

# **ASSIGNMENT -7**

***SUBMITTED BY***

***P.KRANTI ABHISHEK***



## **1. WRAPPER CLASS IN JAVA:**

The automatic conversion of primitive data type into its corresponding wrapper class is known as Autoboxing.

`valueOf()` :method of wrapper classes to convert the primitive into objects.

Primitive Type	Wrapper class
boolean	Boolean
char	Character
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double

## **WRAPPER CHARACTER CLASS METHODS:**

- **`toUpperCase(char ch)`** - Converts the specified character to uppercase.
- **`toLowerCase(char ch)`** - Converts the specified character to lowercase.
- **`toString(char ch)`** - Converts the specified character to a string.
- **`isDigit(char ch)`** - Checks if the specified character is a digit (0-9).
- **`isLetter(char ch)`** - Checks if the specified character is a letter.

These are the some of the methods of Character Wrapper class in java.

## CHARACTER WRAPPER CLASS PREOGRAM EXAMPLE:

The screenshot shows the Eclipse IDE with a project named 'CharacterIdentifier'. The 'Package Explorer' on the left lists several classes, including 'CheckAlphabet.java'. The main editor displays the code for 'CheckAlphabet.java', which uses nested control constructs to classify a character input by the user. The code includes imports, a class definition, a main method, and two helper methods: 'isLowerVowel' and 'isUpperVowel'. The 'Problems' tab on the right shows a warning for a 'terminated' statement, which is a JavaDoc comment.

```

1 import java.util.Scanner;
2
3 public class CheckAlphabet
4 {
5     public static void main(String[] args) {
6         Scanner obj = new Scanner(System.in);
7         char ch = obj.nextLine().charAt(0);
8         if (Character.isLowerCase(ch)) {
9             if (isLowerVowel(ch)) {
10                 System.out.println("LOWERCASE VOWEL");
11             }
12             else {
13                 System.out.println("LOWERCASE CONSONANT");
14             }
15         }
16         else if (Character.isUpperCase(ch)) {
17             if (isUpperVowel(ch)) {
18                 System.out.println("UPPER CASE VOWEL");
19             }
20             else {
21                 System.out.println("UPPERCASE CONSONANT");
22             }
23         }
24         else if (Character.isDigit(ch)) {
25             System.out.println("It is a number");
26         }
27         else {
28             System.out.println("IT IS A SPECIAL CHARACTER");
29         }
30     }
31     static boolean isLowerVowel(char ch)
32     {
33         return ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u';
34     }
35     static boolean isUpperVowel(char ch)
36     {
37         return ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U';
38     }
39 }
40

```

The 'Problems' tab on the right shows a warning for a 'terminated' statement, which is a JavaDoc comment: `<terminated> CheckAlphabet (Java Application) C:\Users\ABHIAMMA\p2\pool\plugins\o`. The console output shows 'UPPER CASE VOWEL'.

### ARRAYS IN JAVA AND ITS METHODS:

An Array is a collection of homogeneous elements in a sequential order and can be retrieved through index.

An Array is an object in java.

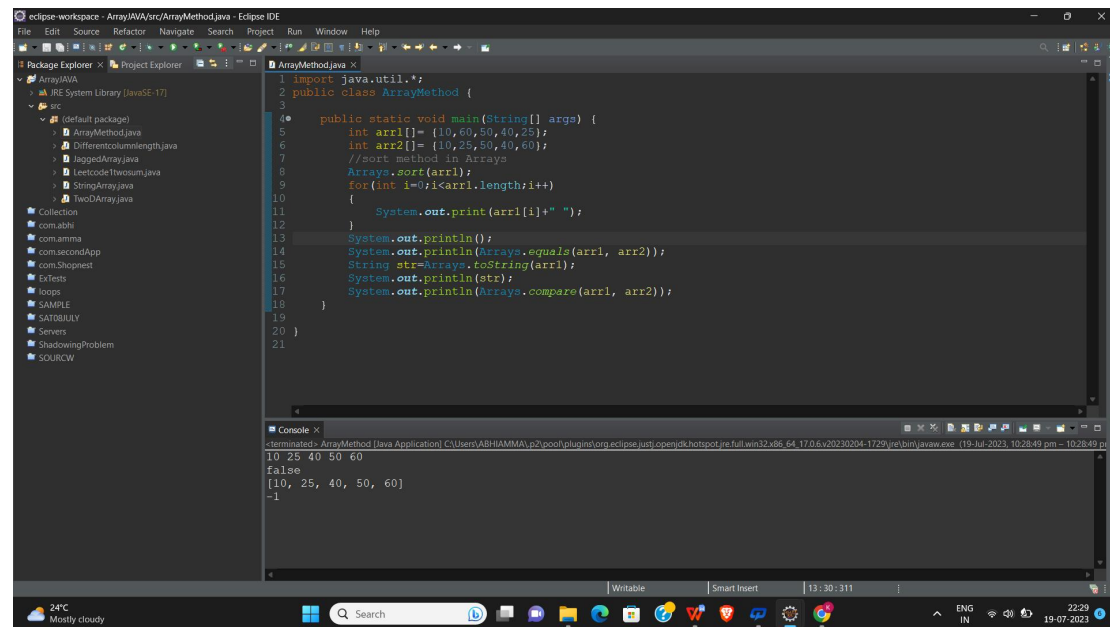
### ARRAY METHODS:

The methods will be present in the package.

import java.util.Arrays;

1. length- returns the length of the array
2. toString(Array) : It returns a string representation of the contents of this array
3. compare(array 1, array 2):Compares two arrays passed as parameters lexicographically.
4. sort(originalArray):Sorts the complete array in ascending order.
5. equals(array1, array2): Checks if both the arrays are equal or not.

### ARRAY METHODS PROGRAM:



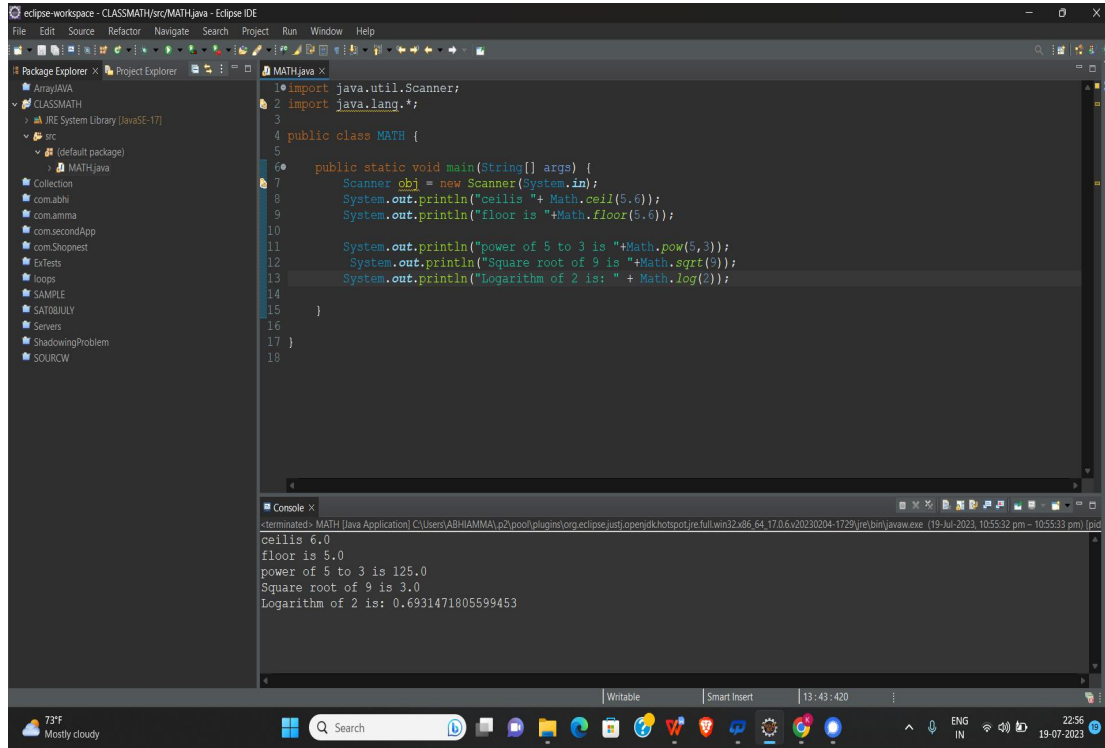
```
1 import java.util.*;
2 public class ArrayMethod {
3
4     public static void main(String[] args) {
5         int arr1[] = {10, 60, 50, 40, 25};
6         int arr2[] = {10, 25, 50, 40, 60};
7         //sort method in Arrays
8         Arrays.sort(arr1);
9         for (int i=0; i<arr1.length; i++)
10             System.out.print(arr1[i]+" ");
11     }
12
13     System.out.println();
14     System.out.println(Arrays.equals(arr1, arr2));
15     String str=Arrays.toString(arr1);
16     System.out.println(str);
17     System.out.println(Arrays.compare(arr1, arr2));
18 }
19
20 }
21
```

Console Output:

```
<terminated> ArrayMethod [Java Application] C:\Users\ABHIJAMMA\p2\pool\plugins\org.eclipse.jdt.launcher\org.eclipse.jdt.launcher.win32.x86_64.17.0.6.v20230204-1729\jet\bin\javaw.exe (19-Jul-2023, 10:28:49 pm - 10:28:49 pm)
10 25 40 50 60
false
[10, 25, 40, 50, 60]
-1
```

## NESTED CONTROL CONSTRUCTS

### MATH CLASS IN JAVA:



The screenshot displays the Eclipse IDE interface. The Package Explorer on the left shows a project named 'CLASSMATH' with a source folder 'src' containing a file 'MATH.java'. The main editor window shows the code for 'MATH.java'. The code imports 'java.util.Scanner' and 'java.lang.\*', and defines a public class 'MATH' with a 'main' method. The 'main' method uses 'Scanner' to read input and then uses various static methods from the 'Math' class: 'ceil', 'floor', 'pow', 'sqrt', and 'log'. The Console window at the bottom shows the output of the program, which matches the printed statements in the code.

```
1 import java.util.Scanner;
2 import java.lang.*;
3
4 public class MATH {
5
6     public static void main(String[] args) {
7         Scanner obj = new Scanner(System.in);
8         System.out.println("ceilis " + Math.ceil(5.6));
9         System.out.println("floor is " + Math.floor(5.6));
10
11         System.out.println("power of 5 to 3 is " + Math.pow(5,3));
12         System.out.println("Square root of 9 is " + Math.sqrt(9));
13         System.out.println("Logarithm of 2 is: " + Math.log(2));
14     }
15 }
16
17 }
18
```

Console Output:

```
<terminated> MATH [Java Application] C:\Users\ABHIAMMA\p2\pooof\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.17.0.v20230204-1729\jre\bin\javaw.exe (19-Jul-2023, 10:55:32 pm) [pid
ceilis 6.0
floor is 5.0
power of 5 to 3 is 125.0
Square root of 9 is 3.0
Logarithm of 2 is: 0.6931471805599453
```