Assignment no.2(Sandeep Sir)

Note: Consider the following before starting the assignment:

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

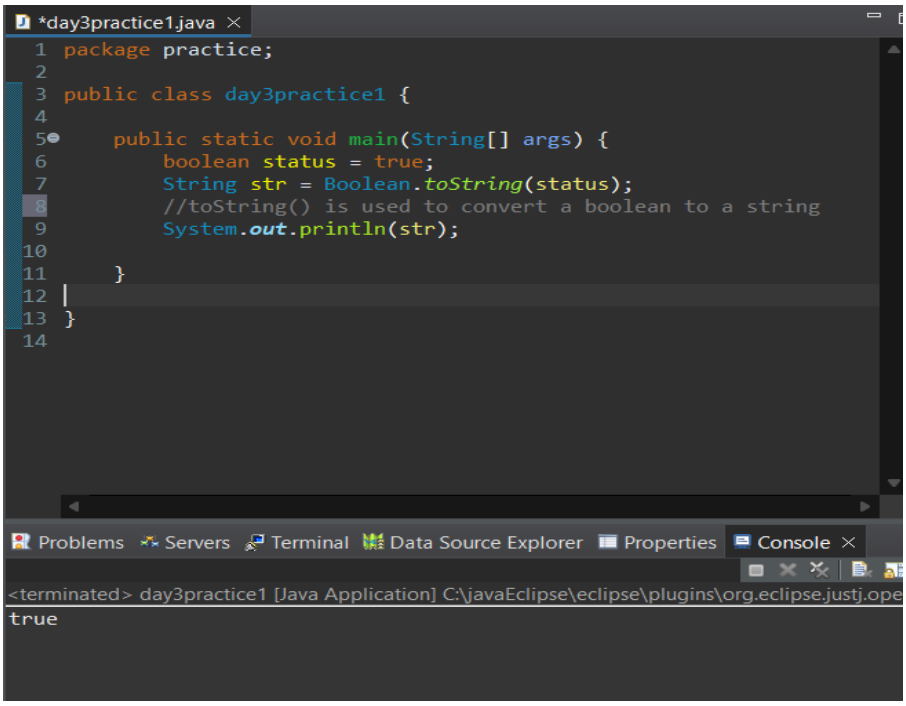
1. Working with `java.lang.Boolean`

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

Sol: Boolean is a class in java.lang(which is a root of all classes and also "cosmic superclass") which is most important classes and pre-imported in the java. The boolean datatype is not a class in it rather defined in wrapper class(java.lang.Boolean) and has a final keyword which signifies that it cannot have a child class.

Boolean datatype----→java.lang.Boolean (wrapper class)----→java.lang(superclass)

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



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

Sol:

```java
package practice;

public class day3practice1 {

    public static void main(String[] args) {
        String strStatus = "true";
        boolean bool = Boolean.parseBoolean(strStatus);
        //Boolean.parseBoolean() is used to convert String a non primitive datatype into a primitive datatype this process is called as unboxing. This would have not worked if the string was anything other than 'true'or'false'and would have just returned as false.
        System.out.println(bool);
```

```
<terminated> day3practice1 [Java Application] C:\javaEclipse\eclipse\plugins\org.eclipse.justj.open
true
```

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"`).

```java
package practice;

public class day3practice1 {

    public static void main(String[] args) {
        String strStatus = "0";
        boolean bool = Boolean.parseBoolean(strStatus);
        //Boolean.parseBoolean() is used to convert String a non primitive datatype into a primitive datatype this process is called as unboxing. This would have not worked if the string was anything other than 'true'or'false'and would have just returned as false.
        //In this we are trying to convert 0 to boolean which is not possible in java as boolean are only either true or false, so no concept of 0 or 1. so it returns false
        System.out.println(bool);
```

```
<terminated> day3practice1 [Java Application] C:\javaEclipse\eclipse\plugins\org.eclipse.justj.ope
false
```

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)`).

Sol:

```
day3practice1.java
1  package practice;
2
3  public class day3practice1 {
4
5      public static void main(String[] args) {
6          boolean status = true;
7          boolean bool = Boolean.valueOf(status);
8          //Boolean.parseBoolean() is used to convert String a non
   primitive datatype into a primitive datatype this process is
   called as unboxing. This would have not worked if the string was
   anything other than 'true'or'false'and would have just returned
   as false.
9          //In this we are trying to convert  to boolean which is
   not possible in java as boolean are only either true or false, so
   no concept of 0 or 1. so it returns false
10         System.out.println(bool);
11
12
13
14
15
Problems  Servers  Terminal  Data Source Explorer  Properties  Console
<terminated> day3practice1 [Java Application] C:\javaEclipse\eclipse\plugins\org.eclipse.justj.op
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)`).

```
*day3practice1.java
1  package practice;
2
3  public class day3practice1 {
4
5      public static void main(String[] args) {
6          String strstatus = "true";
7          boolean bool = Boolean.valueOf(strstatus);
8          //Boolean.valueOf() is same as Boolean.toString() which
   is used to convert String a non primitive datatype into a
   primitive datatype this process is called as unboxing. This would
   have not worked if the string was anything other than
   'true'or'false'and would have just returned as false.
9          //In this we are trying to convert  to boolean which is
   not possible in java as boolean are only either true or false, so
   no concept of 0 or 1. so it returns false
10         System.out.println(bool);
11
12
13
14
15
Problems  Servers  Terminal  Data Source Explorer  Properties  Console
<terminated> day3practice1 [Java Application] C:\javaEclipse\eclipse\plugins\org.eclipse.justj.ope
true
```

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

Sol:

Boolean.valueOf() is same as Boolean.toString() which is used to convert String a non-primitive datatype into a primitive datatype this process is called as unboxing. This would have not worked if the string was anything other than 'true' or 'false' and would have just returned as false.

2. Working with java.lang.Byte

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

Byte is a class in java.lang(which is a root of all classes and also “cosmic superclass”) which is most important classes and pre-imported in the java. The byte datatype is not a class in it rather defined in wrapper class(java.lang.Byte) and has a final keyword which signifies that it cannot have a child class.

byte datatype----→java.lang.Byte (wrapper class)----→java.lang(superclass)

byte ranges from -------→ -128 to 127

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

```
day3practice1.java    day3practice2.java
1  package practice;
2
3  public class day3practice2 {
4
5      public static void main(String[] args) {
6          byte b = Byte.BYTES;
7          System.out.println(b);
8          //Byte.BYTES is used to see how many bytes are use to
   store a byte. we know its 1 byte and so is the answer printed is
   also 1 which means 8 bits.
9
10     }
11
12 }
13
```

```
Problems  Servers  Terminal  Data Source Explorer  Properties  Console
<terminated> day3practice2 [Java Application] C:\javaEclipse\eclipse\plugins\org.eclipse.justj.op
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).

Sol:

```
1  package practice;
2
3  public class day3practice2 {
4
5      public static void main(String[] args) {
6          byte b = Byte.BYTES;
7          //System.out.println(b);
8          //Byte.BYTES is used to see how many bytes are use to
   store a byte. we know its 1 byte and so is the answer printed is
   also 1 which means 8 bits.
9          System.out.println(Byte.MIN_VALUE);
10         System.out.println(Byte.MAX_VALUE);
11         // this shows that the range of byte is
12         //byte-------->-128 to 127
13     }
14
15 }
16
```

```
Problems  Servers  Terminal  Data Source Explorer  Properties  Console
<terminated> day3practice2 [Java Application] C:\javaEclipse\eclipse\plugins\org.eclipse.justj.op
-128
127
```

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)).

Sol:

```java
package practice;

public class day3practice2 {

    public static void b2() {
        byte num = 45;
        String str = Byte.toString(num);
        System.out.println(str);
        // This is used to convert byte(primitive) into a string (non primitive) and so its a process called boxing.
    }

    public static void main(String[] args) {
        byte b = Byte.BYTES;
        //System.out.println(b);
```

Problems Servers Terminal Data Source Explorer Properties Console

```
<terminated> day3practice2 [Java Application] C:\javaEclipse\eclipse\plugins\org.eclipse.justj.oper
-128
127
45
```

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)).

Sol:

```java
package practice;

public class day3practice2 {


    public static void b3() {
        String strNumber = "45";
        byte b = Byte.parseByte(strNumber);
        System.out.println(b);
        //This is used to convert String( non primitive) into a byte(primitive) and so its a process called unboxing.
    }
```

Problems Servers Terminal Data Source Explorer Properties Console

```
<terminated> day3practice2 [Java Application] C:\javaEclipse\eclipse\plugins\org.eclipse.justj.ope
45
```

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).

Sol:

```
day3practice1.java   *day3practice2.java
1  package practice;
2
3  public class day3practice2 {
4
5
6      public static void b3() {
7          String strNumber = "Ab12Cd3";
8          byte b = Byte.parseByte(strNumber);
9          System.out.println(b);
10         //This is used to convert String( non primitive) into a
   byte(primitive) and so its a process called unboxing.
11         //this is because it is not a value which can be a byte
   as it includes alphabets it will not be processed as a byte
   datatype so the error
12     }
13
14
15
16
17
18
```

```
Outline
practice
day3practice2
b3() : void
b2() : void
main(String
```

```
Problems  Servers  Terminal  Data Source Explorer  Properties  Console
<terminated> day3practice2 [Java Application] C:\javaEclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.
Exception in thread "main" java.lang.NumberFormatException: For input string: "Ab12Cd3"
        at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.j
        at java.base/java.lang.Integer.parseInt(Integer.java:662)
        at java.base/java.lang.Byte.parseByte(Byte.java:195)
        at java.base/java.lang.Byte.parseByte(Byte.java:221)
        at CdacJava/practice.day3practice2.b3(day3practice2.java:8)
        at CdacJava/practice.day3practice2.main(day3practice2.java:45)
```

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)).

Sol:

```
1  package practice;
2
3  public class day3practice2 {
4
5
6      public static void b4() {
7          byte b=55;
8          byte by = Byte.valueOf(b);
9          System.out.println(by);
10     }
11
12
13
14
15
16
17
18
19
20
21
```

```
Problems  Servers  Terminal  Data Source Explorer  Prop
<terminated> day3practice2 [Java Application] C:\javaEclipse\eclipse\pl
55
```

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)).

Sol:

```java
package practice;

public class day3practice2 {

    public static void b5() {
        String strNumber = "89";
        byte b = Byte.parseByte(strNumber);
        System.out.println(b);
    }
```

```
<terminated> day3practice2 [Java Application] C:\javaEclipse\eclipse\plu
89
```

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

3. Working with java.lang.Short

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

Sol:

Short is a class in java.lang(which is a root of all classes and also "cosmic superclass") which is most important classes and pre-imported in the java. The short datatype is not a class in it rather defined in wrapper class(java.lang.Short) and has a final keyword which signifies that it cannot have a child class.

short datatype----→java.lang.Short (wrapper class)----→java.lang(superclass)

short ranges from----------→-32768 to 32767

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

sol:

```java
package practice;

public class day3practice3 {

    public static void main(String[] args) {
        System.out.println(Short.BYTES);
        //the output is 2bytes as we know that short datatype has a size of 2 bytes i.e 16 bits

    }

}
```

Problems Servers Terminal Data Source Explorer Properties Console

```
<terminated> day3practice3 [Java Application] C:\javaEclipse\eclipse\plugins\org.eclipse.justj.op
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).

Sol:

```java
package practice;

public class day3practice3 {

    public static void main(String[] args) {
        System.out.println(Short.BYTES);
        System.out.println(Short.MIN_VALUE);
        System.out.println(Short.MAX_VALUE);
        //the output is 2bytes as we know that short datatype has a size of 2 bytes i.e 16 bits
        //The range of short is -32768 to 32767.

    }

}
```

Problems Servers Terminal Data Source Explorer Properties Console

```
<terminated> day3practice3 [Java Application] C:\javaEclipse\eclipse\plugins\org.eclipse.justj.op
2
-32768
32767
```

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)).

Sol:

```java
package practice;

public class day3practice3 {

    public static void s1() {
        short num = 4578;
        String n1 = Short.toString(num);
        System.out.println(n1);
        // This is used to convert short(primitive) into a string (non primitive) and so its a process called boxing.
    }
```

```
<terminated> day3practice3 [Java Application] C:\javaEclipse\eclipse\plugins\org.eclipse.justj.o
4578
```

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)).

Sol:

```java
package practice;

public class day3practice3 {

    public static void s2() {
        String strNumber = "4586";
        short sh = Short.parseShort(strNumber);
        System.out.println(sh);
    }
```

```
4586
```

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).

Sol:

```java
    public static void s3() {
        String strNumber = "Ab12Cd3";
        short b = Short.parseShort(strNumber);
        System.out.println(b);
        //This is used to convert String( non primitive) into a sh
        //this is because it is not a value which can be a short a
    }
```

Problems Servers Terminal Data Source Explorer Properties Console

```
<terminated> day3practice3 [Java Application] C:\javaEclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win3
Exception in thread "main" java.lang.NumberFormatException: For input string: "Ab12Cd3"
        at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.
        at java.base/java.lang.Integer.parseInt(Integer.java:662)
        at java.base/java.lang.Short.parseShort(Short.java:138)
        at java.base/java.lang.Short.parseShort(Short.java:164)
        at CdacJava/practice.day3practice3.s3(day3practice3.java:13)
        at CdacJava/practice.day3practice3.main(day3practice3.java:48)
```

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

```java
package practice;

public class day3practice3 {

    public static void s4() {
        short number = 4587;
        short sh = Short.valueOf(number);
        System.out.println(sh);
    }
```

```
<terminated> day3practice3 [Java Application] C:\javaEclipse\eclipse
4587
```

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

```java
package practice;

public class day3practice3 {

    public static void s5() {
        String strNumber = "4586";
        short sh = Short.valueOf(strNumber);
        System.out.println(sh);
    }
```

```
4586
```

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

4. Working with java.lang.Integer

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

Sol:

Integer is a class in java.lang(which is a root of all classes and also "cosmic superclass") which is most important classes and pre-imported in the java. The int datatype is not a class in it rather defined in wrapper class(java.lang.Short) and has a final keyword which signifies that it cannot have a child class.

int datatype----→java.lang.Integer (wrapper class)----→java.lang(superclass)

int ranges from----------→-2147483648 to 2147483647

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

```
    public static void main(String[] args) {
        int b = Integer.BYTES;
        System.out.println(b);
```

```
4
```

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(Integer.MIN_VALUE);
    System.out.println(Integer.MAX_VALUE);
```

```
-2147483648
2147483647
```

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)).

```
    public static void b2() {
        int num = 455664;
        String str = Integer.toString(num);
        System.out.println(str);
```

```
455664
```

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)).

```java
public static void b1() {
    String strNumber = "6652";
    int b = Integer.parseInt(strNumber);
    System.out.println(b);
}
```

```
6652
```

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).

```java
public static void b3() {
    String strNumber = "Ab12Cd3";
    int b = Integer.parseInt(strNumber);
    System.out.println(b);
```

```
Exception in thread "main" java.lang.NumberFormatException: For input string: "Ab12Cd3"
        at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.j
        at java.base/java.lang.Integer.parseInt(Integer.java:662)
        at java.base/java.lang.Integer.parseInt(Integer.java:778)
        at CdacJava/practice.day3practice4.b3(day3practice4.java:21)
        at CdacJava/practice.day3practice4.main(day3practice4.java:57)
```

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

```java
public static void b4() {
    int b=55;
    int by = Integer.valueOf(b);
    System.out.println(by);
}
```

```
55
```

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

```java
public static void b5() {
    String strNumber = "89";
    int b = Integer.parseInt(strNumber);
    System.out.println(b);
}
```

```
89
```

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

```java
public static void b6() {
    int a=10;
    int b=20;
    System.out.println(Integer.sum(a, b));
}
```

```
30
```

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)).

```
public static void b7() {
    int a=10;
    int b=20;
    System.out.println(Integer.max(a, b));
}
```

```
20
```

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)).

```
public static void b7() {
    int a=7;
    System.out.println(Integer.toBinaryString(a));
    System.out.println(Integer.toOctalString(a));
    System.out.println(Integer.toHexString(a));
}
```

```
111
7
7
```

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

5. Working with java.lang.Long

a. Explore the [Java API documentation for java.lang.Long]() and observe its modifiers and super types.

Long is a class in java.lang(which is a root of all classes and also "cosmic superclass") which is most important classes and pre-imported in the java. The long datatype is not a class in it rather defined in wrapper class(java.lang.Short) and has a final keyword which signifies that it cannot have a child class.

long datatype----→java.lang.Long (wrapper class)----→java.lang(superclass)

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

```
public static void main(String[] args) {
    long b = Long.BYTES;
    System.out.println(b);
```

```
8
```

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(b);
System.out.println(Long.MIN_VALUE);
System.out.println(Long.MAX_VALUE);
```

```
-9223372036854775808
9223372036854775807
```

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)).

```
public static void b2() {
    long num = 455664;
    String str = Long.toString(num);
    System.out.println(str);
```

```
455664
```

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)).

```
public static void b1() {
    String strNumber = "66528";
    long b = Long.parseLong(strNumber);
    System.out.println(b);
}
```

```
66528
```

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).

```
public static void b3() {
    String strNumber = "Ab12Cd3";
    long b = Long.parseLong(strNumber);
    System.out.println(b);
```

```
Exception in thread "main" java.lang.NumberFormatException: For input string: "Ab12Cd3"
        at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.j
        at java.base/java.lang.Long.parseLong(Long.java:709)
        at java.base/java.lang.Long.parseLong(Long.java:832)
        at CdacJava/practice.day3practice5.b3(day3practice5.java:40)
        at CdacJava/practice.day3practice5.main(day3practice5.java:72)
```

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)).

```
public static void b4() {
    long b=558;
    long by = Long.valueOf(b);
    System.out.println(by);
}
```

```
558
```

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)).

```java
public static void b5() {
    String strNumber = "8955";
    long b = Long.valueOf(strNumber);
    System.out.println(b);
}
```

```
8955
```

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)).

```java
public static void b6() {
    long a=1123;
    long b=9845;
    System.out.println(Long.sum(a, b));
}
```

```
10968
```

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)).

```java
public static void b() {
    long a=1122;
    long b=5566;
    System.out.println(Long.max(a, b));
}
```

```
5566
```

l. 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)).

```java
public static void b7() {
    long a=7;
    System.out.println(Long.toBinaryString(a));
    System.out.println(Long.toOctalString(a));
    System.out.println(Long.toHexString(a));
}
```

```
111
7
7
```

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

6. Working with java.lang.Float

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

Float is a class in java.lang(which is a root of all classes and also “cosmic superclass”) which is most important classes and pre-imported in the java. The float datatype is not a class in it rather defined in wrapper class(java.lang.Float) and has a final keyword which signifies that it cannot have a child class.

float datatype----→java.lang.Float (wrapper class)----→java.lang(superclass)

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 static void main(String[] args) {
    float b = Float.BYTES;
    System.out.println(b);
```

```
4.0
```

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(Float.MIN_VALUE);
System.out.println(Float.MAX_VALUE);
```

```
1.4E-45
3.4028235E38
```

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 static void b2() {
    float num = 455.64f;
    String str = Float.toString(num);
    System.out.println(str);
```

```
455.64
```

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 static void b1() {
    String strNumber = "66.528";
    float b = Float.parseFloat(strNumber);
    System.out.println(b);
}
```

```
66.528
```

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 static void b3() {
    String strNumber = "Ab12Cd3";
    float b = Float.parseFloat(strNumber);
    System.out.println(b);
}
```

```
xception in thread "main" java.lang.NumberFormatException: For input string: "Ab12Cd3"
        at java.base/jdk.internal.math.FloatingDecimal.readJavaFormatString(FloatingDecimal.
        at java.base/jdk.internal.math.FloatingDecimal.parseFloat(FloatingDecimal.java:122)
        at java.base/java.lang.Float.parseFloat(Float.java:556)
        at CdacJava/practice.day3practice6.b3(day3practice6.java:39)
        at CdacJava/practice.day3practice6.main(day3practice6.java:61)
```

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 static void b4() {
    float b=558.2f;
    float by = Float.valueOf(b);
    System.out.println(by);
}
```

```
558.2
```

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

```
public static void b5() {
    String strNumber = "89.55";
    float b = Float.valueOf(strNumber);
    System.out.println(b);
}
```

```
89.55
```

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)).

```
public static void b6() {
    float a=112.3f;
    float b=984.5f;
    System.out.println(Float.sum(a, b));
```

```
1096.8
```

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 static void b() {
    float a=112.2f;
    float b=556.6f;
    System.out.println(Float.min(a, b));
    System.out.println(Float.max(a, b));
}
```

```
112.2
556.6
```

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

```java
public static void b8() {
    float a=-25.0f;
    System.out.println(Math.sqrt(a));
}
```

```
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).

```java
public static void b9() {
    float a=0.0f;
    float b=0.0f;
    System.out.println(a/b);
}
```

```
NaN
```

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

7. Working with `java.lang.Double`

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

double is a class in java.lang(which is a root of all classes and also "cosmic superclass") which is most important classes and pre-imported in the java. The double datatype is not a class in it rather defined in wrapper class(java.lang.Double) and has a final keyword which signifies that it cannot have a child class.

double datatype----→java.lang.Double (wrapper class)----→java.lang(superclass)

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

```java
public static void main(String[] args) {
    double b = Double.BYTES;
    System.out.println(b);
```

```
8.0
```

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`).

```java
    System.out.println(Double.MIN_VALUE);
    System.out.println(Double.MAX_VALUE);
```

```
4.9E-324
1.7976931348623157E308
```

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)`).

```java
public static void b2() {
    double num = 455.64f;
    String str = Double.toString(num);
    System.out.println(str);
}
```

```
455.6400146484375
```

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)`).

```java
public static void b1() {
    String strNumber = "66.528";
    double b = Double.parseDouble(strNumber);
    System.out.println(b);
}
```

```
66.528
```

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`).

```java
public static void b3() {
    String strNumber = "Ab12Cd3";
    double b = Double.parseDouble(strNumber);
    System.out.println(b);
}
```

```
Exception in thread "main" java.lang.NumberFormatException: For input string: "Ab12Cd3"
        at java.base/jdk.internal.math.FloatingDecimal.readJavaFormatString(FloatingDecimal
        at java.base/jdk.internal.math.FloatingDecimal.parseDouble(FloatingDecimal.java:110
        at java.base/java.lang.Double.parseDouble(Double.java:792)
        at CdacJava/practice.day3practice6.b3(day3practice6.java:51)
        at CdacJava/practice.day3practice6.main(day3practice6.java:73)
```

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)`).

```java
public static void b4() {
    double b=552558.2;
    double by = Double.valueOf(b);
    System.out.println(by);
}
```

```
552558.2
```

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

```java
public static void b5() {
    String strNumber = "89555.55";
    double b = Double.valueOf(strNumber);
    System.out.println(b);
}
```

```
89555.55
```

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)`).

```java
public static void b6() {
    double a=112.3;
    double b=984.5;
    System.out.println(Double.sum(a, b));
}
```

```
1096.8
```

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)`).

```java
public static void b() {
    double a=112.2d;
    double b=556.6d;
    System.out.println(Double.min(a, b));
    System.out.println(Double.max(a, b));
}
```

```
112.2
556.6
```

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

```java
public static void b8() {
    double a=-25.0;
    System.out.println(Math.sqrt(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).

```java
public static void b9() {
    double a=0.0;
    double b=0.0;
    System.out.println(a/b);
}
```

```
NaN
```

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

8. Conversion between Primitive Types and Strings

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

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

```java
public static void main(String[] args) {
    boolean b=true;
    char ch='s';
    byte by=14;
    short s=88;
    int i=8759;
    long l=5215522;
    float f=25.45f;
    double d=4587.5947;

    System.out.println(Boolean.toString(b));
    System.out.println(Character.toString(ch));
    System.out.println(Byte.toString(by));
    System.out.println(Short.toString(s));
    System.out.println(Integer.toString(i));
    System.out.println(Long.toString(l));
    System.out.println(Float.toString(f));
    System.out.println(Double.toString(d));

}
```

```
true
s
14
88
8759
5215522
25.45
4587.5947
```

```java
public static void main(String[] args) {
    boolean b=true;
    char ch='s';
    byte by=14;
    short s=88;
    int i=8759;
    long l=5215522;
    float f=25.45f;
    double d=4587.5947;

    System.out.println(Boolean.valueOf(b));
    System.out.println(Character.valueOf(ch));
    System.out.println(Byte.valueOf(by));
    System.out.println(Short.valueOf(s));
    System.out.println(Integer.valueOf(i));
    System.out.println(Long.valueOf(l));
    System.out.println(Float.valueOf(f));
    System.out.println(Double.valueOf(d));

}
```

```
true
s
14
88
8759
5215522
25.45
4587.5947
```

9. Default Values of Primitive Types

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

```java
package practice;

public class day3practice8 {
    static boolean b;
    static char ch;
    static byte by;
    static short s;
    static int i;
    static long l;
    static float f;
    static double d;
    public static void main(String[] args) {


        System.out.println(b);
        System.out.println(ch);
        System.out.println(by);
        System.out.println(s);
        System.out.println(i);
        System.out.println(l);
        System.out.println(f);
        System.out.println(d);

    }
```

```
false

0
0
0
0
0.0
0.0
```

10. Arithmetic Operations with Command Line Input

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

```java
package practice;
import java.util.Scanner;
/*
 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 day3practice10 {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter numbers and operation((+, -, *, /): ");
        int num1 = sc.nextInt();
        int num2 = sc.nextInt();
        char ch = sc.next().charAt(0);
        switch(ch) {
        case '+':
            System.out.println("addition is "+(num1+num2));
            break;
        case '-':
            System.out.println("Substract is "+(num1-num2));
            break;
        case '*':
            System.out.println("multiplication is "+(num1*num2));
            break;
        case '/':
            System.out.println("Division is "+(num1/num2));
            break;
        default:
            System.out.println("Invalid output");
        }

    }

}
```

```
Enter numbers and operation((+, -, *, /): 45 25 -
Substract is 20
```