

Lab6

1. Write the programme to sort the integers 8, 4, 3, 5, 6 and the alphabetical string C, O, I, P, U, in ascending order. Show the resulting output.

Code:

```
package lab6;

import java.util.*;

public class SorttheArray {

    public static void main(String[] args) {

        // TODO Auto-generated method stub

        int[] numbers= {8,4,3,5,6};

        char[] alphabeticals= {'C','O','I','P','U'};

        Arrays.sort(numbers);

        Arrays.sort(alphabeticals);

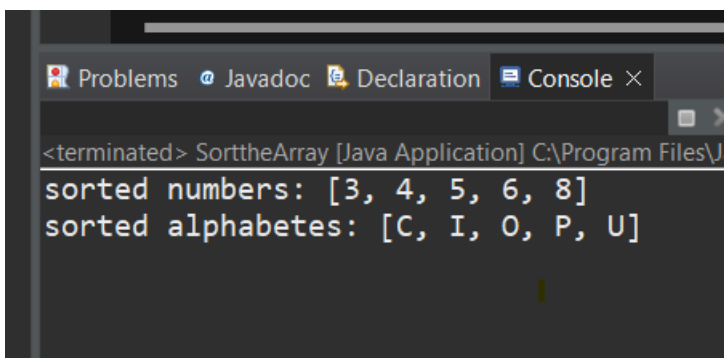
        System.out.println("sorted numbers: " +Arrays.toString(numbers));

        System.out.println("sorted alphabetes: " +Arrays.toString(alphabeticals));

    }

}
```

Output:

A screenshot of a Java IDE's console window. The window has tabs for 'Problems', 'Javadoc', 'Declaration', and 'Console'. The 'Console' tab is active, showing the output of a Java application named 'SorttheArray'. The output consists of two lines: 'sorted numbers: [3, 4, 5, 6, 8]' and 'sorted alphabetes: [C, I, O, P, U]'. The text is displayed in a monospaced font with syntax highlighting. The console title bar indicates the application is terminated and located at 'C:\Program Files\Ja...'.

```
<terminated> SorttheArray [Java Application] C:\Program Files\Ja
sorted numbers: [3, 4, 5, 6, 8]
sorted alphabetes: [C, I, O, P, U]
```

2. Write a Java program to implement the bubble sort algorithm to sort an array of integers in ascending order.

Code:

```
package lab6;
```

```
import java.lang.reflect.Array;
```

```
public class UsingBubbleSortAlgorithmToSortArray {
```

```
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        int[] Array= {23,45,56,21,34,123};  
        System.out.println("Unsorted Array:");  
        printArray(Array);  
        bubbleSort(Array);  
        System.out.println("\nSorted Array: ");  
        printArray(Array);  
    }
```

```
    public static void printArray(int[] array) {  
        // TODO Auto-generated method stub  
        for (int num:array) {  
            System.out.println(num+"");  
        }  
        System.out.println();  
    }
```

```
    public static void bubbleSort(int[] array) {  
        // TODO Auto-generated method stub  
        int n = array.length;  
        boolean swapped;  
        for (int i=0;i<n-1; i++) {  
            swapped = false;  
            for(int j=0; j<n-1-i;j++) {  
                if(array[j]> array[j + 1]) {  
                    int temp=array[j];  
                    array[j]=array[j+1];  
                    array[j+1]=temp;  
                    swapped=true;  
                }  
            }  
        }  
    }
```

Output:

```
<terminated> UsingBubbleSortAlgorithmToSortArray [Java]
Unsorted Array:
23
45
56
21
34
123

Sorted Array:
21
23
34
45
56
123
```

- 3. Write a program to input an array 10 elements and print the cube of prime numbers in it.**

Code:

```
package lab6;

import java.util.Scanner;

public class CubeOfPrimeNumbers {

    // Function to check if a number is prime
    public static boolean isPrime(int num) {
        if (num <= 1) {
```

```

        return false;
    }
    for (int i = 2; i <= Math.sqrt(num); i++) {
        if (num % i == 0) {
            return false;
        }
    }
    return true;
}

```

// Function to calculate the cube of a number

```

public static int cube(int num) {
    return num * num * num;
}

```

```

public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    int[] arr = new int[10];

    // Input 10 elements into the array
    System.out.println("Enter 10 elements:");
    for (int i = 0; i < 10; i++) {
        arr[i] = scanner.nextInt();
    }

    // Print the cube of prime numbers
    System.out.println("Cubes of prime numbers in the array:");
    for (int i = 0; i < 10; i++) {
        if (isPrime(arr[i])) {
            System.out.println("Cube of " + arr[i] + " is " + cube(arr[i]));
        }
    }
}

```

```

        scanner.close();
    }
}

```

Output:

```

<terminated> CubeOfPrimeNumbers [Java Application] C:\Program
Enter 10 elements:
2
4
6
9
7
5
5
4
3
3
Cubes of prime numbers in the array:
Cube of 2 is 8
Cube of 7 is 343
Cube of 5 is 125
Cube of 5 is 125
Cube of 3 is 27
Cube of 3 is 27

```

4. Write a java program to implement integer wrapper class methods.(any 3 methods)

Code:

```

package lab6;

public class IntegerWrapperMethodsDemo {

    public static void main(String[] args) {

        // Example 1: Using Integer.parseInt(String s)

        String numberStr = "123";

        int number = Integer.parseInt(numberStr);

        System.out.println("Parsed integer: " + number);

        // Example 2: Using Integer.toString(int i)

        int num = 456;

        String numStr = Integer.toString(num);
    }
}

```

```
System.out.println("String representation: " + numStr);
```

```
// Example 3: Using Integer.compareTo(Integer anotherInteger)
```

```
Integer int1 = 10;
```

```
Integer int2 = 20;
```

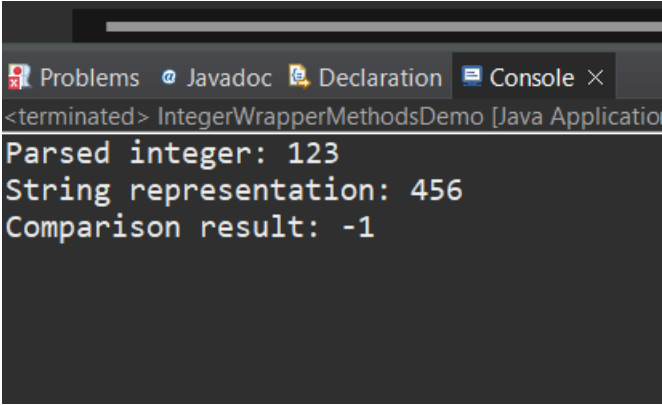
```
int comparisonResult = int1.compareTo(int2);
```

```
System.out.println("Comparison result: " + comparisonResult); // -1 if int1 < int2
```

```
}
```

```
}
```

Output:



```
<terminated> IntegerWrapperMethodsDemo [Java Application]
Parsed integer: 123
String representation: 456
Comparison result: -1
```

5. Write a java program to implement double wrapper class methods.(any 3 methods)

Code:

```
package lab6;
```

```
public class DoubleWrapperMethodsDemo {
```

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

```
        // Example 1: Using Double.parseDouble(String s)
```

```
        String doubleStr = "123.45";
```

```
        double number = Double.parseDouble(doubleStr);
```

```
        System.out.println("Parsed double: " + number);
```

```
        // Example 2: Using Double.toString(double d)
```

```
        double num = 456.78;
```

```
        String numStr = Double.toString(num);
```

```
        System.out.println("String representation: " + numStr);
```

```
// Example 3: Using Double.compareTo(Double anotherDouble)

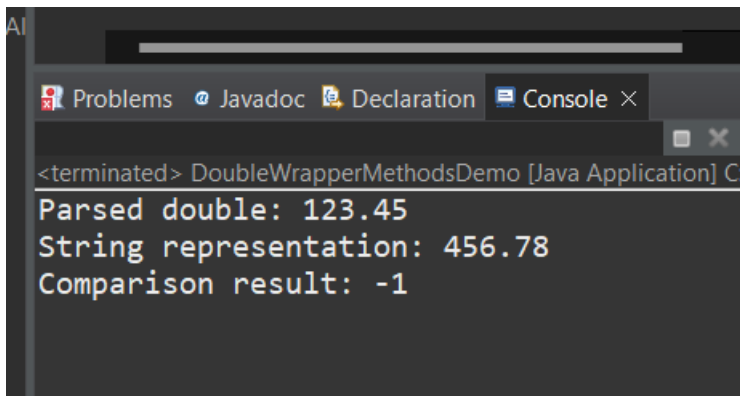
Double double1 = 10.5;

Double double2 = 20.5;

int comparisonResult = double1.compareTo(double2);

System.out.println("Comparison result: " + comparisonResult); // -1 if double1 <
double2
}
}
```

Output:



```
<terminated> DoubleWrapperMethodsDemo [Java Application] C:
Parsed double: 123.45
String representation: 456.78
Comparison result: -1
```

6. Write a java program to implement float wrapper class methods.(any 3 methods)

Code:

```
package lab6;

public class FloatWrapperMethodsDemo {

    public static void main(String[] args) {

        // Example 1

        String floatStr = "123.45";

        float number = Float.parseFloat(floatStr);

        System.out.println("Parsed float: " + number);


        // Example 2

        float num = 456.78f;

        String numStr = Float.toString(num);

        System.out.println("String representation: " + numStr);
```

```
// Example 3

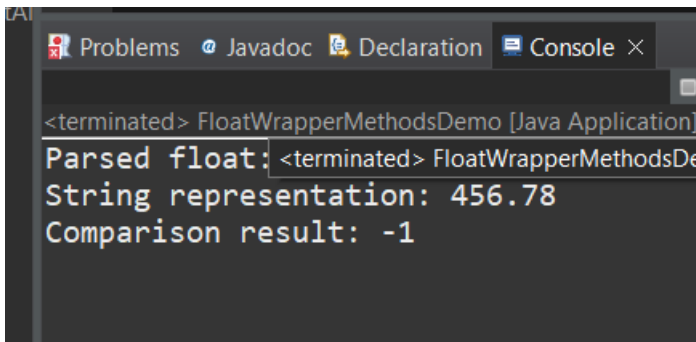
Float float1 = 10.5f;

Float float2 = 20.5f;

int comparisonResult = float1.compareTo(float2);

System.out.println("Comparison result: " + comparisonResult); // -1 if float1 <
float2
    }
}
```

Output:



7. Write a Java program to validate email addresses using regular expressions. The email should have the format `username@domain.com` where username and domain can contain alphanumeric characters, dots, and hyphens.

Code:

```
package lab6;

import java.util.regex.Matcher;
import java.util.regex.Pattern;
import java.util.Scanner;

public class EmailValidator {

    // Regular expression for validating email

    private static final String EMAIL_REGEX = "^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\". [a-zA-Z]{2,6}$";

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);
```



```

// Input email from user

System.out.print("Enter an email address: ");

String email = scanner.nextLine();


// Validate email
if (isValidEmail(email)) {

    System.out.println("The email address is valid.");

} else {

    System.out.println("The email address is invalid.");

}


scanner.close();

}


// Method to validate email using regex
public static boolean isValidEmail(String email) {

    Pattern pattern = Pattern.compile(EMAIL_REGEX);

    Matcher matcher = pattern.matcher(email);

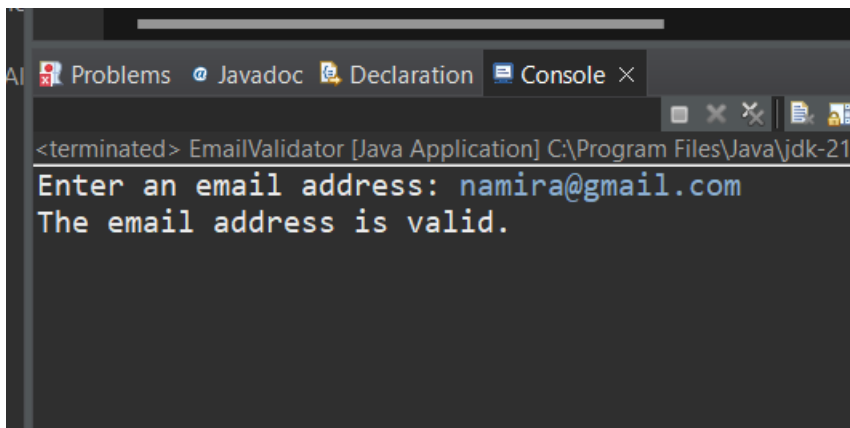
    return matcher.matches();

}

}

```

Output:



The screenshot shows a Java IDE with a console window open. The console displays the following text:

```

<terminated> EmailValidator [Java Application] C:\Program Files\Java\jdk-21\
Enter an email address: namira@gmail.com
The email address is valid.

```

The IDE interface includes tabs for Problems, Javadoc, Declaration, and Console. The console window is titled "EmailValidator [Java Application] C:\Program Files\Java\jdk-21\".

8. **Create a Java program to validate phone numbers. The format should be (xxx) xxx-xxxx where x is a digit.**

Code:

```
package lab6;

import java.util.regex.Matcher;
import java.util.regex.Pattern;
import java.util.Scanner;

public class PhoneNumberValidator {

    // Regular expression for validating phone numbers
    private static final String PHONE_REGEX = "^\\(\\d{3}\\) \\d{3}-\\d{4}$";

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        // Input phone number from user
        System.out.print("Enter a phone number (format: (xxx) xxx-xxxx): ");
        String phoneNumber = scanner.nextLine();

        // Validate phone number
        if (isValidPhoneNumber(phoneNumber)) {
            System.out.println("The phone number is valid.");
        } else {
            System.out.println("The phone number is invalid.");
        }

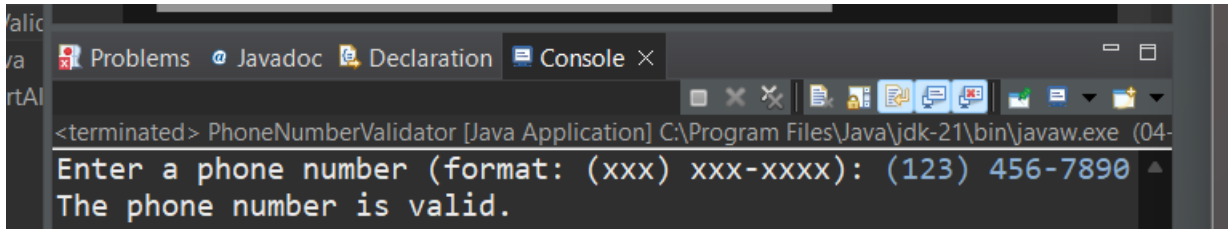
        scanner.close();
    }

    // Method to validate phone number using regex
    public static boolean isValidPhoneNumber(String phoneNumber) {

        Pattern pattern = Pattern.compile(PHONE_REGEX);
```

```
        Matcher matcher = pattern.matcher(phoneNumber);  
        return matcher.matches();  
    }  
}
```

Output:



The screenshot shows an IDE's console window with the following content:

```
<terminated> PhoneNumberValidator [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (04-  
Enter a phone number (format: (xxx) xxx-xxxx): (123) 456-7890  
The phone number is valid.
```