**LAB 4**

**Solve below questions:**

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

**PROGRAM:**

**package** mDemo;

**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** num : numbers) {

sum += num;

}

**return** sum;

}

}

**public** **class** MethodOverloading {

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

Calculator calculator = **new** Calculator();

// Test the methods

System.***out***.println("Sum of 2 and 3: " + calculator.add(2, 3));

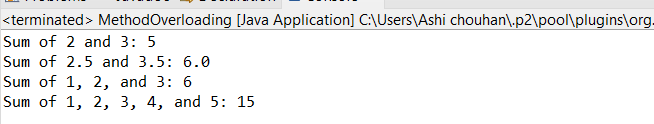
System.***out***.println("Sum of 2.5 and 3.5: " + calculator.add(2.5, 3.5));

System.***out***.println("Sum of 1, 2, and 3: " + calculator.add(1, 2, 3));

System.***out***.println("Sum of 1, 2, 3, 4, and 5: " + calculator.add(1, 2, 3, 4, 5));

}

}

****

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.

**PROGRAM:**

**package** Lab4;

**class** Person {

// Private variables for name and age

**private** String name;

**private** **int** age;

// Parameterized Constructor that accepts name and age

Person (String name, **int** age) {

**this**.name = name;

**this**.age = age;

}

**public** String getName() {

**return** name;

}

**public** **int** getAge() {

**return** age;

}

}

**class** Student **extends** Person {

**private** **char** grade;

// Parameterized Constructor that accepts name and age using super keyword

Student(String name, **int** age, **char** grade) {

**super**(name,age);

**this**.grade = grade;

}

**public** **char** getGrade() {

**return** grade;

}

}

**public** **class** SuperKeyword {

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

// Student class Object

Student s1 = **new** Student("Ashi", 21, 'A');

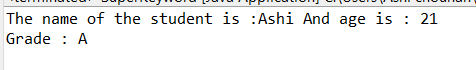
// Using Student class object to get name, age and grade

System.***out***.println("The name of the student is :" + s1.getName() + " And age is : " + s1.getAge());

System.***out***.println("Grade : " + s1.getGrade());

}

}

**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.

**PROGRAM:**

**package** Lab4;

**class** Shape {

**public** **void** draw() {

System.***out***.println("Drawing Shape");

}

}

**class** Circle **extends** Shape {

// Overriding draw() method of parent class Shape

**public** **void** draw() {

System.***out***.println("Drawing Circle");

// Calling Shape class draw() method using super keyword

**super**.draw();

}

}

**public** **class** DrawSuperkeywordDemo {

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

// Creating object of Circle class and calling draw() methos

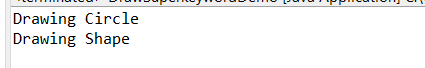
Circle c1 = **new** Circle();

c1.draw();

}

}

**OUTPUT:**

****

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

**PROGRAM:**

**package** Lab4;

**public** **class** WordCounter {

// Function that returns the numbers of words in a string

**public** **static** **int** wordCount(String str) {

// To check is the string is empty or null

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

**return** 0;

}

// Initializing count

**int** count = 0;

// Initializing isSpace for a space character

**boolean** isSpace = **true**;

// Loop to iterate through the length of the string

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

// if condition to check if the character at i index is not a space

**if**(str.charAt(i) != ' ') {

// Increment count if isSpace is true initialing and character is not a space

**if** (isSpace == **true**) {

count++;

}

// set isSpace to false

isSpace = **false**;

}

**else** {

// set isSpace to true if character at i is a space character

isSpace = **true**;

}

}

**return** count;

}

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

String str = "Hello everyone my name is ashi";

System.***out***.println(str);

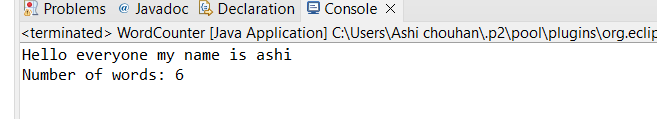
// Prints the Number of words

System.***out***.println("Number of words: " + *wordCount*(str));

}

}

**OUTPUT:**

****

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

**PROGRAM:**

**package** Lab4;

**public** **class** StringWhiteSpaceRemoval {

**public** **static** StringBuilder removeSpace(String string1) {

// if string is empty or null , return null

**if**(string1.isEmpty() || string1 == **null** ) {

**return** **null**;

}

// StringBuilder object to return the result as a string

StringBuilder result = **new** StringBuilder();

**for**(**int** i = 0; i<string1.length() ; i++) {

// Append character at i index if character is not a white space

**if**(string1.charAt(i) != ' ') {

result.append(string1.charAt(i));

}

}

**return** result;

}

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

String string1 = " Hello World ";

// Calling removeSpace function

StringBuilder result = *removeSpace*(string1);

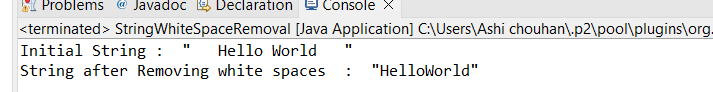
System.***out***.println("Initial String : " + " \"" + string1 + "\"");

System.***out***.println("String after Removing white spaces : " + " \"" + result + "\"");

}

}

**OUTPUT:**

****

6. WAP to find occurrence of given(substring) in the given string.

**PROGRAM:**

**package** Lab4;

**public** **class** SubStringOccurrenceCount {

// Function to count occurrences of 'substring' in 'mainString'

**public** **static** **int** countOccurrences(String mainString, String substring) {

// return 0 if string or substring is empty or null

**if** (mainString == **null** || substring == **null** || substring.isEmpty()) {

**return** 0; // No occurrences if input is invalid

}

**int** count = 0;

**int** index = 0;

// Loop to find and count occurrences of the substring

**while** ((index = mainString.indexOf(substring, index)) != -1) {

count++;

index += substring.length();

}

// Return the total count

**return** count;

}

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

// Declaring mainString and SubString

String mainString = "banana apple orange apple orange banana apple.";

String substring = "apple";

// Printing count of subString occurrences by calling countOccurrences function

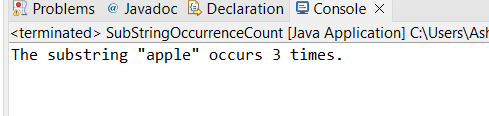
**int** count = *countOccurrences*(mainString, substring);

System.***out***.println("The substring \"" + substring + "\" occurs " + count + " times.");

}

}

**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

**PROGRAM:**

**package** Lab4;

**import** java.util.Arrays;

**public** **class** StringMethodsDemo {

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

String str = "Hello world";

// replace()

System.***out***.println("replace() : " + str.replace("w", "W"));

// contains()

System.***out***.println("contains() : " + str.contains("Hello"));

// replaceAll()

String str2 = str.replaceAll("world", "Ashi");

System.***out***.println("replaceAll() : " + str2);

// indexOf()

System.***out***.println("indexOf() : " + str.indexOf("o"));

// substring()

System.***out***.println("substring() : " + str.substring(5));

// equals()

System.***out***.println("equals() : " + str.equals(str2));

// lastIndexOf()

System.***out***.println("lastIndexOf() : " + str.lastIndexOf("o"));

// startsWith()

System.***out***.println("startsWith() : " + str.startsWith("Hello"));

// endsWith()

System.***out***.println("endsWith() : " + str.endsWith("Hello"));

// equalsIgnoreCase()

System.***out***.println("equalsIgnoreCase() : " + str.equalsIgnoreCase("HELLO WORLD"));

// toLowerCase()

System.***out***.println("toLowerCase() : " + str.toLowerCase());

// toUpperCase()

System.***out***.println("toUpperCase() : " + str.toUpperCase());

// isEmpty()

System.***out***.println("isEmpty() : " + str.isEmpty());

// length()

System.***out***.println("length() : " + str.length());

// split()

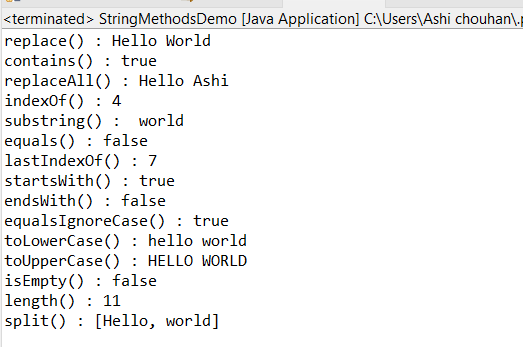
String[] splitStr = str.split(" ");

System.***out***.println("split() : " + Arrays.*toString*(splitStr));

}

}

**OUTPUT:**

****

8. Write a java program to implement string tokenizer.

**PROGRAM:**

**package** Lab4;

**import** java.util.StringTokenizer;

**public** **class** TokenizerDemo {

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

String str = "Hello everyone , nice to meet you all.";

// StringTokenizer object with comma as the delimiter

StringTokenizer tokenizer = **new** StringTokenizer(str, ", ");

// Printing all tokens

System.***out***.println("Tokens:");

// Loop to print all token is there are many tokens in a string

**while** (tokenizer.hasMoreTokens()) {

String token = tokenizer.nextToken();

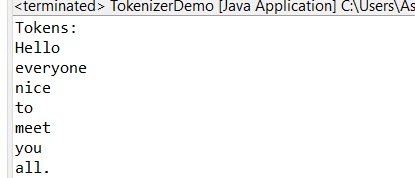
System.***out***.println(token);

}

}

}

**OUTPUT:**

****