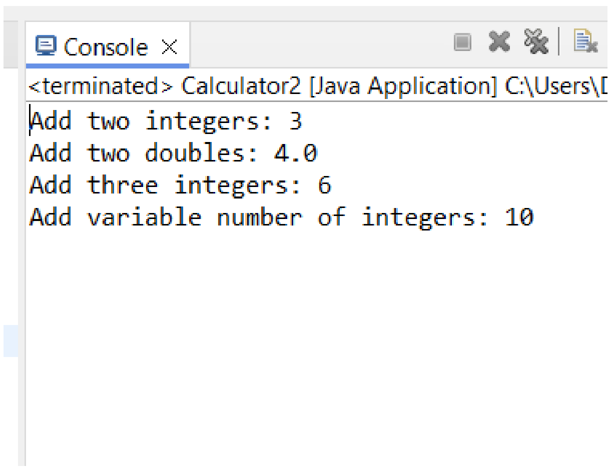
1.Method Overloading: Write a class Calculator with overloaded methods add(). Implement add() methods that take: - Two integers - Two double values - Three integers - A variable number of integers

Code:

```java
package hellow;
public class Calculator2 {
       // 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 num : numbers) {
              sum += num;
              }
              return sum;
              }
              public static void main(String[] args) {
              Calculator2 cal = new Calculator2();
              // Testing the add methods
              System.out.println("Add two integers: " + cal.add(1, 2));
              System.out.println("Add two doubles: " + cal.add(1.5, 2.5));
              System.out.println("Add three integers: " + cal.add(1, 2, 3));
              System.out.println("Add variable number of integers: " + cal.add(1, 
2, 3, 4));
              }

              // TODO Auto-generated method stub
}
```

Output:



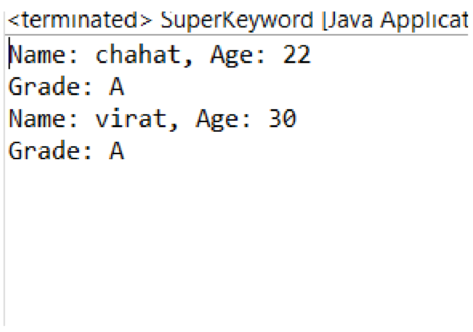
```
<terminated> Calculator2 [Java Application] C:\Users\[
Add two integers: 3
Add two doubles: 4.0
Add three integers: 6
Add variable number of integers: 10
```

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:

```java
package hellow;
class Persoon {
protected String name;
protected int age;
public Persoon(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 Persoon {
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("chahat", 22, "A");
Studeent student2 = new Studeent("virat", 30, "A");
//calling methods
student1.displayInfo();
student2.displayInfo();
}
}
```

Output:



```
<terminated> SuperKeyword [Java Applicat
Name: chahat, Age: 22
Grade: A
Name: virat, Age: 30
Grade: A
```

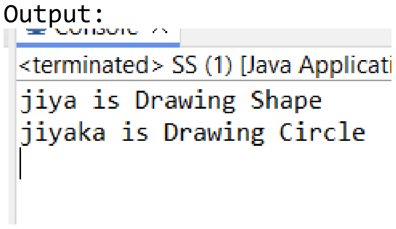
3.Super Keyword: Create a base class Shape with a method draw() that prints "Drawing Shape". - Create a subclass Circle that overrides draw() to print "Drawing Circle". - Inside the draw() method of Circle, call the draw() method of the Shape class using super.draw(). - Write a main method to demonstrate calling draw() on a Circle object.

Code:

```java
package hellow;

//Shape.java
class Shape {
public void draw() {
System.out.println("jiya is Drawing Shape");
}
}
//Circle.java
class Circle extends Shape {
@Override
public void draw() {
super.draw(); // Call the draw() method of Shape
System.out.println("jiyaka is Drawing Circle");
}
}
//Main.java
public class SS {
public static void main(String[] args) {
Circle circle = new Circle();
//This will call the draw method of Circle
circle.draw();
}
}
```

Output:



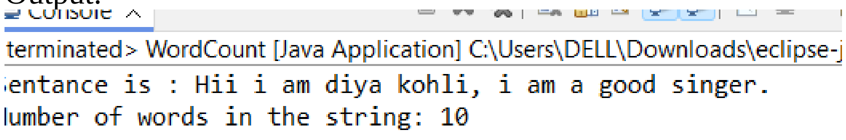
```
<terminated> SS (1) [Java Applicati
jiya is Drawing Shape
jiyaka is Drawing Circle
```

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

Code:

```
package hellow;
public class 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 the character is a letter, word = true.
if (Character.isLetter(characters[i]) && i != endLine) {
isWord = true;
}
// If the character isn't a letter and there have been letters before,
// count the word and set word = false.
else if (!Character.isLetter(characters[i]) && isWord) {
Count++;
isWord = false;
}
// Last word of the string; if it doesn't end with a non-letter, itcounts as a
word.
else if (Character.isLetter(characters[i]) && i == endLine) {
Count++;
}
}
return Count;
}
public static void main(String[] args) {
String line = "Hii i am diya kohli, i am a good singer.";
int numberOfWords = countWords(line);
System.out.println("Sentance is : "+line);
System.out.println("Number of words in the string: " + numberOfWords);
}
}
```

Output:

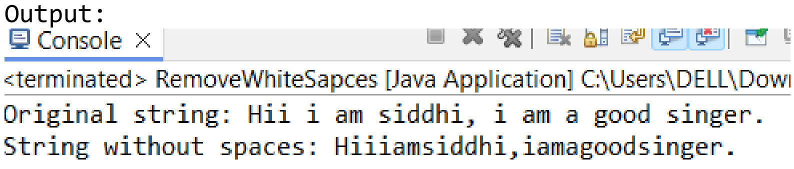


```
terminated> WordCount [Java Application] C:\Users\DELL\Downloads\eclipse-j
entance is : Hii i am diya kohli, i am a good singer.
lumber of words in the string: 10
```

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

```java
package hellow;
import java.util.StringTokenizer;
public class RemoveWhiteSapces {
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 = "Hii i am siddhi, i am a good singer.";
String noSpaces = removeSpaces(input);
System.out.println("Original string: " + input);
System.out.println("String without spaces: " + noSpaces);
}
}
```

Output:



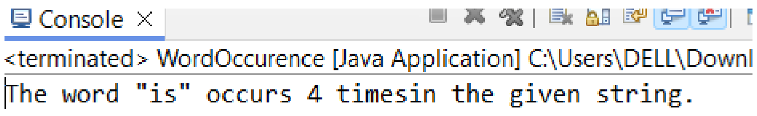
```
<terminated> RemoveWhiteSapces [Java Application] C:\Users\DELL\Dow
Original string: Hii i am siddhi, i am a good singer.
String without spaces: Hiiiamsiddhi,iamagoodsinger.
```

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

Code:
```java
package hellow;

public class WordOccurence {
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;
//whilw loop for finds occurance
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 + " timesin
the given string.");
}
}
```

Output:



```
<terminated> WordOccurence [Java Application] C:\Users\DELL\Downl
The word "is" occurs 4 timesin the given string.
```

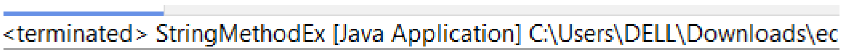
7. Write a java class to implement any 10 string methods: ● replace ● contains ● replaceAll ● indexOf ● substring ● Equals ● lastIndexOf ● startsWith ● endsWith ● EqualsIgnoreCase ● toLowerCase ● toUpperCase ● isEmpty ● Length ● split

Code:

```
package hellow;
public class StringMethodEx {
public static void main(String[] args) {
String str = "i am siya kohli it is my sentence.";
// using replace
String replacedStr = str.replace("World", "Java");
System.out.println("replace: " + replacedStr + "\n");
// using contains
boolean containsStr = str.contains("test");
System.out.println("contains: " + containsStr + "\n");
// implementing replaceAll
String replaceAllStr = str.replaceAll("is", "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");
```

```java
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:


<terminated> StringMethodEx [Java Application] C:\Users\DELL\Downloads\ec

```
replace: i am siya kohli it is my sentence.

contains: false

replaceAll: i am siya kohli it was my sentence.

indexOf: -1

substring: ya ko

equals: false

lastIndexOf: 19

startsWith: false

endsWith: false

equalsIgnoreCase: false

toLowerCase: i am siya kohli it is my sentence.

toUpperCase: I AM SIYA KOHLI IT IS MY SENTENCE.

isEmpty: false

length: 34

split: i |
am |
siya |
kohli |
it |
is |
my |
sentence. |
```

8. . Write a java program to implement string tokenizer.

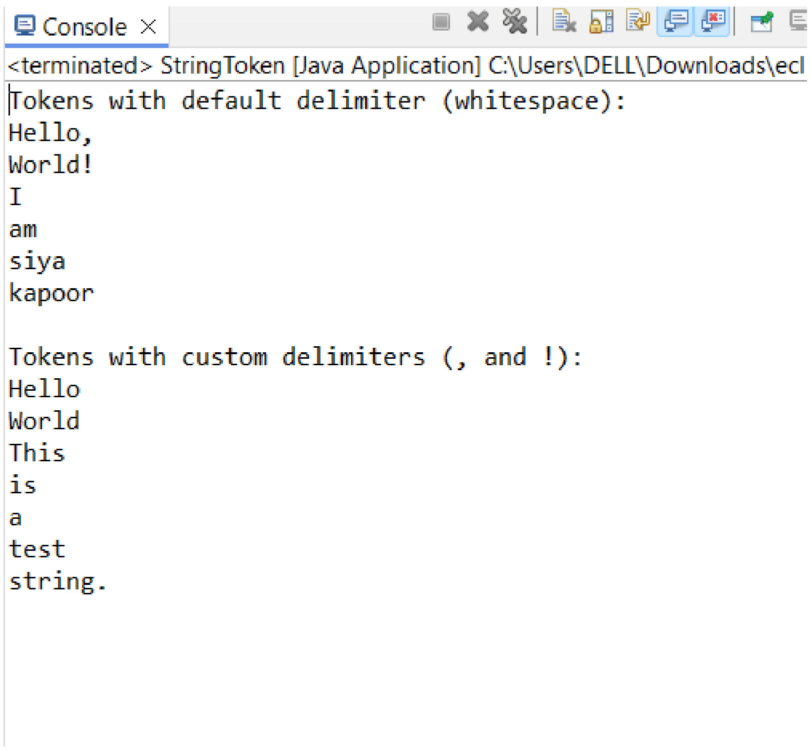
Code:

```java
package hellow;

import java.util.StringTokenizer;
public class StringToken{
public static void main(String[] args) {
String str = "Hello, World! I am siya kapoor";
// 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:



Console

<terminated> StringToken [Java Application] C:\Users\DELL\Downloads\ecl

```
Tokens with default delimiter (whitespace):
Hello,
World!
I
am
siya
kapoor

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