Experiment 1.5

Student Name: Kashni **UID: 22BCS13644**

Branch: BE CSE **Section/Group: IOT-616 A**

Semester: 6th **Date of Performance:17.02.25**

Subject Name: JAVA **Subject Code: 22CSH-359**

Aim: Use of wrapper classes in Java-Integer, Character, Long, Boolean. Autoboxing and Unboxing. Byte stream, Character stream, Object serialization, cloning. System defined annotations, Custom annotations, application of annotations, Testing using JUnit.

Objective:

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:

Add an Employee

Display All

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.

Implementation/Code:

EASY:-

```
import java.util.*;

public class SumUsingWrapper {
   public static void main(String[] args) {
```

```
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter numbers separated by space:");
        String input = scanner.nextLine();

        String[] numbers = input.split(" ");
        int sum = 0;

        for (String num : numbers) {
            sum += Integer.parseInt(num); // Parsing String to Integer
        }

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

MEDIUM:-

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

class Student implements Serializable {
    private static final long serialVersionUID = 1L;
    int id;
    String name;
    double gpa;

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

    public void display() {
        System.out.println("\nStudent Details:");
        System.out.println("ID: " + id);
        System.out.println("Name: " + name);
        System.out.println("GPA: " + gpa);
    }
```

```
}

public class StudentSerialization {
    private static final String FILE_NAME = "student.ser";

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

        // Get student details
        System.out.print("Enter Student ID: ");
        int id = scanner.nextInt();
        scanner.nextLine(); // Consume newline
        System.out.print("Enter Student Name: ");
        String name = scanner.nextLine();
        System.out.print("Enter GPA: ");
        double gpa = scanner.nextDouble();

        Student student = new Student(id, name, gpa);

        // Serialize the student object
        serializeStudent(student);

        // Deserialize and display student details
        deserializeStudent();

        scanner.close();
    }

    private static void serializeStudent(Student student) {
        try (ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream(FILE_NAME))) {
            out.writeObject(student);
            System.out.println("Student object has been serialized to " + FILE_NAME);
        } catch (IOException e) {
            System.out.println("Error during serialization: " + e.getMessage());
        }
    }

    private static void deserializeStudent() {
```

```
        try (ObjectInputStream in = new ObjectInputStream(new FileInputStream(FILE_NAME))) {
            Student deserializedStudent = (Student) in.readObject();
            System.out.println("\nDeserialized Student:");
            deserializedStudent.display();
        } catch (FileNotFoundException e) {
            System.out.println("Error: File not found. Please serialize a student first.");
        } catch (IOException e) {
            System.out.println("Error during deserialization: " + e.getMessage());
        } catch (ClassNotFoundException e) {
            System.out.println("Error: Student class not found.");
        }
    }
}
```

HARD:-

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

class Employee implements Serializable {
    private static final long serialVersionUID = 1L;
    int id;
    String name;
    String designation;
    double salary;

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

    public void display() {
        System.out.println("\nEmployee Details:");
        System.out.println("ID: " + id);
        System.out.println("Name: " + name);
        System.out.println("Designation: " + designation);
        System.out.println("Salary: " + salary);
```

```
    }
}

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

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        List<Employee> employees = loadEmployees();

        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("Choose an option: ");
            int choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    addEmployee(scanner, employees);
                    break;
                case 2:
                    displayEmployees(employees);
                    break;
                case 3:
                    saveEmployees(employees);
                    System.out.println("Exiting...");
                    scanner.close();
                    return;
                default:
                    System.out.println("Invalid choice! Try again.");
            }
        }
    }

    private static void addEmployee(Scanner scanner, List<Employee> employees) {
        System.out.print("Enter Employee ID: ");
```

```
    int id = scanner.nextInt();
    scanner.nextLine(); // Consume newline
    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();

    employees.add(new Employee(id, name, designation, salary));
    System.out.println("Employee added successfully.");
}

private static void displayEmployees(List<Employee> employees) {
    if (employees.isEmpty()) {
        System.out.println("No employees found.");
        return;
    }
    System.out.println("\nAll Employees:");
    for (Employee emp : employees) {
        emp.display();
        System.out.println("--------------------");
    }
}

private static void saveEmployees(List<Employee> employees) {
    try (ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream(FILE_NAME))) {
        out.writeObject(employees);
        System.out.println("Employees saved successfully.");
    } catch (IOException e) {
        System.out.println("Error saving employees: " + e.getMessage());
    }
}

private static List<Employee> loadEmployees() {
    File file = new File(FILE_NAME);
    if (!file.exists()) return new ArrayList<>();

    try (ObjectInputStream in = new ObjectInputStream(new FileInputStream(FILE_NAME))) {
```

```
            return (List<Employee>) in.readObject();
        } catch (IOException | ClassNotFoundException e) {
            System.out.println("Error loading employees: " + e.getMessage());
            return new ArrayList<>();
        }
    }
}
```

4. OUTPUT

```
Enter numbers separated by space:
5 6
Sum of numbers: 11


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

```
Enter Student ID: 1234
Enter Student Name: Sam
Enter GPA: 9.5
Student object has been serialized to student.ser

Deserialized Student:

Student Details:
ID: 1234
Name: Sam
GPA: 9.5


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

```
Employee Management System:
1. Add Employee
2. Display All Employees
3. Exit
Choose an option: 1
Enter Employee ID: 2345
Enter Employee Name: Sam
Enter Designation: Teacher
Enter Salary: 100000
Employee added successfully.

Employee Management System:
1. Add Employee
2. Display All Employees
3. Exit
Choose an option: 2

All Employees:

Employee Details:
ID: 2345
Name: Sam
Designation: Teacher
Salary: 100000.0
--------------------
```

5. Learning Outcome

- We learned how to perform basic operations such as insert, delete, search, and display on a list of strings.
- We learned to implement a menu-driven program to handle user input and execute specific operations based on user choice.
- We learned how to manage and manipulate data in a list dynamically, adding and removing items as needed.
- We learned how to handle errors and edge cases, ensuring the program responds appropriately when an item does not exist.