Experiment 5

Student Name: Shraddha Sharma **UID:** 22BCS15236
Branch: BE CSE **Section/Group:** EPAM-801(B)
Semester: 06 **Date of Performance:** 23-2-25
Subject Name: Project Based Learning in Java **Subject Code:** 22CSH-359

1. Aim-

Easy: 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: 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: 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.

2. Procedure-

Easy Level: Sum of Integers

1. Initialize an empty list.
2. Take user inputs until "end" is entered.
3. Convert each input to an integer (autoboxing) and add to the list.
4. Calculate the sum by unboxing each Integer.
5. Display the sum.

Medium Level: Serialization and Deserialization 1.
Create a Student class implementing Serializable.

2. Serialize:
   - Create a Student object.
   - Save it to a file using ObjectOutputStream.
3. Deserialize:
   - Read the object from the file using ObjectInputStream. o Display the object data.

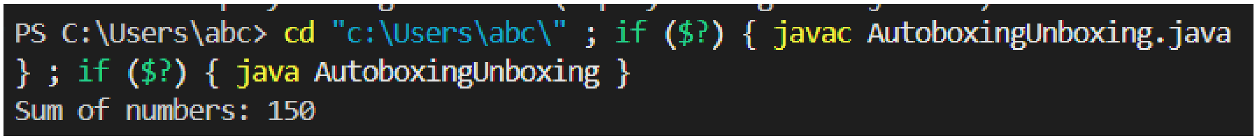
Hard Level: Employee Management

1. Display a menu:
   - Add Employee o Display All Employees o Exit
2. For Add Employee:
   - Take input for ID, Name, Designation, and Salary. o Save it as an Employee object in a list.
   - Serialize the list to a file.
3. For Display All Employees:
   - Deserialize the list from the file.
   - Display each employee's details.
4. Exit the program on user choice.

3. Code- EASY:

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

public class AutoboxingUnboxing {

   public static List<Integer> convertToIntegerList(String[] numbers) {
      List<Integer> intList = new ArrayList<>();
      for (String num : numbers) {
         intList.add(Integer.parseInt(num));
      }
      return intList;
   }

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

   public static void main(String[] args) {
      String[] strNumbers = { "40", "10", "20", "20", "60" };

      List<Integer> integerList = convertToIntegerList(strNumbers);

      int sum = calculateSum(integerList);
```

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



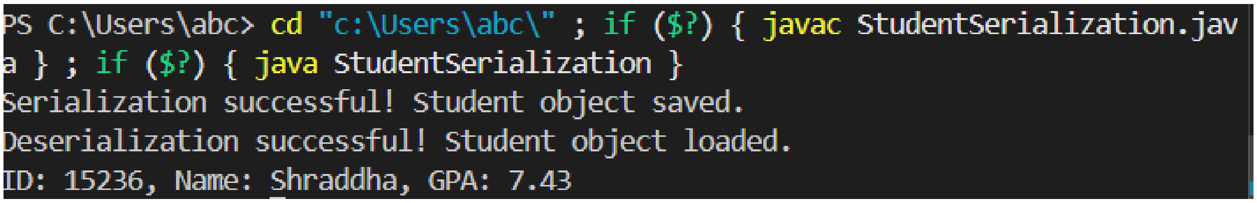
```
PS C:\Users\abc> cd "c:\Users\abc\" ; if ($?) { javac AutoboxingUnboxing.java
} ; if ($?) { java AutoboxingUnboxing }
Sum of numbers: 150
```

```
}
```

MEDIUM:

```
 import java.io.*;


class Student implements Serializable {

    private static final long serialVersionUID = 1L;

    private int id;

    private String name;

    private 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("ID: " + id + ", Name: " + name + ", GPA: " + gpa);
    }
}


class StudentSerialization {
    public static void serializeStudent(Student student, String filename) {
        try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(filename))) {
            oos.writeObject(student);
            System.out.println("Serialization successful! Student object saved.");
        } catch (IOException e) {
            System.out.println("Error during serialization: " + e.getMessage());
        }
    }


    public static Student deserializeStudent(String filename) {
        Student student = null;
        try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(filename))) {
            student = (Student) ois.readObject();
            System.out.println("Deserialization successful! Student object loaded.");
        } catch (IOException | ClassNotFoundException e) {
            System.out.println("Error during deserialization: " + e.getMessage());
        }
```

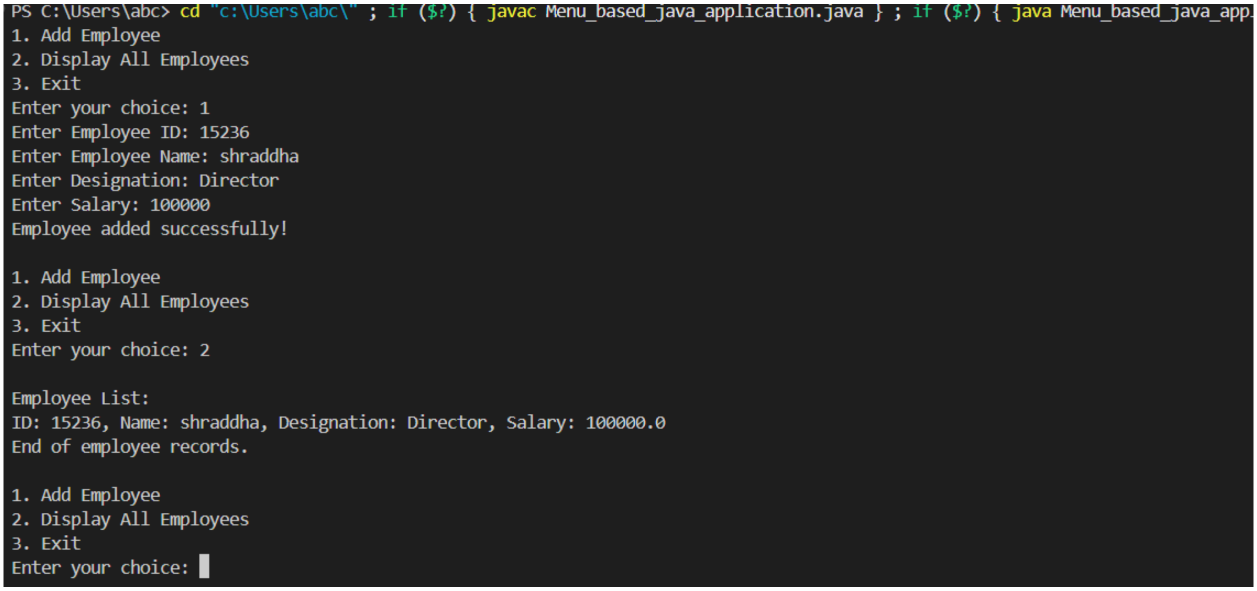
```
            return student;
        }

        public static void main(String[] args) {
            String filename = "student.ser";
            Student student1 = new Student(15236, "Shraddha", 7.43);
            serializeStudent(student1, filename);
            Student deserializedStudent = deserializeStudent(filename);
            if (deserializedStudent != null) {
                deserializedStudent.display();
            }
        }
    }
```



```
PS C:\Users\abc> cd "c:\Users\abc\" ; if ($?) { javac StudentSerialization.jav
a } ; if ($?) { java StudentSerialization }
Serialization successful! Student object saved.
Deserialization successful! Student object loaded.
ID: 15236, Name: Shraddha, GPA: 7.43
```

HARD:

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

class Employee implements Serializable {
    private static final long serialVersionUID = 1L;
    private int id;
    private String name, designation;
    private 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("ID: " + id + ", Name: " + name + ", Designation: " + designation + ",
Salary: " + salary);
    }
}

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

    public static void addEmployee() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter Employee ID: ");
        int id = sc.nextInt();
        sc.nextLine();
        System.out.print("Enter Employee Name: ");
        String name = sc.nextLine();
        System.out.print("Enter Designation: ");
        String designation = sc.nextLine();
        System.out.print("Enter Salary: ");
        double salary = sc.nextDouble();

        try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(FILE_NAME, true))) {
            oos.writeObject(new Employee(id, name, designation, salary));
            System.out.println("Employee added successfully!\n");
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
    public static void displayAllEmployees() {
        try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(FILE_NAME)))
{
            System.out.println("\nEmployee List:");
            while (true) {
                ((Employee) ois.readObject()).display();
            }
        } catch (EOFException e) {
            System.out.println("End of employee records.\n");
        } catch (IOException | ClassNotFoundException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while (true) {
            System.out.println("1. Add Employee\n2. Display All Employees\n3. Exit");
            System.out.print("Enter your choice: ");
            int choice = sc.nextInt();
            switch (choice) {
                case 1:
                    addEmployee();
                    break;
                case 2:
                    displayAllEmployees();
                    break;
                case 3:
                    sc.close();
                    System.exit(0);
                default:
                    System.out.println("Invalid choice!");
            }
        }
    }
}
```



```
PS C:\Users\abc> cd "c:\Users\abc\" ; if ($?) { javac Menu_based_java_application.java } ; if ($?) { java Menu_based_java_app
1. Add Employee
2. Display All Employees
3. Exit
Enter your choice: 1
Enter Employee ID: 15236
Enter Employee Name: shraddha
Enter Designation: Director
Enter Salary: 100000
Employee added successfully!

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

Employee List:
ID: 15236, Name: shraddha, Designation: Director, Salary: 100000.0
End of employee records.

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

4. Learning Outcomes-

- Autoboxing **& Unboxing:** Efficiently convert between primitive types and their wrapper classes in Java.
- Serialization **& Deserialization:** Store and retrieve object states using file handling. Object**-Oriented Design:** Implement classes with attributes and methods, demonstrating encapsulation.
- File **I/O Operations:** Read from and write to files for persistent data storage.
- Menu**-Driven Programming:** Build interactive console applications with dynamic user input handling.