Experiment 5

Student Name: Amika **UID: 22BCS14180**
Branch: B.E CSE **Section: IOT-643-A**
Semester: 6th **DOP:24/02/25**
Subject: PBLJ **Subject Code: 22CSH-359**

Aim: Develop Java programs using autoboxing, serialization, file handling, and efficient data processing and management.

Problem Statement :

Easy Level: Write a Java program to calculate the sum of a list of integers using autoboxing and unboxing. Include methods to parse strings into their respective wrapper classes (e.g., Integer.parseInt()).
Medium Level: Create a Java program to serialize and deserialize a Student object. The program should: Serialize a Student object (containing id, name, and GPA) and save it to a file. Deserialize the object from the file and display the student details. Handle FileNotFoundException, IOException, and ClassNotFoundException using exception handling.
Hard Level: Create a menu-based Java application with the following options. 1.Add an Employee 2. Display All 3. Exit If option 1 is selected, the application should gather details of the employee like employee name, employee id, designation and salary and store it in a file. If option 2 is selected, the application should display all the employee details. If option 3 is selected the application should exit.

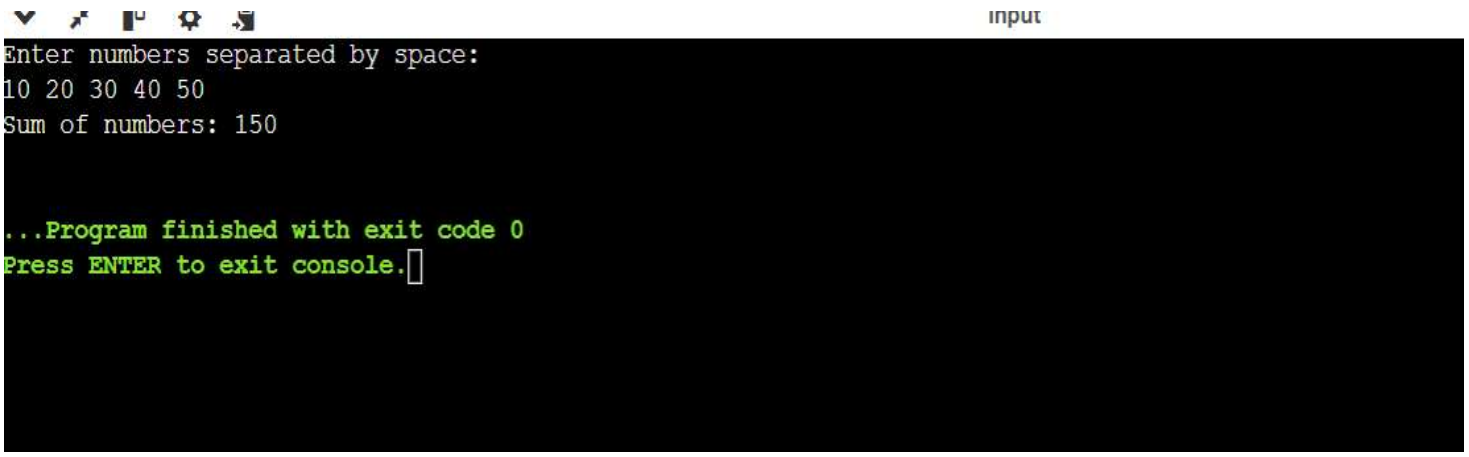
Program :

1. **IntegerSumCalculator:**

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class IntegerSumCalculator {
    public static void main(String[] args) {
        List<Integer> numbers = new ArrayList<>();
        Scanner scanner = new Scanner(System.in);
```

```
        System.out.println("Enter numbers separated by space:");
        String input = scanner.nextLine();
        String[] inputs = input.split(" ");

        for (String str : inputs) {
            numbers.add(parseInteger(str));
        }

        int sum = calculateSum(numbers);
        System.out.println("Sum of numbers: " + sum);
        scanner.close();
    }

    public static Integer parseInteger(String str) {
        return Integer.parseInt(str);
    }

    public static int calculateSum(List<Integer> numbers) {
        int sum = 0;
        for (Integer num : numbers) {
            sum += num;
        }
        return sum;
    }
}
```

Output:



2. implements Serializable:

```
import java.io.*;

class Student implements Serializable {
    int id;
    String name;
    double gpa;

    Student(int id, String name, double gpa) {
        this.id = id;
        this.name = name;
        this.gpa = gpa;
    }

    public String toString() {
        return "ID: " + id + ", Name: " + name + ", GPA: " + gpa;
    }
}
```

```
public class StudentSerialization {
    public static void main(String[] args) {
        Student student = new Student(101, "John Doe", 3.8);
        String filename = "student.ser";

        // Serialization
        try (ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream(filename)))
{
            out.writeObject(student);
            System.out.println("Student serialized successfully!");
        } catch (IOException e) {
            System.out.println("Error during serialization: " + e.getMessage());
        }

        // Deserialization
        try (ObjectInputStream in = new ObjectInputStream(new FileInputStream(filename))) {
            Student deserializedStudent = (Student) in.readObject();
            System.out.println("Deserialized Student: " + deserializedStudent);
        } catch (FileNotFoundException e) {
            System.out.println("File not found!");
        } catch (IOException e) {
            System.out.println("Error during deserialization: " + e.getMessage());
        } catch (ClassNotFoundException e) {
            System.out.println("Class not found!");
        }
    }
}
```

Output:

```
Student serialized successfully!
Deserialized Student: ID: 101, Name: John Doe, GPA: 3.8


...Program finished with exit code 0
Press ENTER to exit console.
```

3. Ticket Booking System:

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

class Employee implements Serializable {
    int id;
    String name;
    String designation;
    double salary;
```

```
    Employee(int id, String name, String designation, double salary) {
        this.id = id;
        this.name = name;
        this.designation = designation;
        this.salary = salary;
    }

    public String toString() {
        return "ID: " + id + ", Name: " + name + ", Designation: " + designation + ", Salary: " +
salary;
    }
}

public class EmployeeManagement {
    static final String FILE_NAME = "employees.dat";

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

        while (true) {
            System.out.println("\nEmployee Management System");
            System.out.println("1. Add Employee");
            System.out.println("2. Display All Employees");
            System.out.println("3. Exit");
            System.out.print("Enter your choice: ");

            int choice = scanner.nextInt();
            scanner.nextLine();

            if (choice == 1) {
                addEmployee(scanner);
            } else if (choice == 2) {
                displayEmployees();
            } else if (choice == 3) {
                System.out.println("Exiting program...");
                break;
            } else {
                System.out.println("Invalid choice. Try again.");
            }
        }
        scanner.close();
    }

    public static void addEmployee(Scanner scanner) {
        System.out.print("Enter Employee ID: ");
        int id = scanner.nextInt();
```

```
        scanner.nextLine();

        System.out.print("Enter Employee Name: ");
        String name = scanner.nextLine();

        System.out.print("Enter Designation: ");
        String designation = scanner.nextLine();

        System.out.print("Enter Salary: ");
        double salary = scanner.nextDouble();

        Employee emp = new Employee(id, name, designation, salary);

        try (ObjectOutputStream out = new ObjectOutputStream(new
FileOutputStream(FILE_NAME, true))) {
            out.writeObject(emp);
            System.out.println("Employee added successfully!");
        } catch (IOException e) {
            System.out.println("Error saving employee: " + e.getMessage());
        }
    }

    public static void displayEmployees() {
        try (ObjectInputStream in = new ObjectInputStream(new FileInputStream(FILE_NAME))) {
            System.out.println("\nEmployee List:");
            while (true) {
                Employee emp = (Employee) in.readObject();
                System.out.println(emp);
            }
        } catch (EOFException e) {
            System.out.println("End of employee list.");
        } catch (FileNotFoundException e) {
            System.out.println("No employee records found.");
        } catch (IOException | ClassNotFoundException e) {
            System.out.println("Error reading employee data: " + e.getMessage());
        }
    }
}
```

Output:

```
1. Add Employee
2. Display All Employees
3. Exit
Enter your choice: 1
Enter Employee ID: 160
Enter Employee Name: Shreya
Enter Designation: Developer
Enter Salary: 70000
Employee added successfully!

Employee Management System
1. Add Employee
2. Display All Employees
3. Exit
Enter your choice: 2

Employee List:
ID: 180, Name: Riya, Designation: Manager, Salary: 50000.0
Error reading employee data: invalid type code: AC

Employee Management System
1. Add Employee
2. Display All Employees
3. Exit
Enter your choice: 3
Exiting program...
```