Exp-5

1. 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()).(Easy)

Code –

```java
import java.util.*;

public class SumCalculator {
    public static int calculateSum(List<Integer> numbers) {
        int sum = 0;
        for (Integer num : numbers) { // Auto-unboxing
            sum += num;
        }
        return sum;
    }

    public static List<Integer> parseNumbers(String[] numberStrings) {
        List<Integer> numbers = new ArrayList<>();
        for (String str : numberStrings) {
            numbers.add(Integer.parseInt(str)); // Autoboxing
        }
        return numbers;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter numbers separated by space: ");
        String input = scanner.nextLine();

        String[] numberStrings = input.split(" ");
        List<Integer> numbers = parseNumbers(numberStrings);

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

        scanner.close();
    }
}
```

Output –

```
PS C:\Users\Vaibhavi\dsa coding>  c:; cd 'c:\Us
java.exe' '-XX:+ShowCodeDetailsInExceptionMessa
ava\jdt_ws\dsa coding_5752218\bin' 'SumCalculat
Enter numbers separated by space: 3 4 5 6
Sum of numbers: 18
PS C:\Users\Vaibhavi\dsa coding>
```

2. 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.(Medium)
   Code –

```
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;
    }

    @Override
    public String toString() {
        return "Student ID: " + id + "\nName: " + name + "\nGPA: " + gpa;
    }
}

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

    public static void serializeStudent(Student student) {
        try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(FILE_NAME))) {
            oos.writeObject(student);
            System.out.println("Student object serialized successfully.");
        } catch (FileNotFoundException e) {
            System.err.println("Error: File not found.");
        } catch (IOException e) {
            System.err.println("Error: Unable to write object to file.");
        }
    }

    public static Student deserializeStudent() {
        try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream(FILE_NAME))) {
            return (Student) ois.readObject();
        } catch (FileNotFoundException e) {
            System.err.println("Error: File not found.");
        } catch (IOException e) {
            System.err.println("Error: Unable to read object from file.");
        } catch (ClassNotFoundException e) {
            System.err.println("Error: Class not found.");
        }
        return null;
    }
```

```
    public static void main(String[] args) {
        Student student = new Student(101, "Vaibhavi Shreya",8.36);
        serializeStudent(student);

        Student deserializedStudent = deserializeStudent();
        if (deserializedStudent != null) {
            System.out.println("Deserialized Student:\n" + deserializedStudent);
        }
    }
}
```

Output –

```
PS C:\Users\Vaibhavi\dsa coding>
java.exe' '-XX:+ShowCodeDetailsIn
ava\jdt_ws\dsa coding_5752218\bin
Student object serialized success
Deserialized Student:
Student ID: 101
Name: Vaibhavi Shreya
GPA: 8.36
```

3. 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.(Hard)

Code –

```
import java.io.*;

import java.util.*;

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

    @Override
    public String toString() {
```

```
        return "Employee ID: " + id + "\nName: " + name + "\nDesignation: " +
designation + "\nSalary: " + salary;
    }
}

public class EmployeeManagemenfile
 {
    private static final String FILE_NAME = "employees.ser";

    public static void addEmployee(Employee employee) {
        List<Employee> employees = getEmployees();
        employees.add(employee);
        try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(FILE_NAME))) {
            oos.writeObject(employees);
            System.out.println("Employee added successfully.");
        } catch (IOException e) {
            System.err.println("Error: Unable to write to file.");
        }
    }

    public static List<Employee> getEmployees() {
        try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream(FILE_NAME))) {
            return (List<Employee>) ois.readObject();
        } catch (FileNotFoundException e) {
            return new ArrayList<>();
        } catch (IOException | ClassNotFoundException e) {
            System.err.println("Error: Unable to read from file.");
            return new ArrayList<>();
        }
    }

    public static void displayEmployees() {
        List<Employee> employees = getEmployees();
        if (employees.isEmpty()) {
            System.out.println("No employees found.");
        } else {
            employees.forEach(System.out::println);
        }
    }

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

            switch (choice) {
                case 1:
                    System.out.print("Enter Employee ID: ");
                    int id = scanner.nextInt();
                    scanner.nextLine();
                    System.out.print("Enter Name: ");
```

```
                    String name = scanner.nextLine();
                    System.out.print("Enter Designation: ");
                    String designation = scanner.nextLine();
                    System.out.print("Enter Salary: ");
                    double salary = scanner.nextDouble();
                    addEmployee(new Employee(id, name, designation, salary));
                    break;
                case 2:
                    displayEmployees();
                    break;
                case 3:
                    System.out.println("Exiting...");
                    scanner.close();
                    return;
                default:
                    System.out.println("Invalid choice, please try again.");
            }
        }
    }
}
```

Output-

```
loyeeManagementFile'
1. Add Employee
2. Display All Employees
3. Exit
Enter your choice: 1
Enter Employee ID: 34
Enter Name: Saksham
Enter Designation: It
Enter Salary: 500000
Employee added successfully.
1. Add Employee
2. Display All Employees
3. Exit
Enter your choice:
```