LAB4

1. Method Overloading: Write a class Calculator with overloaded methods add(). Implement add() methods that take:

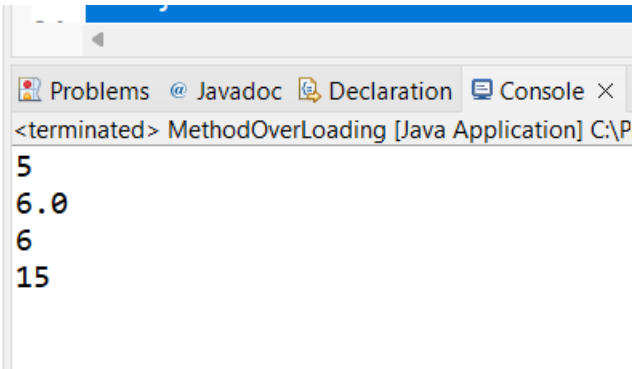
- **Two integers**
- **Two double values**
- **Three integers**
- **A variable number of integers**

Code:

```
package lab4;

public class MethodOverLoading {

	public int add(int a, int b) {

		return a+b;

	}

	public double add(double a, double b) {

		return a+b;

	}

	public int add(int a,int b,int c) {

		return a+b+c;

	}

	public int add(int... numbers) {

		int sum = 0;

		for(int num: numbers) {

			sum+= num;

		}

		return sum;

	}

	public static void main(String[] args) {

		MethodOverLoading calc = new MethodOverLoading();

		System.out.println(calc.add(2,3));

		System.out.println(calc.add(2.5,3.5));

		System.out.println(calc.add(1,2,3));
```

```
System.out.println(calc.add(1,2,3,4,5));}}
```

Output:



2. Super Keyword: Create a class Person with a constructor that accepts and sets name and age.

- Create a subclass Student that adds a grade property and initializes name and age using the super keyword in its constructor.

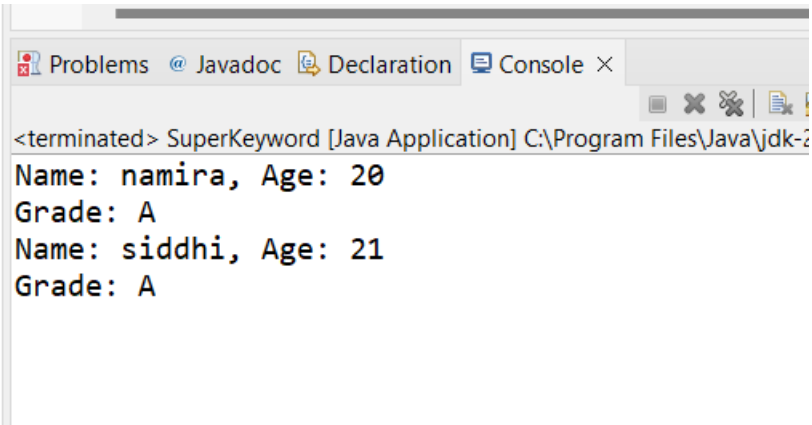
- Demonstrate the creation of Student objects and the usage of super to call the parent class constructor.

Code:

```
package lab4;

//Person class

class Person {

	protected String name;

	protected int age;

	public Person(String name, int age) {

		this.name = name;

		this.age = age;

	}

	public void displayInfo() {

		System.out.println("Name: " + name + ", Age: " + age);

	}

}

//base Student class

class Studeent extends Person {

	private String grade;
```

```
        public Studeent(String name, int age, String grade) {
                super(name, age);
                this.grade = grade;
        }
        @Override
        public void displayInfo() {
                super.displayInfo(); //using super keyword
                System.out.println("Grade: " + grade);
        }
}
//main class
public class SuperKeyword {
        public static void main(String[] args) {
                Studeent student1 = new Studeent("namira", 20, "A");
                Studeent student2 = new Studeent("siddhi", 21, "A");
                //calling methods
                student1.displayInfo();
                student2.displayInfo();
        }
}
```

Output:



3. Super Keyword: Create a base class Shape with a method draw() that prints "Drawing Shape".

- Create a subclass Circle that overrides draw() to print "Drawing Circle".

- Inside the draw() method of Circle, call the draw() method of the Shape class using super.draw().

- Write a main method to demonstrate calling draw() on a Circle object.

Code:

```
package lab4;

//shape.java

class Shape{

	public void draw() {

		System.out.println("Namira is drawing shape");

	}

}

//circle.java

class Circle extends Shape{

	@Override

	public void draw() {

		super.draw();

		System.out.println("Namira is drawing Circle");

	}

}

public class SuperKeyword2 {


	public static void main(String[] args) {

		Circle circle=new Circle();

		circle.draw();

	}


}
```

Output:

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> SuperKeyword2 [Java Application] C:\Progra
Namira is drawing shape
Namira is drawing Circle
```

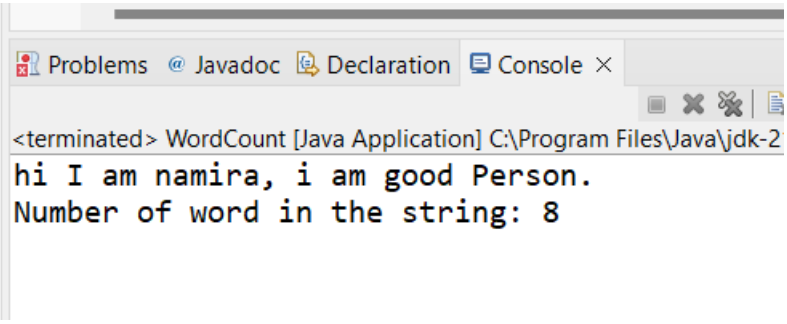
4. Write a Java Program to count the number of words in a String without using the Predefined method?

Code:

```
package lab4;

public class WordCount {

	public static int countWords(String str) {

		if(str == null || str.isEmpty()) {

			return 0;

		}

		int Count =0;

		boolean isWord = false;

		int endLine= str.length() - 1;

		char[] characters = str.toCharArray();


		for (int i=0; i< characters.length; i++) {

			if(Character.isLetter(characters[i]) && i != endLine) {

				isWord = true;

			}

			else if (!Character.isLetter(characters[i]) && isWord) {

				Count++;

				isWord = false;

			}

			else if(Character.isLetter(characters[i]) && i == endLine) {

				Count++;

			}
```

```
            }
            return Count;
        }
        public static void main(String[] args) {
            String line = "hi I am namira, i am good Person.";
            int numberOfWords = countWords(line);
            System.out.println(line);
            System.out.println("Number of word in the string: " + numberOfWords);

            // TODO Auto-generated method stub

        }

}
```

Output:



5. Write a Java Program to remove all white spaces from a String?

Code:

```
package lab4;

import java.util.StringTokenizer;

public class RemoveWhiteSpaces {
        public static String removeSpaces(String str) {
            if (str == null || str.isEmpty()) {
                return str;
```

```
        }
        StringTokenizer token =new StringTokenizer(str);
        StringBuilder result = new StringBuilder();

        while(token.hasMoreTokens()) {
            result.append(token.nextToken());
        }
        return result.toString();
    }

    public static void main(String[] args) {
        String input = "hi I am namira, i am good Person.";
        String noSpaces = removeSpaces(input);
        System.out.println("Original string: " + input);
        System.out.println("String without spaces: " + noSpaces);
    }
}
```

Output:

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> RemoveWhiteSpaces [Java Application] C:\Program Files\Java\jdk-21\bin\javav
Original string: hi I am namira, i am good Person.
String without spaces: hiIamnamira,iamgoodPerson.
```

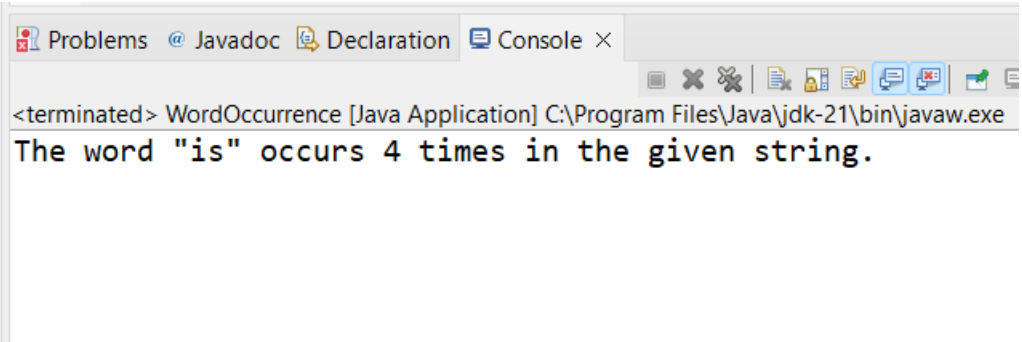
6. WAP to find occurrence of given in the given string.

Code:

```
package lab4;
public class WordOccurrence {
    public static int countOccurrences(String str, String word) {
        // lest check first string or word is empty or not
        if (str == null || word == null || str.isEmpty() || word.isEmpty()) {
            return 0;
```

```
        }
        int count = 0;
        int index = 0;
        while ((index = str.indexOf(word, index)) != -1) {
            count++;
            index += word.length();
        }
        return count;
    }
    public static void main(String[] args) {
        String input = "This is a test string. This string is for testing.";
        String word = "is";
        int occurrences = countOccurrences(input, word);
        System.out.println("The word \"" + word + "\" occurs " + occurrences + " times in the given string.");
    }
}
```

Output:



7. Write a java class to implement any 10 string methods:

● replace ● contains ● replaceAll ● indexOf ● substring ● Equals ● lastIndexOf ● startsWith

● endsWith ● EqualsIgnoreCase ● toLowerCase ● toUpperCase ● isEmpty ● Length ● split

Code:

```
package lab4;
public class StringMethodExample {
```

```
public static void main(String[] args) {
        String str= "hello i am the great person.";

        //using replace
        String replacedStr= str.replace("hello", "hi");
        System.out.println("replace:" + replacedStr + "\n");

        //using contains
        boolean containStr=str.contains("test");
        System.out.println("contain:" + containStr + "\n");

        //implementing replaceALL
        String replaceAllStr = str.replaceAll("am", "was");
        System.out.println("replaceAll: " + replaceAllStr + "\n");
        // implementing indexOf
        int indexOfStr = str.indexOf("test");
        System.out.println("indexOf: " + indexOfStr + "\n");
        // implementing substring
        String substringStr = str.substring(7, 12);
        System.out.println("substring: " + substringStr + "\n");
        // implementing equals
        boolean equalsStr = str.equals("Hello, World! This is a test string.");
        System.out.println("equals: " + equalsStr + "\n");
        // implementing lastIndexOf
        int lastIndexOfStr = str.lastIndexOf("is");
        System.out.println("lastIndexOf: " + lastIndexOfStr + "\n");
        // startsWith
        boolean startsWithStr = str.startsWith("Hello");
        System.out.println("startsWith: " + startsWithStr + "\n");
        // implementing endsWith
        boolean endsWithStr = str.endsWith("string.");
```

```
System.out.println("endsWith: " + endsWithStr + "\n");
// implementing equalsIgnoreCase
boolean equalsIgnoreCaseStr = str.equalsIgnoreCase("hello, world! this is a test
string.");
System.out.println("equalsIgnoreCase: " + equalsIgnoreCaseStr + "\n");
// implementing toLowerCase
String lowerCaseStr = str.toLowerCase();
System.out.println("toLowerCase: " + lowerCaseStr + "\n");
// implementing toUpperCase
String upperCaseStr = str.toUpperCase();
System.out.println("toUpperCase: " + upperCaseStr + "\n");
// implementing isEmpty
boolean isEmptyStr = str.isEmpty();
System.out.println("isEmpty: " + isEmptyStr + "\n");
// implementing length
int lengthStr = str.length();
System.out.println("length: " + lengthStr + "\n");
// implementing split
String[] splitStr = str.split(" ");
System.out.print("split: ");
for (String s : splitStr) {
        System.out.print(s + " | " + "\n");
}
    }

}
```

Output:

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> StringMethodExample [Java Application] C:\Program Files\Java\jdk-21\bi
replace:hi i am the great person.

contain:false

replaceAll: hello i was the great person.

indexOf: -1

substring:  am t

equals: false

lastIndexOf: -1

startsWith: false

endsWith: false

equalsIgnoreCase: false

toLowerCase: hello i am the great person.

toUpperCase: HELLO I AM THE GREAT PERSON.

isEmpty: false

length: 28

split: hello |
i |
am |
the |
```

8. Write a java program to implement string tokenizer.

Code:

```
package lab4;

import java.util.StringTokenizer;

public class StringTokenizerExample {

        public static void main(String[] args) {

                String str = "Hello, World! I am the nice person.you know!!!!";

                // Create a StringTokenizer with the default delimiter (whitespace)
```

```
        StringTokenizer tokenizer = new StringTokenizer(str);
        System.out.println("Tokens with default delimiter (whitespace):");
        while (tokenizer.hasMoreTokens()) {
            System.out.println(tokenizer.nextToken());
        }
        // Create a StringTokenizer with a custom delimiter
        String customStr = "Hello,World!This,is,a,test,string.";
        StringTokenizer customTokenizer = new StringTokenizer(customStr, ",!");
        System.out.println("\nTokens with custom delimiters (, and !):");
        while (customTokenizer.hasMoreTokens()) {
            System.out.println(customTokenizer.nextToken());
        }
    }
}
```

Output:

```
<terminated> StringTokenizerExample [Java Application] C:\Program Files\Java
Tokens with default delimiter (whitespace):
Hello,
World!
I
am
the
nice
person.you
know!!!!

Tokens with custom delimiters (, and !):
Hello
World
This
is
a
test
string.
```