into byte type.

**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)).

-> byte number=20;

    Byte numByte=Byte.valueOf(number);

    System.out.println("The value of Byte is: "+numByte);

Output:

The value of Byte is: 20

byte value easily gets converted into Byte class.

**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)).

-> String strNumber="80";

    Byte byteWrapper=Byte.valueOf(strNumber);

    System.out.println("The value of Byte is: "+byteWrapper);

Output: String class value gets easily converted into Byte wrapper class object.

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

=> byte num4=60;

    int num5=(int)num4;

    double num6=(double) num4;

Output:

Byte is getting converted into all primitive types as it has smallest range.

#### **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.

**=>It has two super class java .lang.Object and java.lang.Number**

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

**=>** **System.out.println("Number of bytes in Short: "+Short.BYTES);**

Output: **Number of bytes in Short:2**

**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).

=> **System.out.println("Max value in Short ="+Short.MAX\_VALUE);**

**System.out.println("Min value in Short ="+Short.MIN\_VALUE);**

Output: **Max value in Short =32767**

**Min value in Short =-32768**

**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)).

**=>** **short num=12011;**

**System.out.println("The value of num in String is: "+Short.toString(num));**

**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)).

**=>** **String strNumber="7878";**

**Short num=Short.parseShort(strNumber);**

**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).

**=>** **String strNumber="Ab12Cd3";**

**short shNumber=Byte.parseByte("strNumber");**

**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)).

**=>** **short number=5657;**

**Short numShort=Short.valueOf(number);**

**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)).

**=>** **String strNumber ="9090";**

**Short shortVal=Short.valueOf(strNumber);**

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

**=> short num4=6099;**

**int num5=(int)num4;**

**byte strNum= (byte) num4;**

**double num6=(double) num4;**

#### **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.

=> **It has two super class java .lang.Object and java.lang.Number**

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

=> **System.out.println("Number of bytes in Integer: "+Integer.BYTES);**

**c.** Write a program to find the minimum and maximum values of int using the MIN\_VALUE and MAX\_VALUE fields. (Hint: Use Integer.MIN\_VALUE and Integer.MAX\_VALUE).

=> System.out.println("Max value in Integer ="+Integer.MAX\_VALUE);

        System.out.println("Min value in integer ="+Integer.MIN\_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)).

=> int number=6768678;

        System.out.println("The value of num in String is: "+Integer.toString(number));

**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)).

=> String strNumber="7577878";

        int number=Integer.parseInt(strNumber);

**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).

=>  String strNumber="Ab12Cd3";

        int strNumber=Integer.parseInt("strNumber");

**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)).

=> int number=67687;

        Integer intObj=Integer.valueOf(number);

**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)).

=> String strNumber="345356";

        int number=Integer.parseInt(strNumber);

**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)).

=> int n1=10,n2=20;

        System.out.println(Integer.sum(n1,n2));

**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)).

=> int n1=10,n2=20;

        System.out.println("Minimun of two number is :"+Integer.min(n1,n2));

        System.out.println("Maximum of two number is :"+Integer.max(n1,n2));

**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)).

=> int n=7;

        System.out.println("Binary of 7 is :"+Integer.toBinaryString(n));

        System.out.println("Octal of 7 is :"+Integer.toOctalString(n));

        System.out.println("Hexadecimal of 7 is :"+Integer.toHexString(7));

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

=>int cannot be converted into narrower types like byte and short if it is out of their range. It can be converted into float and double types.

#### **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

#### => It has two super class java .lang.Object and java.lang.Number.

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

=> System.out.println("Number of bytes in Long: "+Long.BYTES);

**c.** Write a program to find the minimum and maximum values of long using the MIN\_VALUE and MAX\_VALUE fields. (Hint: Use Long.MIN\_VALUE and Long.MAX\_VALUE).

=> System.out.println("Max value in Long ="+Long.MAX\_VALUE);

        System.out.println("Min value in Long ="+Long.MIN\_VALUE);

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

=> Long number=6749923040676l;

        System.out.println(Long.toString(number));

**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)).

=> String strNumber="6733402948409284";

        System.out.println(Long.parseLong(strNumber));

**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).

=>         String strNumber="Ab12Cd3";

        Long strNum=Long.parseLong("strNumber");

**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)).

=> long number=7789789678689l;

        System.out.println(Long.valueOf(number));

**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)).

=> String strNumber="7789789678689";

        System.out.println(Long.valueOf(strNumber));

**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)).

=> long n1=1123l,n2=9845l;

        System.out.println(Long.sum(n1,n2));

**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)).

=> long n1=1122l,n2=5566l;

        System.out.println("Minimun of two number is :"+Long.min(n1,n2));

        System.out.println("Maximum of two number is :"+Long.max(n1,n2));

**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)).

=> long n=7l;

        System.out.println("Binary of 7 is :"+Long.toBinaryString(n));

        System.out.println("Octal of 7 is :"+Long.toOctalString(n));

        System.out.println("Hexadecimal of 7 is :"+Long.toHexString(n));

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

=> long n=23984847723893432l;

        float  num=(float)n;

        double  num2=(double)n;

        int  num3=(int)n;

        System.out.println(num3);

#### **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.

=> It has two super class java .lang.Object and java.lang.Number

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

=> System.out.println("Number of bytes in float: "+Float.BYTES)

Number of bytes in float: 4

**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).

=>  System.out.println("Max value in float ="+Float.MAX\_VALUE);

    System.out.println("Min value in float="+Float.MIN\_VALUE);

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

=>float number=63721.738912f;

    String strFloat=Float.toString(number);

**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)).

=> String strNumber="7667.7098";

    float number=Float.parseFloat(strNumber);

**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).

=> String strNumber="Ab12Cd3";

    Float.parseFloat(strNumber);

**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)).

=> float number=3434.6789f;

    System.out.println(Float.valueOf(number));

**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)).

=> String strNumber ="4566.768";

    System.out.println(Float.valueOf(strNumber));

**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)).

=>    float f1=112.3f,f2=984.5f;

    System.out.println(Float.sum(f1,f2));

**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)).

=> float f1=112.2f,f2=556.6f;

    System.out.println("Minimun of two number is :"+Float.min(f1,f2));

    System.out.println("Maximum of two number is :"+Float.max(f1,f2));

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

=> float numFloat=(-25.0f);

    System.out.println("Square root of number is :"+Math.sqrt(numFloat));

Output:

Output is not a “Not a number” i.e. NaN.

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

=> float f1=0.0f,f2=0.0f;

    System.out.println("Diivision is :"+(f1/f2));

Output:

Output is NaN.

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

=>    float f1=67576.78f;

    int i=(int)f1;

    double d=(double) f1;

While converting float into non decimal datatype like int , the value after decimal point gets trimeed.

#### **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.

=> It has two super class java .lang.Object and java.lang.Number.

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

=> System.out.println("Number of bytes in double: "+Double.BYTES);

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

=> System.out.println("Max value in double ="+Double.MAX\_VALUE);

    System.out.println("Min value in double="+Double.MIN\_VALUE);

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

=> double number=233.789890;

    System.out.println("The number is: "+Double.toString(number));

**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)).

=>     String strNumber="678423.90723";

    System.out.println(Double.parseDouble(strNumber));

**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).

=> String strNumber="Ab12Cd3";

    System.out.println(Double.parseDouble(strNumber));

**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)).

=> double number=78.8098980;

    System.out.println("The number is: "+Double.valueOf(number));

**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)).

=> String strNumber="5675.9808";

    System.out.println(Double.valueOf(strNumber));

**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)).

=> double d1=112.3,d2=984.5;

    System.out.println(Double.sum(d1,d2));

**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)).

=> double d1=112.2,d2=556.6;

    System.out.println("Minimun of two number is :"+Double.min(d1,d2));

    System.out.println("Maximum of two number is :"+Double.max(d1,d2));

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

=> double numDouble=(-25.0);

    System.out.println("Square root of number is :"+Math.sqrt(numDouble));

Outputis not a NaN.

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

=> double d1=0.0,d2=0.0;

    System.out.println("Diivision is :"+(d1/d2));

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

=>  double d=67687.9778;

    float f=(float) d;

    int i=(int) d;

Converting double into other primitive type results trimmed result .

#### **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()).

=> boolean b=true;

        String bool=Boolean.toString(b);

        String.valueOf(bool);

        char c='h';

        String character=Character.toString(c);

        String.valueOf(character);

        byte numByte=40;

        String strBool=Byte.toString(numByte);

        String.valueOf(strBool);

        short numShort=5768;

        String strShort=Short.toString(numShort);

        String.valueOf(strShort);

        int numInt=565677;

        String strInt=Integer.toString(numInt);

        String.valueOf(strInt);

        long numLong=77645378293l;

        String strLong=Long.toString(numLong);

        String.valueOf(strLong);

        float numFloat=5656.7898f;

        String strFloat=Float.toString(numFloat);

        String.valueOf(strFloat);

        double numDouble=45456.989989008;

        String strDouble=Double.toString(numDouble);

        String.valueOf(strDouble);

#### **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).

#### **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).

=>import java.util.\*;

public class Assignment3\_Q10{

    public static void main(String args[]){

        Scanner sc=new Scanner(System.in);

        int n1,n2;

        System.out.println("Enter first integer=");

        n1=sc.nextInt();

        System.out.println("Enter Second integer=");

        n2=sc.nextInt();

        String operator="q";

        System.out.println("Enter operator=");

        System.out.println("+ -for addition\n - - for substraction\n \* -for multiplication\n/ -for division\n0-for exit");

        while(operator!="0"){

            operator=sc.nextLine();

            switch(operator){

            case "+":

                        System.out.println("Addition is="+(n1+n2));

                        break;

            case "-":

                        System.out.println("Substraction is="+(n1-n2));

                        break;

            case "\*":

                        System.out.println("Multiplication is="+(n1\*n2));

                        break;

            case "/":

                        System.out.println("Division is ="+(n1/n2));

                        break;

            default:

                        System.out.println("You have entered a wrong value.Please enter again");

        }

        }

        sc.close();

    }

}