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 Hellow;

public class Calculator {

	// Method to add two integers
	public int add(int a, int b) {
		return a + b;
	}

	// Method to add two double values
	public double add(double a, double b) {
		return a + b;
	}

	// Method to add three integers
	public int add(int a, int b, int c) {
		return a + b + c;
	}

	// Method to add a variable number of integers
	public int add(int... numbers) {
		int sum = 0;
		for (int number : numbers) {
			sum += number;
		}
		return sum;
	}

	public static void main(String[] args) {
		Calculator calc = new Calculator();

		// Test cases
		System.out.println(calc.add(10, 20));                // Two integers
		System.out.println(calc.add(10.5, 2.5));            // Two double values
		System.out.println(calc.add(10, 20, 30));           // Three integers
		System.out.println(calc.add(10, 20, 30, 40, 50)); // Variable number of integers
	}
}
```

OUTPUT:

```
<terminat
30
13.0
60
150
```

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 Hellow;

class Person {
	private String name;
	private int age;

	// Constructor for Person class
	public Person(String name, int age) {
		this.name = name;
		this.age = age;
	}

	// methods for name and age
	public String getName() {
		return name;
	}

	public int getAge() {
		return age;
	}
}

class Student extends Person {
	private String grade;

	// Constructor for Student class
	public Student(String name, int age, String grade) {
		// Call the parent class constructor
		super(name, age);
		this.grade = grade;
	}

	// method for grade
	public String getGrade() {
		return grade;
	}
}

public class Person1 {
	public static void main(String[] args) {
		// Create Student objects
		Student student1 = new Student("Sudeep", 23, "A");
		Student student2 = new Student("Ashish", 24, "B");

		// Demonstrate the usage of super to call the parent class constructor
		System.out.println("Student 1: Name = " + student1.getName() + ", Age = " +
student1.getAge() + ", Grade = " + student1.getGrade());
		System.out.println("Student 2: Name = " + student2.getName() + ", Age = " +
student2.getAge() + ", Grade = " + student2.getGrade());
	}
}
```

OUTPUT :

```
<terminated> Person1 [Java Application] C:\Users\Mr. User\.p2\p
Student 1: Name = Sudeep, Age = 23, Grade = A
Student 2: Name = Ashish, Age = 24, Grade = B
```

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 Hellow;

class Shape {
	// Method to draw a shape
	public void draw() {
		System.out.println("Drawing Shape");
	}
}

class Circle extends Shape {
	// Method to draw a circle, overrides the draw method of Shape
	@Override
	public void draw() {
		// Call the draw method of the Shape class
		super.draw();
		// Additional functionality for drawing a circle
		System.out.println("Drawing Circle");
	}
}

public class Shapes {
	public static void main(String[] args) {
		// Create a Circle object
		Circle circle = new Circle();
		// Call the draw method on the Circle object
		circle.draw();
	}
}
```

OUTPUT :

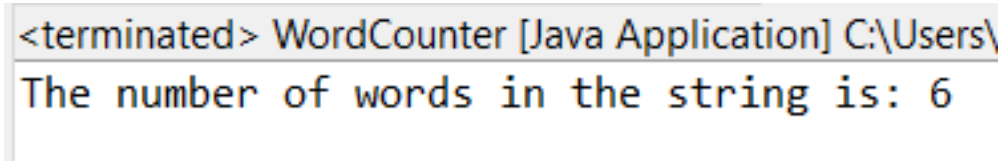
```
<terminated> Shapes
Drawing Shape
Drawing Circle
```

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

CODE :

```
package Hellow;

public class WordCounter {

	public static int countWords(String str) {
		if (str == null || str.isEmpty()) {
			return 0;
		}

		int wordCount = 0;
		boolean isWord = false;
		int endOfLine = str.length() - 1;
		char[] characters = str.toCharArray();

		for (int i = 0; i < characters.length; i++) {
			// Check if the current character is a letter or a digit
			if (Character.isLetterOrDigit(characters[i]) && i != endOfLine) {
				isWord = true;
				// If the current character is not a letter or digit and we are
inside a word, increase the word count
			} else if (!Character.isLetterOrDigit(characters[i]) && isWord) {
				wordCount++;
				isWord = false;
				// Check the last word in the string
			} else if (Character.isLetterOrDigit(characters[i]) && i == endOfLine) {
				wordCount++;
			}
		}
		return wordCount;
	}

	public static void main(String[] args) {
		String testString = "Hello, how are you doing today?";
		int count = countWords(testString);
		System.out.println("The number of words in the string is: " + count);
	}
}
```

OUTPUT :



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

CODE :

```
package Hellow;

public class RemoveWhiteSpaces {

	// Method to remove all white spaces from a string
	public static String removeWhiteSpaces(String str) {
		if (str == null || str.isEmpty()) {
			return str;
		}

		char[] characters = str.toCharArray();
		StringBuilder result = new StringBuilder();

		for (char ch : characters) {
			if (!Character.isWhitespace(ch)) {
				result.append(ch);
			}
		}

		return result.toString();
	}

	public static void main(String[] args) {
		String testString = "Hello how are you doing today?";
		String resultString = removeWhiteSpaces(testString);
		System.out.println("Original string: \"" + testString + "\"");
		System.out.println("String without white spaces: \"" + resultString + "\"");
	}
}
```

OUTPUT :

```
<terminated> RemoveWhiteSpaces [Java Application] C:\Users\Mr. User\.p2\pc
Original string: "Hello how are you doing today?"
String without white spaces: "Hellohowareyoudoingtoday?"
```

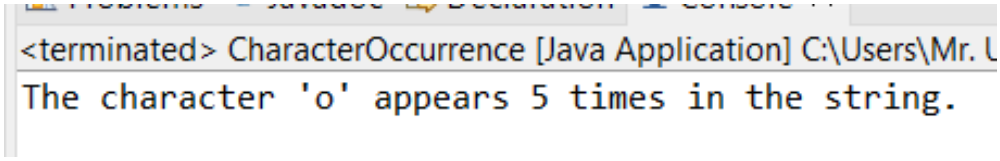
6. WAP to find occurrence of given in the given string.

CODE :

```
package Hellow;

public class CharacterOccurrence {

	public static int countOccurrences(String str, char ch) {
		if (str == null || str.isEmpty()) {
			return 0;
		}

		int count = 0;
		char[] characters = str.toCharArray();
```

```
                for (char currentChar : characters) {
                        if (currentChar == ch) {
                                count++;
                        }
                }

                return count;
        }

        public static void main(String[] args) {
                String testString = "Hello, how are you doing today?";
                char characterToFind = 'o';

                int occurrences = countOccurrences(testString, characterToFind);
                System.out.println("The character '" + characterToFind + "' appears " +
occurrences + " times in the 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 Hellow;

public class StringMethodsDemo {

        public static void main(String[] args) {
                String str = "Hello, how are you doing today?";
                String str2 = "hello, how are you doing today?";

                // replace
                String replacedStr = str.replace('o', 'a');
                System.out.println("replace: " + replacedStr);

                // contains
                boolean containsStr = str.contains("how");
                System.out.println("contains: " + containsStr);

                // replaceAll
                String replacedAllStr = str.replaceAll(" ", "_");
                System.out.println("replaceAll: " + replacedAllStr);

                // indexOf
                int indexOfStr = str.indexOf("you");
                System.out.println("indexOf: " + indexOfStr);
```

```
		// substring
		String substringStr = str.substring(7, 10);
		System.out.println("substring: " + substringStr);

		// equals
		boolean equalsStr = str.equals(str2);
		System.out.println("equals: " + equalsStr);

		// lastIndexOf
		int lastIndexOfStr = str.lastIndexOf("o");
		System.out.println("lastIndexOf: " + lastIndexOfStr);

		// startsWith
		boolean startsWithStr = str.startsWith("Hello");
		System.out.println("startsWith: " + startsWithStr);

		// endsWith
		boolean endsWithStr = str.endsWith("today?");
		System.out.println("endsWith: " + endsWithStr);

		// equalsIgnoreCase
		boolean equalsIgnoreCaseStr = str.equalsIgnoreCase(str2);
		System.out.println("equalsIgnoreCase: " + equalsIgnoreCaseStr);

		// toLowerCase
		String toLowerCaseStr = str.toLowerCase();
		System.out.println("toLowerCase: " + toLowerCaseStr);

		// toUpperCase
		String toUpperCaseStr = str.toUpperCase();
		System.out.println("toUpperCase: " + toUpperCaseStr);

		// isEmpty
		boolean isEmptyStr = str.isEmpty();
		System.out.println("isEmpty: " + isEmptyStr);

		// length
		int lengthStr = str.length();
		System.out.println("length: " + lengthStr);

		// split
		String[] splitStr = str.split(" ");
		System.out.println("split:");
		for (String word : splitStr) {
			System.out.println(word);
		}
	}
}
```

OUTPUT :

```
<terminated> StringMethodsDemo [Java Application] C:\Use
equalsIgnoreCase: true
toLowerCase: hello, how are you doing today?
toUpperCase: HELLO, HOW ARE YOU DOING TODAY?
isEmpty: false
length: 31
split:
Hello,
how
are
you
doing
today?
```

8. Write a java program to implement string tokenizer.

CODE :

```
package Hellow;

import java.util.StringTokenizer;

public class StringTokenizerDemo {

	public static void main(String[] args) {
		String str = "Hello, how are you doing today?";

		// Create a StringTokenizer with default delimiters (whitespace, tab, newline,
carriage return, form feed)
		StringTokenizer defaultTokenizer = new StringTokenizer(str);
		System.out.println("Tokens with default delimiters:");
		while (defaultTokenizer.hasMoreTokens()) {
			System.out.println(defaultTokenizer.nextToken());
		}

		// Create a StringTokenizer with a custom delimiter (comma and space)
		StringTokenizer customTokenizer = new StringTokenizer(str, ", ");
		System.out.println("\nTokens with custom delimiters (comma and space):");
		while (customTokenizer.hasMoreTokens()) {
			System.out.println(customTokenizer.nextToken());
		}

		// Create a StringTokenizer with custom delimiters and return delimiters as
tokens
		StringTokenizer returnDelimitersTokenizer = new StringTokenizer(str, ", ",
true);
		System.out.println("\nTokens with custom delimiters and returning delimiters:");
		while (returnDelimitersTokenizer.hasMoreTokens()) {
			System.out.println(returnDelimitersTokenizer.nextToken());
		}
	}
}
```

OUTPUT :

```
<terminated> StringT
Hello
,

how

are

you

doing

today?
```