

ATME COLLEGE OF ENGINEERING

13th KM Stone, Bannur Road, Mysore - 560 028



A T M E

College of Engineering

DEPARTMENT OF CSE – CYBER SECURITY

(ACADEMIC YEAR 2026-27)

LABORATORY MANUAL

SUBJECT: OBJECT ORIENTED PROGRAMMING WITH JAVA LAB

SUB CODE: 1BCS302

SEMESTER: III

Composed by

Verified by

Approved by

Mr. MUKESH K

Dr. SHWETHA G K

Dr. NASREEN FATHIMA

INSTRUCTOR

FACULTY CO-ORDINATOR

HOD, CSE-CY

INSTITUTIONAL MISSION AND VISION

Objectives

- To provide quality education and groom top-notch professionals, entrepreneurs and leaders for different fields of engineering, technology and management.
- To open a Training-R & D-Design-Consultancy cell in each department, gradually introduce doctoral and postdoctoral programs, encourage basic & applied research in areas of social relevance, and develop the institute as a center of excellence.
- To develop academic, professional and financial alliances with the industry as well as the academia at national and transnational levels.
- To cultivate strong community relationships and involve the students and the staff in local community service.
- To constantly enhance the value of the educational inputs with the participation of students, faculty, parents and industry.

Vision

Development of academically excellent, culturally vibrant, socially responsible and globally competent human resources.

Mission

- To keep pace with advancements in knowledge and make the students competitive and capable at the global level.
- To create an environment for the students to acquire the right physical, intellectual, emotional and moral foundations and shine as torch bearers of tomorrow's society.
- To strive to attain ever-higher benchmarks of educational excellence.

DEPARTMENT MISSION AND VISION

Vision

"To nurture competent and ethical cyber security professionals capable of contributing to secure and resilient digital ecosystems".

Mission

- To impart strong foundational knowledge in cyber security through effective teaching– learning practices and a continuously evolving curriculum that addresses emerging cyber threats and technologies
- To promote research, innovation, and industry collaboration in emerging areas of cyber security to address real-world digital security challenges.
- To instill professional ethics, teamwork, leadership qualities, and a sense of social responsibility in addressing real-world cyber challenges
- To foster a culture of lifelong learning among students by encouraging higher studies, professional development, and active engagement with the global cyber security community.

Program Specific Outcomes:

PSO1: Apply principles of computer science, programming, algorithms, operating systems, and computer networks to design and develop efficient and reliable computing solutions

PSO2: Identify, analyze, and mitigate cyber threats and vulnerabilities using techniques related to network security, ethical hacking, penetration testing, and digital forensics.

PSO3: Apply cyber security policies, cyber laws, risk management practices, and incident response strategies to ensure the protection and resilience of digital systems

OBJECT ORIENTED PROGRAMMING WITH JAVA			
Subject Code	1BCS302	CIE Marks	50
Number of Contact Hours/Week	2:0:2	SEE Marks	50
Total Number of Lab Contact Hours	28	Exam Hours	3 Hrs.
Credits – 1			
Experiments			
1.	Develop a Java program to add TWO matrices of suitable order N.		
2.	Develop a stack class to hold a maximum of 10 integers with suitable methods. Develop a JAVA main method to illustrate Stack operations		
3.	Create a Java class called Student with the USN, Name, Branch, Phone details as variables within it. Write a Java program to create N Student objects and print the USN, Name, Branch, and Phone of these objects with suitable headings.		
4.	Develop a Java program to create a class named Shape. Create three sub classes, namely: circle, triangle and square. Each class has two member functions named draw and erase. Demonstrate the concepts of polymorphism by developing appropriate methods, defining member data, and implementing the main program.		
5.	Develop a Java program to create an interface Resizable with methods resizeWidth(int width) and resizeHeight(int height) that allow an object to be resized. Create a class Rectangle that implements the Resizable interface and implements the resize methods.		
6.	Develop a Java program to create a package named mypack and import & implement it in a suitable class.		
7.	Develop a Java program to raise a custom exception (user-defined exception) for DivisionByZero using try, catch, throw and finally.		
8.	Develop a program to create a class MyThread with a class constructor, call the base class constructor, using super and start the thread. The run method of the class starts after this. It can be observed that both main thread and created child thread are executed concurrently.		
9.	Develop a Java program using an enumeration called Department containing CSE, ISE, ECE, and MECH. Accept a department name as input and display the selected department using valueOf().		
10	Develop a Java program to read the marks of three subjects as strings, convert them into wrapper objects, calculate the total and average using autoboxing/unboxing and display the result.		
11	Micro Project: A team of two to three students shall develop a project and submit a brief report for a specific task assigned by the course instructor based on the concepts covered in the syllabus.		

Course Outcomes (Course Skill Set):

At the end of the course the student will be able to:

- CO1. Demonstrate proficiency writing simple programs involving branching and looping structures.
- CO2. Construct object-oriented solutions using classes, objects, and methods
- CO3. Develop programs using inheritance, packages, and interfaces.
- CO4. Make use of exception handling and multithreading in solving complex problems
- CO5. Apply concepts of autoboxing, enumerations, and generics in program development.
- CO6. Choose a suitable IDE tool to design, develop, and debug Java programs.

Assessment Details (both CIE and SEE)

The assessment for each course is equally divided between Continuous Internal Evaluation (CIE) and the Semester End Examination (SEE), with each component carrying 50% weightage (i.e., 50 marks each).

The CIE Theory component will be 25 marks and CIE Practical component will be 25 marks.

The CIE Theory component consists of IA tests for 25 marks. The CIE Practical component for continuous assessments will be for 15 marks through rubrics and for lab tests will be for 10 marks.

- To qualify and become eligible to appear for SEE, in the CIE theory component, a student must score at least 40% of 25 marks, i.e., 10 marks.
- To qualify and become eligible to appear for SEE, in the CIE Practical component, a student must secure a minimum of 40% of 25 marks, i.e., 10 marks.
- To pass the SEE, a student must secure a minimum of 35% of 50 marks, i.e., 18 marks.
- A student is deemed to have successfully completed the course if the combined total of CIE and SEE is at least 40 out of 100 marks.

CIE Practical component:

The CIE marks awarded in the case of the Practical component shall be based on the continuous evaluation of the laboratory report using a defined set of rubrics. Each experiment report can be evaluated for 30 marks. The summation of all the experiments marks to be scaled down to 15 marks. The laboratory test (duration 03 hours) at the end of the last week of the semester /after completion of all the experiments (whichever is early) shall be conducted for 50 marks and scaled down to 10 marks. For the laboratory test, the student is required to conduct one experiment (excluding the micro project).

Suggested Learning Resources:

Textbook

Herbert Schildt and Danny Coward, Java: The Complete Reference, 13th Edition, McGraw-Hill, 2024.

Reference books / Manuals:

1. Cay S. Horstmann, Core Java Volume I – Fundamentals, 14th Edition, Oracle Press, 2026.
2. E. Balagurusamy, Programming with Java, 7th Edition, McGraw-Hill Education, 2024.

CONTENTS

Sl. No	Experiment	CO 's	Page No
1	Develop a Java program to add TWO matrices of suitable order N.		1-2
2	Develop a stack class to hold a maximum of 10 integers with suitable methods. Develop a Java main method to illustrate Stack operations		3-6
3	Create a Java class called Student with the USN,Name,Branch,Phone details as variables within it. Write a Java program to create N Student objects and print the USN, Name, Branch, and Phone of these objects with suitable headings.		7-8
4	Develop a Java program to create a class named Shape. Create three sub classes, namely: circle, triangle and square. Each class has two member functions named draw and erase. Demonstrate the concepts of polymorphism by developing appropriate methods, defining member data, and implementing the main program.		9-11
5	Develop a Java program to create an interface Resizable with methods resizeWidth(int width) and resizeHeight(int height) that allow an object to be resized. Create a class Rectangle that implements the Resizable interface and implements the resize methods.		12-13
6	Develop a Java program to create a package named mypack and import & implement it in a suitable class.		14-14
7	Develop a Java program to raise a custom exception (user-defined exception) for DivisionByZero using try, catch, throw and finally.		15-15
8	Develop a program to create a class MyThread with a class constructor, call the base class constructor, using super and start the thread. The run method of the class starts after this. It can be observed that both main thread and created child thread are executed concurrently.		16-17
9	Develop a Java program using an enumeration called Department containing CSE, ISE, ECE, and MECH. Accept a department name as input and display the selected department using valueOf()..		18-18
10	Develop a Java program to read the marks of three subjects as strings, convert them into wrapper objects, calculate the total and average using autoboxing/unboxing and display the result.		19-20

1. Develop a JAVA program to add TWO matrices of suitable order N.

```
import java.util.Scanner;
public class MatrixAdd
{
    public static void main(String[] args)
    {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter matrix size N: ");
        int N = scanner.nextInt();

        int[][] A = new int[N][N];
        int[][] B = new int[N][N];
        int[][] C = new int[N][N];

        System.out.println("Enter elements of Matrix A:");
        for (int i = 0; i < N; i++)
        {
            for (int j = 0; j < N; j++)
            {
                A[i][j] = scanner.nextInt();
            }
        }

        System.out.println("Enter elements of Matrix B:");
        for (int i = 0; i < N; i++)
        {
            for (int j = 0; j < N; j++)
            {
                B[i][j] = scanner.nextInt();
            }
        }
        System.out.println("Matrix C (A + B):");
        for (int i = 0; i < N; i++)
        {
            for (int j = 0; j < N; j++)
            {
                C[i][j] = A[i][j] + B[i][j];
                System.out.print(C[i][j] + "\t");
            }
            System.out.println();
        }
        scanner.close();
    }
}
```

Output:

Enter matrix size N: 2

Enter elements of Matrix A:

1

2

3

4

Enter elements of Matrix B:

5

6

7

8

Matrix C (A + B):

6 8

10 12

2. Develop a stack class to hold a maximum of 10 integers with suitable methods. Develop a JAVA main method to illustrate Stack operations.

```
import java.util.Scanner;

class Stack
{
    private int[] stack = new int[5];
    private int top = -1;

    void push(int value)
    {
        if (top < 4)
        {
            stack[++top] = value;
            System.out.println(value + " pushed.");
        }
        else
        {
            System.out.println("Stack Overflow!");
        }
    }

    void pop()
    {
        if (top >= 0)
        {
            System.out.println(stack[top--] + " popped.");
        } else
        {
            System.out.println("Stack Underflow!");
        }
    }

    void display()
    {
        if (top == -1)
        {
            System.out.println("Stack is empty.");
        }
        else
        {
            System.out.println("Stack elements:");
            for (int i = 0; i <= top; i++)
            {
                System.out.print(stack[i] + " ");
            }
        }
    }
}
```

```
        System.out.println();
    }
}

public class StackDemo
{
    public static void main(String[] args)
    {
        Stack obj = new Stack();
        Scanner sc = new Scanner(System.in);
        int choice, value;

        do {
            System.out.println("\n Choose an operation:");
            System.out.println("1. Push");
            System.out.println("2. Pop");
            System.out.println("3. Display");
            System.out.println("4. Exit");
            System.out.print("Enter your choice: ");
            choice = sc.nextInt();

            switch (choice)
            {
                case 1:
                    System.out.print("Enter value to push: ");
                    value = sc.nextInt();
                    obj.push(value);
                    break;
                case 2:
                    obj.pop();
                    break;
                case 3:
                    obj.display();
                    break;
                case 4:
                    System.out.println("Exiting...");
                    break;
                default:
                    System.out.println("Invalid choice! Please choose again.");
            }
        } while (choice != 4);
    }
}
```

Output:

Choose an operation:

1. Push
2. Pop
3. Display
4. Exit

Enter your choice: 1

Enter value to push: 10

10 pushed.

Choose an operation:

1. Push
2. Pop
3. Display
4. Exit

Enter your choice: 1

Enter value to push: 20

20 pushed.

Choose an operation:

1. Push
2. Pop
3. Display
4. Exit

Enter your choice: 3

Stack elements:

10 20

Choose an operation:

1. Push
2. Pop
3. Display
4. Exit

Enter your choice: 2

20 popped.

Choose an operation:

1. Push
2. Pop
3. Display
4. Exit

Enter your choice: 3

Stack elements:

10

Choose an operation:

1. Push
2. Pop

3. Display

4. Exit

Enter your choice: 2

Stack Underflow!

Choose an operation:

1. Push

2. Pop

3. Display

4. Exit

Enter your choice: 4

Exiting...

3. Create a Java class called Student with the USN,Name,Branch,Phone details as variables within it. Write a Java program to create N Student objects and print the USN, Name, Branch, and Phone of these objects with suitable headings.

```
import java.util.Scanner;

class Student {
    String usn;
    String name;
    String branch;
    String phone;

    // Constructor
    Student(String usn, String name, String branch, String phone) {
        this.usn = usn;
        this.name = name;
        this.branch = branch;
        this.phone = phone;
    }

    // Method to display student details
    void display() {
        System.out.printf("%-15s %-20s %-15s %-15s\n", usn, name, branch, phone);
    }
}

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

        System.out.print("Enter the number of students: ");
        int n = sc.nextInt();
        sc.nextLine(); // Consume newline

        Student[] students = new Student[n];

        // Input student details
        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details of Student " + (i + 1));

            System.out.print("USN: ");
            String usn = sc.nextLine();

            System.out.print("Name: ");
            String name = sc.nextLine();

            System.out.print("Branch: ");
            String branch = sc.nextLine();
        }
    }
}
```

```
        System.out.print("Phone: ");
        String phone = sc.nextLine();
        students[i] = new Student(usn, name, branch, phone);
    }

    // Display student details
    System.out.println("\nStudent Details");
    System.out.printf("%-15s %-20s %-15s %-15s\n", "USN", "Name", "Branch", "Phone");
    System.out.println("-----");

    for (Student s : students) {
        s.display();
    }

    sc.close();
}
```

Output:

Enter the number of students: 2

Enter details of Student 1

USN: 1RV23CS001

Name: Rahul

Branch: CSE

Phone: 9876543210

Enter details of Student 2

USN: 1RV23EC002

Name: Priya

Branch: ECE

Phone: 9123456780

Student Details

USN	Name	Branch	Phone
4AD23CG001	Rahul	CSE	9876543210
4AD23CG002	Priya	ECE	9123456780

4. Develop a JAVA program to create a class named shape. Create three sub classes namely: circle, triangle and square, each class has two member functions named draw () and erase (). Demonstrate polymorphism concepts by developing suitable methods, defining member data and main program.

```
// Base class Shape
class Shape
{
    // Methods to be overridden by subclasses
    void draw()
    {
        System.out.println("Drawing a shape");
    }

    void erase()
    {
        System.out.println("Erasing a shape");
    }
}

// Subclass Circle
class Circle extends Shape
{
    // Overriding the draw method for Circle
    @Override
    void draw()
    {
        System.out.println("Drawing a circle");
    }

    // Overriding the erase method for Circle
    @Override
    void erase()
    {
        System.out.println("Erasing a circle");
    }
}

class Triangle extends Shape
{
    // Overriding the draw method for Triangle
    @Override
    void draw()
    {
        System.out.println("Drawing a triangle");
    }

    // Overriding the erase method for Triangle
    @Override
```

```
void erase()
{
    System.out.println("Erasing a triangle");
}

// Subclass Square
class Square extends Shape
{
    // Overriding the draw method for Square
    @Override
    void draw()
    {
        System.out.println("Drawing a square");
    }

    // Overriding the erase method for Square
    @Override
    void erase()
    {
        System.out.println("Erasing a square");
    }
}

public class Shapes
{
    public static void main(String[] args)
    {
        // Creating references of Shape and pointing to different objects
        Shape shape;

        // Polymorphism: shape refers to Circle object
        shape = new Circle();
        shape.draw();
        shape.erase();

        // Polymorphism: shape refers to Triangle object
        shape = new Triangle();
        shape.draw();
        shape.erase();

        // Polymorphism: shape refers to Square object
        shape = new Square();
        shape.draw();
        shape.erase();
    }
}
```

Output:

Drawing a circle

Erasing a circle

Drawing a triangle

Erasing a triangle

Drawing a square

Erasing a square

5. Develop a JAVA program to create an interface Resizable with methods `resizeWidth(int width)` and `resizeHeight(int height)` that allow an object to be resized. Create a class `Rectangle` that implements the `Resizable` interface and implements the `resize` methods.

```
import java.util.Scanner;

// Interface
interface Resizable {
    void resizeWidth(int width);
    void resizeHeight(int height);
}

// Rectangle class implementing Resizable
class Rectangle implements Resizable {
    private int width;
    private int height;

    // Constructor
    public Rectangle(int width, int height) {
        this.width = width;
        this.height = height;
    }

    // Implement methods from Resizable
    @Override
    public void resizeWidth(int width) {
        this.width = width;
    }

    @Override
    public void resizeHeight(int height) {
        this.height = height;
    }

    // Method to display rectangle details
    public void display() {
        System.out.println("Rectangle width: " + width);
        System.out.println("Rectangle height: " + height);
        System.out.println("Area: " + (width * height));
    }
}

// Main class
public class Resize {
    public static void main(String[] args) {
```

```
Scanner sc = new Scanner(System.in);
// Create Rectangle
System.out.print("Enter initial width of rectangle: ");
int width = sc.nextInt();
System.out.print("Enter initial height of rectangle: ");
int height = sc.nextInt();

Rectangle rect = new Rectangle(width, height);
System.out.println("\nInitial Rectangle:");
rect.display();

// Resize Rectangle
System.out.print("\nEnter new width: ");
int newWidth = sc.nextInt();
rect.resizeWidth(newWidth);

System.out.print("Enter new height: ");
int newHeight = sc.nextInt();
rect.resizeHeight(newHeight);

System.out.println("\nResized Rectangle:");
rect.display();

sc.close();
}
}
```

Output:

Enter initial width of rectangle: 6
Enter initial height of rectangle: 9

Initial Rectangle:
Rectangle width: 6
Rectangle height: 9
Area: 54

Enter new width: 5
Enter new height: 8

Resized Rectangle:
Rectangle width: 5
Rectangle height: 8
Area: 40

6. Develop a JAVA program to create a package named mypack and import & implement it in a suitable class.

```
package mypack;

public class MyPackageClass
{
    public void display()
    {
        System.out.println("Hello from MyPackageClass in mypack package!");
    }
    public static int addN(int a, int b)
    {
        return a + b;
    }
}

import mypack.MyPackageClass;

public class DemoPackage
{
    public static void main(String[] args)
    {
        // Creating an instance of MyPackageClass from the mypack package
        MyPackageClass myPackageObject = new MyPackageClass();

        // Calling the display method from MyPackageClass
        myPackageObject.display();

        // Using the utility method addN from MyPackageClass
        int result = MyPackageClass.addN(5, 3);
        System.out.println("Result of adding numbers: " + result);
    }
}
```

Output:

Hello from MyPackageClass in mypack package!
Result of adding numbers: 8

7. Develop a JAVA program to raise a custom exception (user defined exception) for Division ByZero using try, catch, throw and finally.

```
import java.util.Scanner;

// Custom Exception without message
class DivisionByZeroException extends Exception { }

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

        try {
            System.out.print("Enter numerator: ");
            int num = sc.nextInt();

            System.out.print("Enter denominator: ");
            int den = sc.nextInt();

            if (den == 0) {
                throw new DivisionByZeroException(); // no message
            }

            int result = num / den;
            System.out.println("Result = " + result);
        }
        catch (DivisionByZeroException e) {
            System.out.println("Exception: Division by zero not allowed!");
        }
        finally {
            System.out.println("Program finished.");
            sc.close();
        }
    }
}
```

Output:

```
Enter numerator: 50
Enter denominator: 0
Exception: Division by zero not allowed!
Program finished.
```

8. Develop a program to create a class `MyThread` in this class a constructor, call the base class constructor, using `super` and start the thread. The `run` method of the class starts after this. It can be observed that both main thread and created child thread is executed concurrently.

```
//super class
class MyThread extends Thread
{
    // Constructor to call the base class (Thread) constructor public
    public MyThread(String threadName)
    {
        super(threadName); // Calling Constructor of thread class
    }
    // Override the run() method
    public void run()
    {
        for (int i = 1; i <= 5; i++)
        {
            System.out.println(getName() + " - Count: " + i);
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                System.out.println(getName() + "interrupted.");
            }
        }
        System.out.println(getName() + " has finished execution.");
    }
}

public class Prog12
{
    public static void main(String[] args)
    {
        // creating thread objects
        MyThread t1 = new MyThread("Thread 1");
        MyThread t2 = new MyThread("Thread 2");
        //starting threads
        t1.start();
        t2.start();
    }
}
```

Output:

Thread 2 - Count: 1
Thread 1 - Count: 1
Thread 1 - Count: 2
Thread 2 - Count: 2
Thread 1 - Count: 3
Thread 2 - Count: 3
Thread 1 - Count: 4
Thread 2 - Count: 4
Thread 1 - Count: 5
Thread 2 - Count: 5
Thread 1 has finished execution.
Thread 2 has finished execution.

9. Develop a Java program using an enumeration called Department containing CSE, ISE, ECE, and MECH. Accept a department name as input and display the selected department using valueOf().

```
import java.util.Scanner;

public class DepartmentSelector {
    // Define the Department enumeration inside the main class
    enum Department {
        CSE,
        ISE,
        ECE,
        MECH
    }

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

        System.out.print("Enter department name (CSE, ISE, ECE, MECH): ");
        String input = scanner.next().toUpperCase();

        try {
            Department selectedDept = Department.valueOf(input);
            System.out.println("Selected Department: " + selectedDept);
        } catch (IllegalArgumentException e) {
            System.out.println("Error: Invalid department name entered.");
        } finally {
            scanner.close();
        }
    }
}
```

Output:

```
Enter department name (CSE, ISE, ECE, MECH): CSE
Selected Department: CSE
```

10. Develop a Java program to read the marks of three subjects as strings, convert them into wrapper objects, calculate the total and average using autoboxing/unboxing and display the result.

```
import java.util.Scanner;

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

        try {
            System.out.print("Enter marks for Subject 1: ");
            String markStr1 = scanner.next();

            System.out.print("Enter marks for Subject 2: ");
            String markStr2 = scanner.next();

            System.out.print("Enter marks for Subject 3: ");
            String markStr3 = scanner.next();

            Double mark1 = Double.valueOf(markStr1);
            Double mark2 = Double.valueOf(markStr2);
            Double mark3 = Double.valueOf(markStr3);

            Double total = mark1 + mark2 + mark3;

            Double average = total / 3.0;

            System.out.println("\n--- Results ---");
            System.out.println("Subject 1: " + mark1);
            System.out.println("Subject 2: " + mark2);
            System.out.println("Subject 3: " + mark3);
            System.out.println("Total Marks: " + total);
            System.out.println("Average Marks: " + average);

        } catch (NumberFormatException e) {
            System.out.println("\nError: Please enter valid numerical values for marks.");
        } finally {
            scanner.close();
        }
    }
}
```

Output:

Enter marks for Subject 1: 85.5
Enter marks for Subject 2: 90.0
Enter marks for Subject 3: 78.5

--- Results ---

Subject 1: 85.5
Subject 2: 90.0
Subject 3: 78.5
Total Marks: 254.0
Average Marks: 84.67