A LABORATORY REPORT FILE ON

JAVA PROGRAMMING LAB

BACHELOR OF TECHNOLOGY

SUBJECT CODE: CS102491; 4TH SEMESTER

GUIDED BY

Mr. Rajeshwar Kumar Dewangan

(Assistant Professor)

Computer Science & Engineering

SUBMITTED BY

Name:- .....

University Roll No:- .....

Enrollment No:- .....

Specialization:- .....

Section:- .....

Computer Science & Engineering

SESSION: 2025-26

ज्ञानादेव तु कैवल्यम्

(Shri Gangajali Education Society)

Established 1999

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Shri Shankaracharya Technical Campus, Bhilai

Approved by AICTE, New Delhi

Constituent part of

Shri Shankaracharya Professional University, Bhilai

Khapri, Junwani Road, Bhilai, District-Durg, Chhattisgarh Pin Code- 490020, INDIA.

Ph. No.:0788-4088888, Fax No.: 0788-2298606, E-mail : sstc@sstc.ac.in, Web : www.sstc.ac.in

CERTIFICATE

Shri Shankaracharya Technical Campus, Bhilai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

This is certify that Mr./Ms. ..... whose University Roll No. ....., Enrollment No. ..... is a student of **B.TECH. (COMPUTER SCIENCE & ENGINEERING) – 4TH SEMESTER.** Under Specialization .....and Section ..... at **SHRI SHANKARACHARYA TECHNICAL CAMPUS, BHILAI CG INDIA.** has completed his/ her Practical work in the **SUBJECT :- JAVA PROGRAMMING LABORATORY & SUBJECT CODE :- (CS102491)** is accepted & approved after proper evaluation as a creditable work required as per the norms of **SHRI SHANKARACHARYA PROFESSIONAL UNIVERSITY, BHILAI CG INDIA.** in the laboratory of this college in the **SESSION 2025-26 .**

(Signature of the Guide)
Mr. Rajeshwar Kumar Dewangan
Assistant Professor (CSE)
SSTC, Bhilai

(Signature of the Student)
Place:-_____
Date:-_____

(Signature of the HOD)
Dr. Abha Choubey
Professor & Head of Department (CSE)
SSTC, Bhilai

ज्ञानादेव तु कैवल्यम्

Shri Shankaracharya Technical Campus, Bhilai

Student Name :- ---------- **University Roll No. :-** ----------

Branch/ Semester :- B.Tech. (CSE) - 4th SEM **Enrollment No. :-** ----------

Subject/ Code :- Java Programming Lab (CS102491) **Specialization/ Section :-** ----------

INDEX

Sr. No.	List of Programs/ Experiment Description	Page No.	Date of Performing	Date of Submission	Signature/ Remarks
1.	Write a java program to find the volume of a box having sides w, h, and d (width, height, and depth). Calculate the volume using the formula v=(w*h*d) and the surface area using s=2(wh+hd+dw). Use appropriate constructors for the class.	1-2			
2.	Develop a java program to illustrate a copy constructor so that a string may be duplicated into another variable, either by assignment or by copying.	3-4			
3.	Create a java base class called Shape. Apart from constructors, it should contain two methods: getXYValue() and showXYValue(), to accept and display coordinates. Create a subclass called Rectangle that contains a method called showXYValue() to display the length and breadth of the rectangle. Illustrate the concepts of Method Overriding and Constructor call sequence.	5-6			
4.	Write a java program that creates an abstract class called Dimension. Create two subclasses, Rectangle and Triangle. Include appropriate methods for both subclasses to calculate and display the area of the rectangle and triangle.	7-8			
5.	Write a java program that throws an ArithmeticException. Write another class (in a different file) that handles the exception.	9			
6.	Create a java user-defined Exception class that throws an exception when the user inputs marks greater than 100. Catch the exception and then rethrow it.	10			
7.	Write a java program to illustrate various String class methods.	11-14			
8.	Write a java program to illustrate various StringBuffer class methods.	15-16			
9.	Write a java program in which a MyThread class is created by extending the Thread class. In another class, create objects of the MyThread class and run them. In the run method, print "SSPU-BHILAI" 10 times. Identify each thread by setting a name.	17-18			
10.	Write a java program to illustrate various Thread methods and constructors.	19-20			
11.	Write a java program to implement a BankAccount class that illustrates the concept of Thread Synchronization.	21-22			
12.	Write a java program to create a text file using the Byte Stream class.	23			
13.	Write a java program to copy the contents of one file to another.	24-25			
14.	Write a java program to find the number of occurrences of vowels and consonants in a file.	26-27			
15.	Write a java program, which illustrates capturing of Mouse Events. Use Applet for this.	28-29			

ज्ञानादेव तु कैवल्यम्

Student Name :- ---------- **University Roll No. :-** ----------

Branch/ Semester :- B.Tech. (CSE) - 4th SEM **Enrollment No. :-** ----------

Subject/ Code :- Java Programming Lab (CS102491) **Specialization/ Section :-** ----------

INDEX

ज्ञानादेव तु कैवल्यम्

Program No-01

Aim: Write a java program to find the volume of a box having sides w, h, and d (width, height, and depth). Calculate the volume using the formula $v = (w * h * d)$ and the surface area using $s = 2(wh + hd + dw)$. Use appropriate constructors for the class.

Code: -

```
import java.util.Scanner; // Import the Scanner class

class Box {
    double width, height, depth;

    // Constructor to initialize the box dimensions
    Box(double w, double h, double d) {
        this.width = w;
        this.height = h;
        this.depth = d;
    }

    // Formula: V = (w * h * d)
    double calculateVolume() {
        return width * height * depth;
    }

    // Formula: A = 2(wd + hd + wh)
    double calculateSurfaceArea() {
        return 2 * (width * depth + height * depth + width *
height);
    }
}

public class ProgramNo1 {
    public static void main(String[] args) {
        // Create a Scanner object to read input
        Scanner input = new Scanner(System.in);

        // Asking user for dimensions
        System.out.print("Enter width: ");
        double w = input.nextDouble();

        System.out.print("Enter height: ");
        double h = input.nextDouble();

        System.out.print("Enter depth: ");
        double d = input.nextDouble();

        // Create the Box object using the constructor
        Box myBox = new Box(w, h, d);

        // Displaying results
        System.out.println("\n--- Box Details ---");
```

```
        System.out.println("Volume: " +
myBox.calculateVolume());
        System.out.println("Surface Area: " +
myBox.calculateSurfaceArea());

        // Close the scanner to prevent resource leaks
        input.close();
    }
}
```

Output of Program No-01: -

```
Enter width: 10
Enter height: 20
Enter depth: 30

--- Box Details ---
Volume: 6000.0
Surface Area: 2200.0
```

Program No-02

Aim: Develop a java program to illustrate a copy constructor so that a string may be duplicated into another variable, either by assignment or by copying.

Code: -

```
class StringDuplicator {
    String data;

    // Standard constructor to initialize with a value
    StringDuplicator(String data) {
        this.data = data;
    }

    // Copy Constructor: Takes an object of the same class as
a parameter
    StringDuplicator(StringDuplicator original) {
        this.data = original.data; // Duplicates the
reference/value
    }

    void display(String label) {
        System.out.println(label + ": " + data);
    }
}

public class ProgramNo2 {
    public static void main(String[] args) {
        // 1. Original Object
        StringDuplicator originalObj = new
StringDuplicator("Hello World!");

        // 2. Duplication via Copy Constructor
        StringDuplicator copiedObj = new
StringDuplicator(originalObj);

        // 3. Duplication via Assignment (Points to the same
memory location)
        StringDuplicator assignedObj = originalObj;

        // Display results
        originalObj.display("Original");
        copiedObj.display("Copied (via Constructor)");
        assignedObj.display("Assigned (via Reference)");

        // To prove it's a copy:
        System.out.println("\nIs copiedObj the same instance
as originalObj? " + (originalObj == copiedObj));
        System.out.println("Is assignedObj the same instance
as originalObj? " + (originalObj == assignedObj));
    }
```

```
}
```

Output of Program No-02: -

```
Original: Hello World!
Copied (via Constructor): Hello World!
Assigned (via Reference): Hello World!

Is copiedObj the same instance as originalObj? false
Is assignedObj the same instance as originalObj? true
```

Program No-03

Aim: Create a java base class called Shape. Apart from constructors, it should contain two methods: getXYValue() and showXYValue(), to accept and display coordinates. Create a subclass called Rectangle that contains a method called showXYValue() to display the length and breadth of the rectangle. Illustrate the concepts of Method Overriding and Constructor call sequence.

Code: -

```
import java.util.Scanner;

// Base Class
class Shape {
    double x, y;

    // Base Class Constructor
    Shape() {
        System.out.println("1. Base Class (Shape) Constructor
Called.");
    }

    // Method to accept coordinates
    void getXYValue(double x, double y) {
        this.x = x;
        this.y = y;
    }

    // Method to display coordinates
    void showXYValue() {
        System.out.println("Coordinates: (" + x + ", " + y +
")");
    }
}

// Subclass
class Rectangle extends Shape {
    double length, breadth;

    // Subclass Constructor
    Rectangle(double l, double b) {
        // super(); is implicitly called here by Java
        this.length = l;
        this.breadth = b;
        System.out.println("2. Subclass (Rectangle)
Constructor Called.");
    }

    // Overriding the showXYValue() method
    @Override
    void showXYValue() {
        // Calling the parent version of the method
```

```
        super.showXYValue();
        System.out.println("Rectangle Dimensions: Length = " +
length + ", Breadth = " + breadth);
        System.out.println("Area: " + (length * breadth));
    }
}

public class ProgramNo3 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter value of X coordinate: ");
        double x = sc.nextDouble();
        System.out.print("Enter value of Y coordinate: ");
        double y = sc.nextDouble();
        System.out.print("Enter value of Length: ");
        double l = sc.nextDouble();
        System.out.print("Enter value of Breadth: ");
        double b = sc.nextDouble();

        System.out.println("\n--- Initializing Objects ---");
        // Creating the subclass object
        Rectangle rect = new Rectangle(l, b);

        // Setting coordinates using base class method
        rect.getXYValue(x, y);

        System.out.println("\n--- Displaying Values (Method
Overriding) ---");
        // Calling the overridden method
        rect.showXYValue();

        sc.close();
    }
}
```

Output of Program No-03: -

```
Enter value of X coordinate: 25
Enter value of Y coordinate: 50
Enter value of Length: 25
Enter value of Breadth: 50

--- Initializing Objects ---
1. Base Class (Shape) Constructor Called.
2. Subclass (Rectangle) Constructor Called.

--- Displaying Values (Method Overriding) ---
Coordinates: (25.0, 50.0)
Rectangle Dimensions: Length = 25.0, Breadth = 50.0
Area: 1250.0
```

Program No-04

Aim: Write a java program that creates an abstract class called Dimension. Create two subclasses, Rectangle and Triangle. Include appropriate methods for both subclasses to calculate and display the area of the rectangle and triangle.

Code: -

```
import java.util.Scanner;

// Abstract Base Class
abstract class Dimension {
    double val1, val2;

    // Constructor to initialize dimensions
    Dimension(double a, double b) {
        this.val1 = a;
        this.val2 = b;
    }

    // Abstract method: No body here, implementation is forced
on subclasses
    abstract void displayArea();
}

// Subclass for Rectangle
class RectangleSubClass  extends Dimension {

    RectangleSubClass (double length, double breadth) {
        super(length, breadth);
    }

    // Implementing the abstract method
    @Override
    void displayArea() {
        double area = val1 * val2;
        System.out.println("Rectangle Area (Length * Breadth):
" + area);
    }
}

// Subclass for Triangle
class TriangleSubClass  extends Dimension {

    TriangleSubClass (double base, double height) {
        super(base, height);
    }

    // Implementing the abstract method
    @Override
    void displayArea() {
        double area = 0.5 * val1 * val2;
```

```
        System.out.println("Triangle Area (0.5 * Base *
Height): " + area);
    }
}

public class ProgramNo4 {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        // Input for Rectangle
        System.out.print("Enter Rectangle Length and Breadth:
");
        RectangleSubClass  rect = new RectangleSubClass
(input.nextDouble(), input.nextDouble());

        // Input for Triangle
        System.out.print("Enter Triangle Base and Height: ");
        TriangleSubClass  tri = new TriangleSubClass
(input.nextDouble(), input.nextDouble());

        System.out.println("\n--- Calculating Areas ---");
        rect.displayArea();
        tri.displayArea();

        input.close();
    }
}
```

Output of Program No-04: -

```
Enter Rectangle Length and Breadth: 78
5
Enter Triangle Base and Height: 34
3

--- Calculating Areas ---
Rectangle Area (Length * Breadth): 390.0
Triangle Area (0.5 * Base * Height): 51.0
```

Program No-05

Aim: Write a java program that throws an ArithmeticException. Write another class (in a different file) that handles the exception.

Code: -

```
public class Calculator {
    // Method that performs division
    public int divide(int numerator, int denominator) {
        // This will throw ArithmeticException automatically
if denominator is 0
        return numerator / denominator;
    }
}
```

```
public class ProgramNo5 {
    public static void main(String[] args) {
        Calculator myCalc = new Calculator();

        int a = 10;
        int b = 0; // This will cause the error

        System.out.println("Attempting to divide " + a + " by
" + b + "...");

        try {
            // Calling the method from the other class
            int result = myCalc.divide(a, b);
            System.out.println("Result: " + result);
        }
        catch (ArithmeticException e) {
            // Handling the specific exception
            System.err.println("Error Caught: You cannot
divide by zero!");
            System.err.println("Exception Message: " +
e.getMessage());
        }
        finally {
            System.out.println("Execution complete. (The
'finally' block always runs).");
        }
    }
}
```

Output of Program No-05: -

```
Attempting to divide 10 by 0...
Execution complete. (The 'finally' block always runs).
Error Caught: You cannot divide by zero!
Exception Message: / by zero
```

Program No-06

Aim: Create a java user-defined Exception class that throws an exception when the user inputs marks greater than 100. Catch the exception and then rethrow it.

Code: -

```
import java.util.Scanner;

// 1. User-defined Exception Class
class userDefineException  extends Exception {
    public userDefineException (String message) {
        super(message);
    }
}

public class ProgramNo6 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter your marks: ");
        int marks = sc.nextInt();

        try {
            // 2. Check the condition and THROW
            if (marks > 100) {
                throw new userDefineException ("Error: Marks
cannot be more than 100!");
            }
            System.out.println("Result: " + marks + " is a
valid marks.");

        } catch (userDefineException  e) {
            // 3. CATCH and print the error
            System.out.println(e.getMessage());
        }

        sc.close();
    }
}
```

Output of Program No-06: -

```
Enter your marks: 110
Error: Marks cannot be more than 100!
```

Program No-07

Aim: Write a java program to illustrate various String class methods.

Code: -

```
public class ProgramNo7 {
    public static void main(String[] args) {
        // Create a sample string
        String str = "Hello, World!";

        // 1. length() - Returns the length of the string
        System.out.println("1. Length of the string: " +
str.length());

        // 2. charAt(int index) - Returns the character at the
specified index
        System.out.println("2. Character at index 4: " +
str.charAt(4));

        // 3. substring(int beginIndex) - Returns a substring
from the specified index to the end
        System.out.println("3. Substring from index 7: " +
str.substring(7));

        // 4. substring(int beginIndex, int endIndex) -
Returns a substring within the specified range
        System.out.println("4. Substring from index 0 to 5: "
+ str.substring(0, 5));

        // 5. concat(String str) - Concatenates the specified
string to the end
        System.out.println("5. Concatenated string: " +
str.concat(" Welcome!"));

        // 6. toUpperCase() - Converts the string to uppercase
        System.out.println("6. Uppercase string: " +
str.toUpperCase());

        // 7. toLowerCase() - Converts the string to lowercase
        System.out.println("7. Lowercase string: " +
str.toLowerCase());

        // 8. trim() - Removes leading and trailing whitespace
        String strWithSpaces = "   Hello, World!   ";
        System.out.println("8. Trimmed string: '" +
strWithSpaces.trim() + "'");

        // 9. replace(char oldChar, char newChar) - Replaces
all occurrences of a character
        System.out.println("9. Replaced 'l' with 'L': " +
str.replace('l', 'L'));
```

```
        // 10. replace(CharSequence target, CharSequence
replacement) - Replaces all occurrences of a sequence
        System.out.println("10. Replaced 'World' with 'Java':
" + str.replace("World", "Java"));

        // 11. contains(CharSequence sequence) - Checks if the
string contains the specified sequence
        System.out.println("11. Contains 'World': " +
str.contains("World"));

        // 12. startsWith(String prefix) - Checks if the
string starts with the specified prefix
        System.out.println("12. Starts with 'Hello': " +
str.startsWith("Hello"));

        // 13. endsWith(String suffix) - Checks if the string
ends with the specified suffix
        System.out.println("13. Ends with '!': " +
str.endsWith("!"));

        // 14. indexOf(String str) - Returns the index of the
first occurrence of the specified substring
        System.out.println("14. Index of 'World': " +
str.indexOf("World"));

        // 15. lastIndexOf(String str) - Returns the index of
the last occurrence of the specified substring
        System.out.println("15. Last index of 'l': " +
str.lastIndexOf("l"));

        // 16. equals(Object obj) - Compares the string to the
specified object
        String str2 = "Hello, World!";
        System.out.println("16. Equals 'Hello, World!': " +
str.equals(str2));

        // 17. equalsIgnoreCase(String anotherString) -
Compares strings ignoring case
        String str3 = "hello, world!";
        System.out.println("17. Equals ignore case: " +
str.equalsIgnoreCase(str3));

        // 18. compareTo(String anotherString) - Compares two
strings lexicographically
        System.out.println("18. Compare to 'Hello, Java!': " +
str.compareTo("Hello, Java!"));

        // 19. isEmpty() - Checks if the string is empty
        String emptyStr = "";
        System.out.println("19. Is empty: " +
emptyStr.isEmpty());
```

```
        // 20. split(String regex) - Splits the string into an
array based on the regex
        String[] splitArray = str.split(", ");
        System.out.println("20. Split string: ");
        for (String s : splitArray) {
            System.out.println("   - " + s);
        }

        // 21. toCharArray() - Converts the string to a
character array
        char[] charArray = str.toCharArray();
        System.out.println("21. Character array: ");
        for (char c : charArray) {
            System.out.print(c + " ");
        }
        System.out.println();

        // 22. valueOf() - Converts different types to a
string
        int num = 123;
        String numStr = String.valueOf(num);
        System.out.println("22. String value of 123: " +
numStr);

        // 23. matches(String regex) - Checks if the string
matches the given regex
        System.out.println("23. Matches regex 'Hello.*': " +
str.matches("Hello.*"));

        // 24. format() - Formats a string
        String formattedStr = String.format("Formatted: %s,
Number: %d", str, 42);
        System.out.println("24. Formatted string: " +
formattedStr);

        // 25. join() - Joins strings with a delimiter
        String joinedStr = String.join(" - ", "Hello",
"World", "Java");
        System.out.println("25. Joined string: " + joinedStr);
    }
}
```

Output of Program No-07: -

1. Length of the string: 13
2. Character at index 4: o
3. Substring from index 7: World!
4. Substring from index 0 to 5: Hello
5. Concatenated string: Hello, World! Welcome!
6. Uppercase string: HELLO, WORLD!
7. Lowercase string: hello, world!
8. Trimmed string: 'Hello, World!'
9. Replaced 'l' with 'L': HeLLo, WorLd!
10. Replaced 'World' with 'Java': Hello, Java!
11. Contains 'World': true
12. Starts with 'Hello': true
13. Ends with '!': true
14. Index of 'World': 7
15. Last index of 'l': 10
16. Equals 'Hello, World!': true
17. Equals ignore case: true
18. Compare to 'Hello, Java!': 13
19. Is empty: true
20. Split string:
    - Hello
    - World!
21. Character array:
H e l l o , W o r l d !
22. String value of 123: 123
23. Matches regex 'Hello.*': true
24. Formatted string: Formatted: Hello, World!, Number: 42
25. Joined string: Hello - World – Java

Program No-08

Aim: Write a java program to illustrate various StringBuffer class methods.

Code: -

```
public class ProgramNo8 {
    public static void main(String[] args) {
        // 1. Creating a StringBuffer object
        StringBuffer sb = new StringBuffer("Hello");

        // 2. append() method
        sb.append(" World");
        System.out.println("After append: " + sb); // Output:
Hello World

        // 3. insert() method
        sb.insert(5, " Java");
        System.out.println("After insert: " + sb); // Output:
Hello Java World

        // 4. replace() method
        sb.replace(6, 10, "Programming");
        System.out.println("After replace: " + sb); // Output:
Hello Programming World

        // 5. delete() method
        sb.delete(6, 18);
        System.out.println("After delete: " + sb); // Output:
Hello World

        // 6. reverse() method
        sb.reverse();
        System.out.println("After reverse: " + sb); // Output:
dlroW olleH

        // 7. capacity() method
        System.out.println("Capacity: " + sb.capacity()); //
Output: 21 (default capacity is 16 + length of "Hello")

        // 8. ensureCapacity() method
        sb.ensureCapacity(50);
        System.out.println("New Capacity after ensureCapacity:
" + sb.capacity()); // Output: 50

        // 9. setLength() method
        sb.setLength(5);
        System.out.println("After setLength: " + sb); //
Output: dlroW

        // 10. charAt() method
        char ch = sb.charAt(0);
```

```
        System.out.println("Character at index 0: " + ch); //
Output: d

        // 11. setCharAt() method
        sb.setCharAt(0, 'D');
        System.out.println("After setCharAt: " + sb); //
Output: DlroW

        // 12. substring() method
        String subStr = sb.substring(1, 4);
        System.out.println("Substring from index 1 to 4: " +
subStr); // Output: lro

        // 13. indexOf() method
        int index = sb.indexOf("ro");
        System.out.println("Index of 'ro': " + index); //
Output: 2

        // 14. lastIndexOf() method
        int lastIndex = sb.lastIndexOf("o");
        System.out.println("Last index of 'o': " + lastIndex);
// Output: 4

        // 15. length() method
        int length = sb.length();
        System.out.println("Length of StringBuffer: " +
length); // Output: 5

        // 16. toString() method
        String str = sb.toString();
        System.out.println("String representation: " + str);
// Output: DlroW
    }
}
```

Output of Program No-08: -

1.
2. After append: Hello World
3. After insert: Hello Java World
4. After replace: Hello Programming World
5. After delete: Hello World
6. After reverse: dlroW olleH
7. Capacity: 44
8. New Capacity after ensureCapacity: 90
9. After setLength: dlroW
10. Character at index 0: d
11. After setCharAt: DlroW
12. Substring from index 1 to 4: lro
13. Index of 'ro': 2
14. Last index of 'o': 3
15. Length of StringBuffer: 5
16. String representation: DlroW

Program No-09

Aim: Write a java program in which a MyThread class is created by extending the Thread class. In another class, create objects of the MyThread class and run them. In the run method, print "SSPU-BHILAI" 10 times. Identify each thread by setting a name.

Code: -

```
// 1. Create the thread class by extending Thread
class MyThread extends Thread {

    // Constructor to set the thread name immediately
    MyThread(String name) {
        super(name); // Passes the name to the parent Thread
class
    }

    // 2. Override the run() method to define the task
    @Override
    public void run() {
        for (int i = 1; i <= 10; i++) {
            System.out.println(getName() + " (Iteration " + i
+ "): SSPU-BHILAI");

            try {
                // Adding a tiny sleep so we can see the
threads interleaving
                Thread.sleep(100);
            } catch (InterruptedException e) {
                System.out.println(e);
            }
        }
    }
}

public class ProgramNo9 {
    public static void main(String[] args) {
        // 3. Create objects of the MyThread class
        MyThread t1 = new MyThread("Thread-Alpha");
        MyThread t2 = new MyThread("Thread-Beta");

        // 4. Start the threads
        System.out.println("Starting Threads...");
        t1.start();
        t2.start();
    }
}
```

Output of Program No-09: -

```
Starting Threads...
Thread-Alpha (Iteration 1): SSPU-BHILAI
Thread-Beta (Iteration 1): SSPU-BHILAI
Thread-Beta (Iteration 2): SSPU-BHILAI
```

```
Thread-Alpha (Iteration 2): SSPU-BHILAI
Thread-Alpha (Iteration 3): SSPU-BHILAI
Thread-Beta (Iteration 3): SSPU-BHILAI
Thread-Alpha (Iteration 4): SSPU-BHILAI
Thread-Beta (Iteration 4): SSPU-BHILAI
Thread-Alpha (Iteration 5): SSPU-BHILAI
Thread-Beta (Iteration 5): SSPU-BHILAI
Thread-Alpha (Iteration 6): SSPU-BHILAI
Thread-Beta (Iteration 6): SSPU-BHILAI
Thread-Alpha (Iteration 7): SSPU-BHILAI
Thread-Beta (Iteration 7): SSPU-BHILAI
Thread-Beta (Iteration 8): SSPU-BHILAI
Thread-Alpha (Iteration 8): SSPU-BHILAI
Thread-Beta (Iteration 9): SSPU-BHILAI
Thread-Alpha (Iteration 9): SSPU-BHILAI
Thread-Alpha (Iteration 10): SSPU-BHILAI
Thread-Beta (Iteration 10): SSPU-BHILAI
```

Program No-10

Aim: Write a java program to illustrate various Thread methods and constructors.

Code: -

```
class MyThread1 extends Thread {
    // Constructor: Using the name parameter
    MyThread1(String name) {
        super(name);
    }
    // Override the run() method to define the task
    @Override
    public void run() {
        // Method: getName() - identifies the thread
        System.out.println(getName() + " is now STARTING.");

        try {
            // Method: sleep() - makes the thread wait
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            System.out.println(e);
        }

        System.out.println(getName() + " is now FINISHED.");
    }
}

public class ProgramNo10 {
    public static void main(String[] args) {
        // 1. Using Constructor to name threads
        MyThread1 t1 = new MyThread1("Thread-A");
        MyThread1 t2 = new MyThread1("Thread-B");

        // 2. Method: setPriority() - Give Thread-A more
importance
        t1.setPriority(Thread.MAX_PRIORITY);
        t2.setPriority(Thread.MIN_PRIORITY);

        // 3. Method: start() - Begin execution
        t1.start();
        t2.start();

        // 4. Method: isAlive() - Check if they are running
        System.out.println("Is Thread-A running? " +
t1.isAlive());
        System.out.println("Is Thread-B running? " +
t2.isAlive());

        try {
            // 5. Method: join() - Wait for Thread-A to finish
before continuing
            t1.join();
```

```
        } catch (Exception e) {}

        System.out.println("Main thread execution is
complete.");
    }
}
```

Output of Program No-10: -

```
Thread-A is now STARTING.
Thread-B is now STARTING.
Is Thread-A running? true
Is Thread-B running? true
Thread-A is now FINISHED.
Thread-B is now FINISHED.
Main thread execution is complete.
```

Program No-11

Aim: Write a java program to implement a BankAccount class that illustrates the concept of Thread Synchronization.

Code: -

```
class BankAccount {
    private int balance = 1000; // Starting balance

    // 'synchronized' keyword ensures only one thread can
withdraw at a time
    public synchronized void withdraw(String name, int amount)
{
        System.out.println(name + " is attempting to withdraw
₹" + amount);

        if (balance >= amount) {
            try {
                // Simulating time taken for processing
                Thread.sleep(500);
            } catch (InterruptedException e) {}

            balance -= amount;
            System.out.println(name + " successfully withdrew
₹" + amount);
            System.out.println("Remaining Balance: ₹" +
balance);
        } else {
            System.out.println(name + " failed! Not enough
balance for ₹" + amount);
            System.out.println("Current Balance: ₹" +
balance);
        }
        System.out.println("----------------------------------
-");
    }
}

class UserThread extends Thread {
    BankAccount account;
    String name;
    int amount;

    UserThread(BankAccount account, String name, int amount) {
        this.account = account;
        this.name = name;
        this.amount = amount;
    }

    public void run() {
        account.withdraw(name, amount);
    }
```

```
}

public class ProgramNo11 {
    public static void main(String[] args) {
        // One shared account
        BankAccount jointAccount = new BankAccount();

        // Two users trying to withdraw money at the same time
        UserThread user1 = new UserThread(jointAccount,
"Suresh", 700);
        UserThread user2 = new UserThread(jointAccount,
"Ramesh", 600);

        user1.start();
        user2.start();
    }
}
```

Output of Program No-11: -

```
Suresh is attempting to withdraw ₹700
Suresh successfully withdrew ₹700
Remaining Balance: ₹300
---------------------------------
Ramesh is attempting to withdraw ₹600
Ramesh failed! Not enough balance for ₹600
Current Balance: ₹300
---------------------------------
```

Program No-12

Aim: Write a java program to create a text file using the Byte Stream class.

Code: -

```
import java.io.*;

public class ProgramNo12 {
    public static void main(String[] args) {
        // Custom text to be written down in above file
        // Step 1: Storing text into String datatype
        String data = "Welcome to SSPU-BHILAI. This file was
created using Byte Streams.";

        // Try block to check if any exception/s occur
        try  {
            // Step 2: Creating object of the file and passing
local directory path of file as input
            FileOutputStream myFile  = new
FileOutputStream("myFile.txt");

            // Step 3: Convert the String into a byte array
            byte[] byteArray = data.getBytes();

            // Step 4: Write the byte array to the file
            myFile.write(byteArray);

            // Step 5: Close the file using close() method
            myFile.close();

            System.out.println("Success: The text file
'myFile.txt' has been created.");
        // Catch block to handle exceptions
        } catch (IOException e) {
            // Display and print the exception
            System.out.println("An error occurred: " +
e.getMessage());
        }
    }
}
```

Output of Program No-12: -

Success: The text file 'myFile.txt' has been created.

myFile.txt: -

Welcome to SSPU-BHILAI. This file was created using Byte Streams.

Program No-13

Aim: Write a java program to copy the contents of one file to another.

Code: -

```
import java.io.*;
import java.util.Scanner;

public class ProgramNo13 {

    // Method to handle the copying logic
    public static void copyContent(File source, File dest) {
        // Try-with-resources: Automatically closes 'in' and
'out' even if an error occurs
        try {
             FileInputStream in = new FileInputStream(source);
             FileOutputStream out = new
FileOutputStream(dest);

            int n;
            System.out.println("Copying in progress...");

            // read() function to read the byte of data until
End of File (-1)
            while ((n = in.read()) != -1) {
                // write() function to write the byte of data
                out.write(n);
            }
            System.out.println("Success: File copied from " +
source.getName() + " to " + dest.getName());

        } catch (FileNotFoundException e) {
            System.err.println("Error: The source file was not
found.");
        } catch (IOException e) {
            System.err.println("Error: An I/O error occurred
during the copy process.");
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // 1. Get Source File
        System.out.print("Enter the source filename (to read
from): ");
        String sourcePath = sc.nextLine();
        File sourceFile = new File(sourcePath);

        // 2. Get Destination File
        System.out.print("Enter the destination filename (to
write to): ");
```

```
            String destPath = sc.nextLine();
            File destFile = new File(destPath);

            // 3. Perform Copy
            if (sourceFile.exists()) {
                copyContent(sourceFile, destFile);
            } else {
                System.err.println("Execution Failed: Source file
does not exist!");
            }
            sc.close();
        }
}
```

Output of Program No-13: -

Enter the source filename (to read from): sourcefile.txt
Enter the destination filename (to write to): destinationfile.txt
Copying in progress...
Success: File copied from sourcefile.txt to destinationfile.txt

sourcefile.txt: -
This content is copied to the destination file.

destinationfile.txt: -
This content is copied to the destination file.

Program No-14

Aim: Write a java program to find the number of occurrences of vowels and consonants in a file.

Code: -

```
import java.util.Scanner;
import java.io.*;

public class ProgramNo14 {
    public static void main(String[] args) {
        // Scanner for keyboard input
        Scanner keyboard = new Scanner(System.in);
        System.out.print("Enter the filename: ");
        String fileName = keyboard.nextLine();

        int vCount = 0, cCount = 0;

        // Use try-with-resources to ensure fileReader and
fileScanner close automatically
        try {
            FileReader fileReading = new FileReader(fileName);
            Scanner fileScanner = new Scanner(fileReading);

            System.out.println("Reading from file: " +
fileName);

            while (fileScanner.hasNextLine()) {
                // Reading a line from the FILE (not the
keyboard)
                String sentence = fileScanner.nextLine();
                System.out.println("The sentence is: " +
sentence);

                // Convert to lowercase
                sentence = sentence.toLowerCase();

                for (int i = 0; i < sentence.length(); i++) {
                    char ch = sentence.charAt(i);

                    // Vowel check
                    if (ch == 'a' || ch == 'e' || ch == 'i' ||
ch == 'o' || ch == 'u') {
                        vCount++;
                    }
                    // Consonant check (a-z and not a vowel)
                    else if (ch >= 'a' && ch <= 'z') {
                        cCount++;
                    }
                }
            }
```

```
        } catch (FileNotFoundException e) {
            System.err.println("Error: The file '" + fileName
+ "' was not found.");
        } catch (IOException e) {
            System.err.println("Error: An issue occurred while
reading the file.");
        } finally {
            keyboard.close(); // Close the keyboard scanner
        }

        System.out.println("\n--- Final Counts ---");
        System.out.println("Number of vowels in the given
sentence is: " + vCount);
        System.out.println("Number of consonants in the given
sentence is: " + cCount);
    }
}
```

Output of Program No-14: -

Enter the filename: sentence.txt
Reading from file: sentence.txt
The sentence is: java is a high level, class based, object-oriented programming language that is designed to have as few implementation dependencies as possible.

--- Final Counts ---
Number of vowels in the given sentence is: 48
Number of consonants in the given sentence is: 72

Program No-15

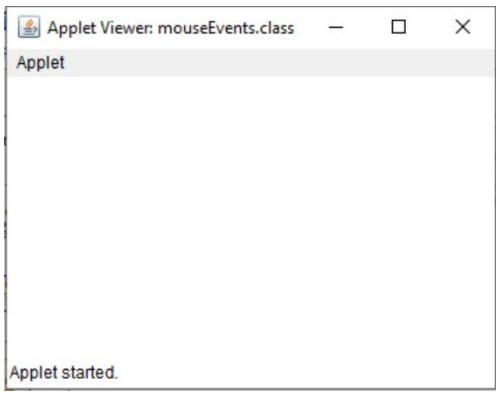
Aim: Write a java program, which illustrates capturing of Mouse Events. Use Applet for this.

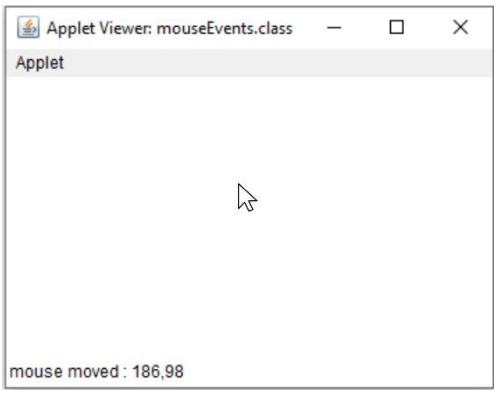
Code: -

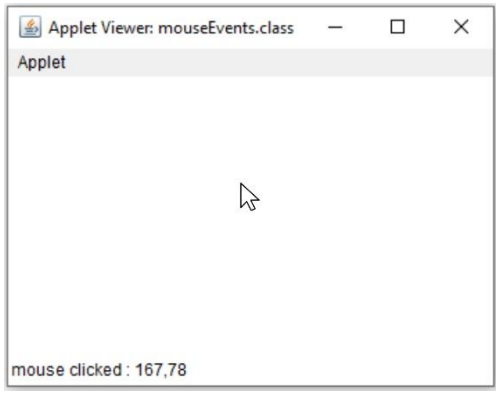
```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;

public class mouseEvents extends Applet implements
MouseListener, MouseMotionListener{
    public void init(){
        addMouseListener(this);
        addMouseMotionListener(this);
    }
    public void mouseEntered(MouseEvent e){
        showStatus("mouse entered : "+e.getX()+","+e.getY());
        for(int i=0;i<100000;i++);
    }
    public void mouseMoved(MouseEvent e){
        showStatus("mouse moved : "+e.getX()+","+e.getY());
    }
    public void mouseClicked(MouseEvent e){
        showStatus("mouse clicked : "+e.getX()+","+e.getY());
    }
    public void mousePressed(MouseEvent e){
        showStatus("mouse pressed : "+e.getX()+","+e.getY());
    }
    public void mouseReleased(MouseEvent e){
        showStatus("mouse released : "+e.getX()+","+e.getY());
    }
    public void mouseDragged(MouseEvent e){
        showStatus("mouse dragged : "+e.getX()+","+e.getY());
    }
    public void mouseExited(MouseEvent e){
        showStatus("mouse exited : "+e.getX()+","+e.getY());
    }
}
/* <body>
   <applet code = "mouseEvents.class" height = 200 width=200>
</applet>
   </body>  */
```

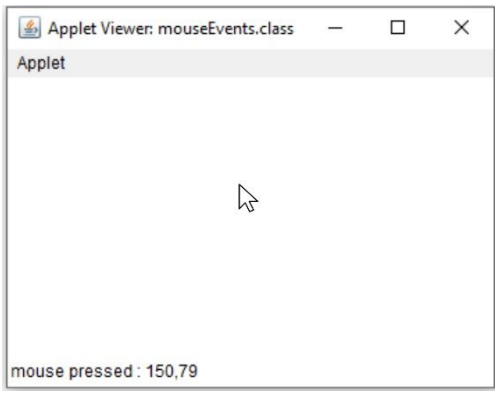
Output of Program No-15: -

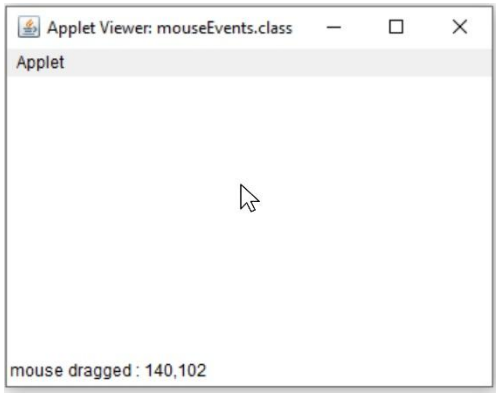
Remarks: All applet GUI runs on java jdk 8 (32bit) and older version. Because java.applet is deprecated with no replacement and marked for removal.

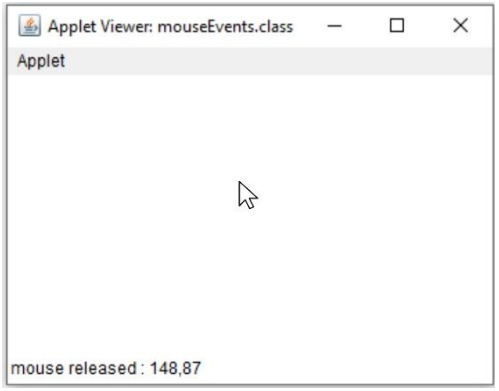


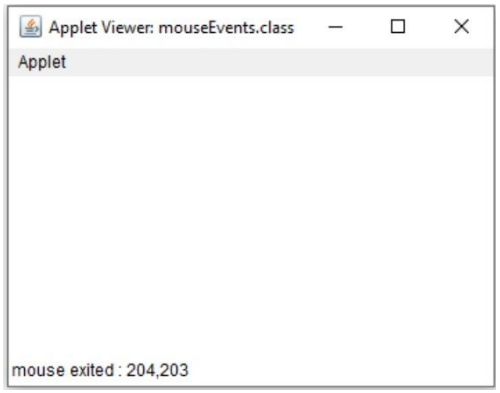












Program No-16

Aim: Write a java program using swing components which simulates simple calculator.

Code: -

```
import javax.swing.*;
import java.awt.event.*;
public class SwingCalci implements ActionListener {
    JFrame frame;
    JTextField textField;
    JButton button1, button2, button3, button4, button5,
button6, button7, button8, button9, button0, buttonDot,
buttonAdd, buttonSub, buttonMul, buttonDiv, buttonEql,
buttonDel, buttonClr;
    double a, b, result;
    int operator;
    SwingCalci(){
        frame=new JFrame("Swing-Calculator");
        frame.setLayout(null);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setBounds(20, 20, 300, 350);
        frame.setResizable(false);
        textField=new JTextField();
        textField.setBounds(40,40,200,50);
        button1=new JButton("1");
        button2=new JButton("2");
        button3=new JButton("3");
        button4=new JButton("4");
        button5=new JButton("5");
        button6=new JButton("6");
        button7=new JButton("7");
        button8=new JButton("8");
        button9=new JButton("9");
        button0=new JButton("0");
        buttonDot=new JButton(".");
        buttonAdd =new JButton("+");
        buttonSub=new JButton("-");
        buttonMul=new JButton("*");
        buttonDiv=new JButton("/");
        buttonEql =new JButton("=");
        buttonDel=new JButton("Delete");
        buttonClr=new JButton("Clear");

        button7.setBounds(40,100,50,40);
        button8.setBounds(90,100,50,40);
        button9.setBounds(140,100,50,40);
        buttonDiv.setBounds(190,100,50,40);
        button4.setBounds(40,140,50,40);
        button5.setBounds(90,140,50,40);
        button6.setBounds(140,140,50,40);
        buttonMul.setBounds(190,140,50,40);
```

```
button1.setBounds(40,180,50,40);
button2.setBounds(90,180,50,40);
button3.setBounds(140,180,50,40);
buttonSub.setBounds(190,180,50,40);
buttonDot.setBounds(40,220,50,40);
button0.setBounds(90,220,50,40);
buttonEql.setBounds(140,220,50,40);
buttonAdd.setBounds(190,220,50,40);

buttonDel.setBounds(40,260,100,40);
buttonClr.setBounds(140,260,100,40);

frame.add(textField);
frame.add(button1);
frame.add(button2);
frame.add(button3);
frame.add(button4);
frame.add(button5);
frame.add(button6);
frame.add(button7);
frame.add(button8);
frame.add(button9);
frame.add(button0);
frame.add(buttonDot);
frame.add(buttonAdd);
frame.add(buttonSub);
frame.add(buttonMul);
frame.add(buttonDiv);
frame.add(buttonEql);
frame.add(buttonDel);
frame.add(buttonClr);

button1.addActionListener(this);
button2.addActionListener(this);
button3.addActionListener(this);
button4.addActionListener(this);
button5.addActionListener(this);
button6.addActionListener(this);
button7.addActionListener(this);
button8.addActionListener(this);
button9.addActionListener(this);
button0.addActionListener(this);
buttonDot.addActionListener(this);
buttonAdd.addActionListener(this);
buttonSub.addActionListener(this);
buttonMul.addActionListener(this);
buttonDiv.addActionListener(this);
buttonEql.addActionListener(this);
buttonDel.addActionListener(this);
buttonClr.addActionListener(this);
frame.setVisible(true);
```

```
}

public static void main(String[] args) {
    new SwingCalci();
}
@Override
public void actionPerformed(ActionEvent e) {
    Object source = e.getSource();
    if (source==button1){
        textField.setText(textField.getText() + "1");
    } else if (source==button2){
        textField.setText(textField.getText() + "2");
    } else if (source==button3){
        textField.setText(textField.getText() + "3");
    } else if (source==button4){
        textField.setText(textField.getText() + "4");
    } else if (source==button5){
        textField.setText(textField.getText() + "5");
    } else if (source==button6){
        textField.setText(textField.getText() + "6");
    } else if (source==button7){
        textField.setText(textField.getText() + "7");
    } else if (source==button8){
        textField.setText(textField.getText() + "8");
    } else if (source==button9){
        textField.setText(textField.getText() + "9");
    } else if (source==button0){
        textField.setText(textField.getText() + "0");
    }
    else if (source == buttonDot) {
        if (textField.getText().contains(".")) {
            return;
        } else {
            textField.setText(textField.getText() + ".");
        }
    }
    else if (source==buttonAdd) {
        a = Double.parseDouble(textField.getText());
        textField.setText("");
        operator = 1;
    } else if (source==buttonSub) {
        a = Double.parseDouble(textField.getText());
        textField.setText("");
        operator = 2;
    } else if (source==buttonMul) {
        a = Double.parseDouble(textField.getText());
        textField.setText("");
        operator = 3;
    } else if (source==buttonDiv) {
        a = Double.parseDouble(textField.getText());
        textField.setText("");
```

```
            operator = 4;
        }

        else if (source==buttonEql) {
            b = Double.parseDouble(textField.getText());
            switch (operator){
                case 1:result =a+b;
                    break;
                case 2:result =a-b;
                    break;
                case 3:result =a*b;
                    break;
                case 4:result =a/b;
                    break;
            }
            textField.setText(""+result);
        }
        else if (source==buttonClr){
            textField.setText("");
        }
        else if (source==buttonDel){
            String s=textField.getText();
            textField.setText("");
            for (int i=0; i<s.length()-1; i++)

textField.setText(textField.getText()+s.charAt(i));
        }
    }
}
```

Output of Program No-16: -

Program No-17

Aim: Write a java JDBC program for Student Mark List Processing.

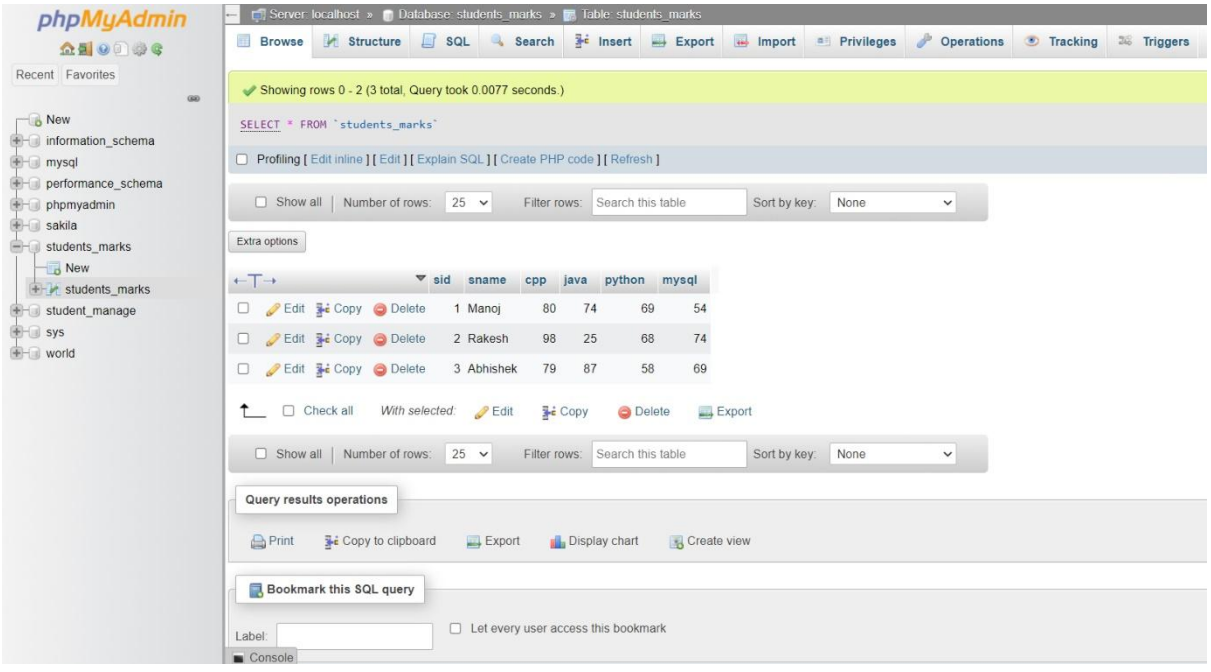
Code: -

```
/*In order to deal with JDBC standard 7 steps are supposed to
be followed:
    1. Import or create the database
    2. Load and register drivers
    3. Create the connection
    4. Create a statement/query
    5. Execute the statement/query
    6. Process the results
    7. Close the connection*/
// JDBC program for Student Mark Insert and Display
Processing.
import java.sql.*;
import java.util.Scanner;
public class studentsMarksJDBC {
    public static void main(String[] args) throws Exception {
        // Load and register driver (mysql-connector-java-
8.0.28.jar) file on this project
        Class.forName("com.mysql.cj.jdbc.Driver");
        // Create the connection
        String user = "root";
        String password = "password";
        String url =
"jdbc:mysql://localhost:3306/students_marks";
        Connection con = DriverManager.getConnection(url,
user, password);
        // Check the database connection
        if (con.isClosed()) {
            System.out.println("Database connection is
closed");
        } else {
            System.out.println("Database connection created
successfully...");
        }
        while (true) {
            Scanner intVal = new Scanner(System.in); // for
integer value
            Scanner strVal = new Scanner(System.in); // for
string value
            System.out.println("PRESS 1 to ADD student name
and marks");
            System.out.println("PRESS 2 to SHOW student name
and marks");
            System.out.println("PRESS 0 for EXIT program");
            int choice = intVal.nextInt();

            if (choice == 1) {
                // ADD student name and marks
```

```
                System.out.println("Enter the student name
:");
                String sname = strVal.nextLine();
                System.out.println("Enter the C++ obtained
marks :");
                int cpp = intVal.nextInt();
                System.out.println("Enter the JAVA obtained
marks :");
                int java = intVal.nextInt();
                System.out.println("Enter the PYTHON obtained
marks :");
                int python = intVal.nextInt();
                System.out.println("Enter the MySQL obtained
marks :");
                int mysql = intVal.nextInt();

                // Create a statement/query
                String query = "INSERT INTO
students_marks(sname, cpp, java, python, mysql)
VALUES(?,?,?,?,?)";
                // PreparedStatement
                PreparedStatement pstmt =
con.prepareStatement(query);
                // Set the value of parameter
                pstmt.setString(1, sname);
                pstmt.setInt(2, cpp);
                pstmt.setInt(3, java);
                pstmt.setInt(4, python);
                pstmt.setInt(5, mysql);

                // Execute the statement/query
                pstmt.executeUpdate();
                System.out.println("Data inserted
successfully....");

                // Close the database connection
                con.close();
                break; // Exit the loop

            } else if (choice == 2) {
                // SHOW student name and marks

                String query = "SELECT * FROM
students_marks;";
                // CreateStatement
                Statement pstmt = con.createStatement();
                // Set the value in ResultSet
                ResultSet set = pstmt.executeQuery(query);

                // Process the results
                while (set.next()) {
```

```
                    int sid = set.getInt(1);
                    String sname = set.getString(2);
                    int cpp = set.getInt(3);
                    int java = set.getInt(4);
                    int python = set.getInt(5);
                    int mysql = set.getInt(6);

                    System.out.println("Student ID :" + sid);
                    System.out.println("Student Name :" +
sname);
                    System.out.println("C++ Marks :" + cpp);
                    System.out.println("JAVA Marks :" + java);
                    System.out.println("PYTHON Marks :" +
python);
                    System.out.println("MySQL Marks :" +
mysql);
System.out.println("========================================");
                }
                // Close the database connection
                con.close();
                break; // Exit the loop
            } else if (choice == 0) {
                // EXIT program
                break;
            }
        }
    }
}
```

Output of Program No-17: -



Program No-18

Aim: Write a java program to sort an array of names (strings) in alphabetical order using the Quick Sort algorithm and display the list before and after sorting.

Code: -

```
import java.util.Arrays;
public class ProgramNo18{

    public static void main(String[] args) {
        String[] names = {"Zayan", "Amit", "Rahul", "Vikas",
"Bina", "Priya"};

        System.out.println("Before Sorting: " +
Arrays.toString(names));

        // Call quicksort on the entire array
        quickSort(names, 0, names.length - 1);

        System.out.println("After Sorting:  " +
Arrays.toString(names));
    }

    // Main QuickSort function
    public static void quickSort(String[] arr, int low, int
high) {
        if (low < high) {
            // partitionIndex is the index where the pivot is
now in the right place
            int partitionIndex = partition(arr, low, high);

            // Recursively sort elements before and after
partition
            quickSort(arr, low, partitionIndex - 1);
            quickSort(arr, partitionIndex + 1, high);
        }
    }

    public static int partition(String[] arr, int low, int
high) {
        // We pick the last name as the pivot
        String pivot = arr[high];
        int i = (low - 1); // Index of smaller element

        for (int j = low; j < high; j++) {
            // Compare current string with pivot
alphabetically
            // compareTo returns < 0 if arr[j] comes before
pivot
            if (arr[j].compareTo(pivot) < 0) {
                i++;
```

```
                // Swap arr[i] and arr[j]
                String temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }

        // Swap the pivot element with the element at i+1
        String temp = arr[i + 1];
        arr[i + 1] = arr[high];
        arr[high] = temp;

        return i + 1;
    }
}
```

Output of Program No-18: -

Before Sorting: [Zayan, Amit, Rahul, Vikas, Bina, Priya]
After Sorting: [Amit, Bina, Priya, Rahul, Vikas, Zayan]

Program No-19

Aim: Write a java program to find duplicate elements in an integer array using the simple nested loop and detect duplicate elements in a string array using the HashSet collection.

Code: -

```
import java.util.HashSet;

public class ProgramNo19 {
    public static void main(String[] args) {
        // Integer array for Nested Loop Method
        int[] numbers = {1, 2, 3, 4, 2, 7, 8, 8, 3};
        // String array for HashSet Method
        String[] languages = {"Java", "Python", "C++", "Java",
"Ruby", "C++"};

        System.out.println("=== Duplicate Detection using
Nested Loop ===");
        findDuplicateNested(numbers);
        System.out.println("\n=== Duplicate Detection using
HashSet ===");
        findDuplicateHash(languages);
    }
    // Method 1: Nested Loop (Brute Force)
    public static void findDuplicateNested(int[] arr) {
        // Compare each element with others
        for (int i = 0; i < arr.length; i++) {
            for (int j = i + 1; j < arr.length; j++) {
                if (arr[i] == arr[j]) {
                    System.out.println(arr[i]);
                    break; // avoid printing same duplicate
multiple times
    }   }   }   }
    // Method 2: HashSet (Optimized)
    public static void findDuplicateHash(String[] arr) {
        HashSet<String> set = new HashSet<>();

        for (String element : arr) {
            // .add() returns false if the element already
exists in the set
            if (!set.add(element)) {
                System.out.println(element);
    }   }   }   }
```

Output of Program No-19: -

```
=== Duplicate Detection using Nested Loop ===
2
3
8

=== Duplicate Detection using HashSet ===
Java
C++
```

Program No-20

Aim: Write a java program to remove duplicate elements from both sorted and unsorted arrays using the two-pointer technique and LinkedHashSet collection.

Code: -

```
import java.util.Arrays;
import java.util.LinkedHashSet;
public class ProgramNo20 {
    // Method 1: Remove duplicates from SORTED array (Two-
pointer technique)
    public static int removeDuplicatesSorted(int[] arr) {
        if (arr.length == 0) return 0;
        int j = 0;
        for (int i = 0; i < arr.length - 1; i++) {
            if (arr[i] != arr[i + 1]) {
                arr[j++] = arr[i];
            }
        }
        arr[j++] = arr[arr.length - 1];
        return j;
    }
    // Method 2: Remove duplicates from UNSORTED array
(LinkedHashSet)
    public static Integer[] removeDuplicatesUnsorted(Integer[]
arr) {
        LinkedHashSet<Integer> set = new
LinkedHashSet<>(Arrays.asList(arr));
        return set.toArray(new Integer[0]);
    }
    public static void main(String[] args) {
        // Sorted array
        int[] sortedArr = {10, 20, 20, 30, 30, 40, 50, 50};
        // Unsorted array
        Integer[] unsortedArr = {5, 1, 2, 6, 4, 2, 1, 3, 5};
        // ---- Sorted Method ----
        int newLength = removeDuplicatesSorted(sortedArr);
        System.out.print("Sorted Array after removal: ");
        for (int i = 0; i < newLength; i++) {
            System.out.print(sortedArr[i] + " ");        }
        // ---- Unsorted Method ----
        Integer[] uniqueArr =
removeDuplicatesUnsorted(unsortedArr);
        System.out.println("\nUnsorted Array after removal: "
+ Arrays.toString(uniqueArr));
    }
}
```

Output of Program No-20: -

Sorted Array after removal: 10 20 30 40 50

Unsorted Array after removal: [5, 1, 2, 6, 4, 3]