Experiment 5

Student Name: Diksha
UID: 23BCS10994
Branch: CSE
Section/Group: KRG_2B
Semester: 5th
Date of Performance: 24/9/25
Subject Name: PBLJ
Subject Code: 23CSH-304

1. Aim:

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

A) 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()).

B) 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.

C) 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

2. Objectives:

- ❖ To demonstrate the use of Java Wrapper classes and automatic conversion between primitive types and their wrapper equivalents.
- ❖ To demonstrate object serialization, file handling, and exception management in Java.
- ❖ To combine object-oriented programming, file handling, and menu-driven console interaction.

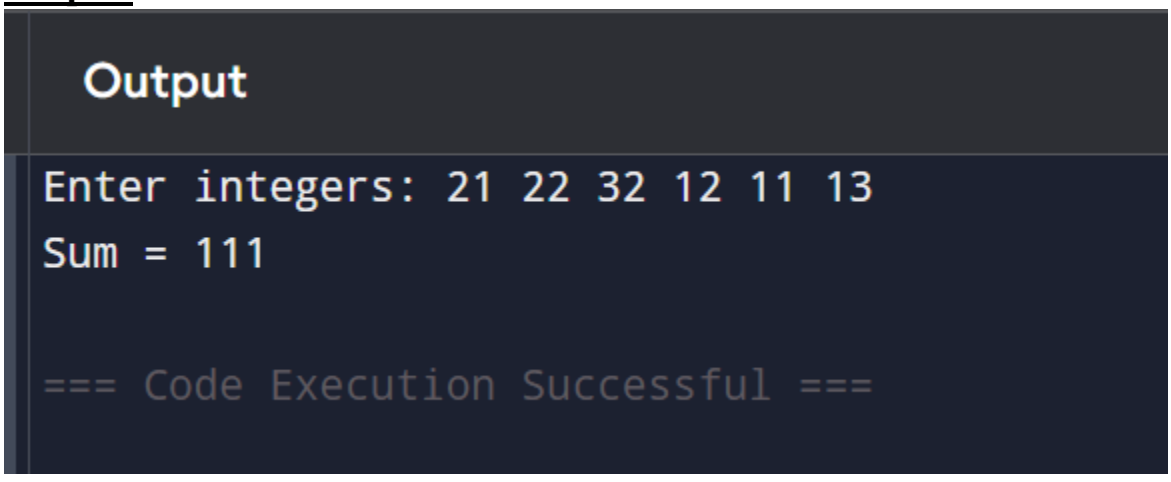
3. JAVA script and output:

EASY-LEVEL PROBLEM

```
import java.util.*;
public class SumList {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      List<Integer> numbers = new ArrayList<>();

      System.out.print("Enter integers: ");
      String input = sc.nextLine();
      String[] parts = input.split(" ");
      for (String s : parts) {
         numbers.add(Integer.parseInt(s));
      }
      int sum = 0;
      for (Integer num : numbers) {
         sum += num;
      }
      System.out.println("Sum = " + sum);
      sc.close();
   }
}
```

Output:



MEDIUM LEVEL PROBLEM:

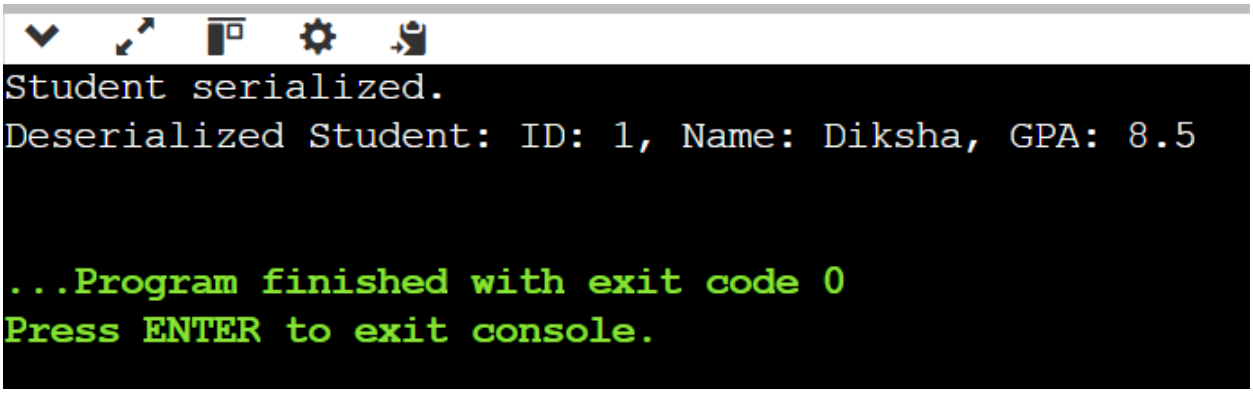
```
import java.io.*;
class Student implements Serializable {
    int id;
    String name;
    double gpa;
    Student(int id, String name, double gpa) {
        this.id = id;
        this.name = name;
        this.gpa = gpa;
    }
    public String toString() {
        return "ID: " + id + ", Name: " + name + ", GPA: " + gpa;
    }
}
public class StudentSerialization {
    public static void main(String[] args) {
        String filename = "student.dat";

        try {
            Student s = new Student(1, "Diksha", 8.5);
            ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(filename));
            oos.writeObject(s);
            oos.close();
            System.out.println("Student serialized.");
```

```
            ObjectInputStream ois = new ObjectInputStream(new FileInputStream(filename));
            Student deserialized = (Student) ois.readObject();
            ois.close();
            System.out.println("Deserialized Student: " + deserialized);

        } catch (FileNotFoundException e) {
            System.out.println("File not found: " + e.getMessage());
        } catch (IOException e) {
            System.out.println("I/O Error: " + e.getMessage());
        } catch (ClassNotFoundException e) {
            System.out.println("Class not found: " + e.getMessage());
        }
    }
}
```

Output:



HARD LEVEL PROBLEM

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

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 EmployeeApp {
    static String filename = "employees.dat";
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        List<Employee> employees = new ArrayList<>();
        try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(filename))) {
            employees = (List<Employee>) ois.readObject();
        } catch (Exception e) {
        }

        while (true) {
            System.out.println("\n1. Add Employee");
            System.out.println("2. Display All");
            System.out.println("3. Exit");
            System.out.print("Enter choice: ");
            int choice = sc.nextInt();
```

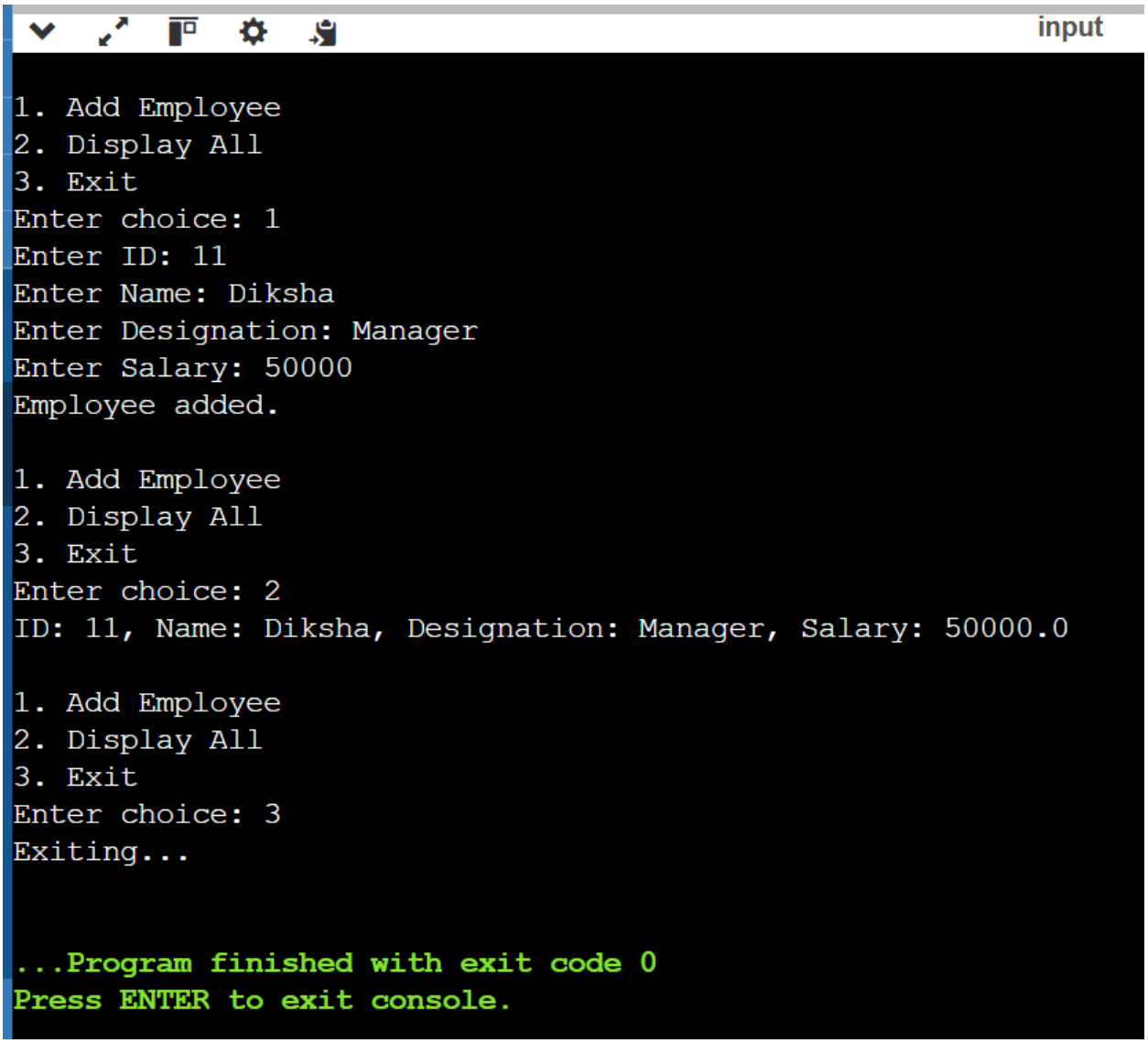
```
            switch (choice) {
                case 1:
                    System.out.print("Enter ID: ");
                    int id = sc.nextInt();
                    sc.nextLine();
                    System.out.print("Enter Name: ");
                    String name = sc.nextLine();
                    System.out.print("Enter Designation: ");
                    String designation = sc.nextLine();
                    System.out.print("Enter Salary: ");
                    double salary = sc.nextDouble();
                    employees.add(new Employee(id, name, designation, salary));

                    try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(filename))) {
                        oos.writeObject(employees);
                    } catch (IOException e) {
                        System.out.println("Error saving: " + e.getMessage());
                    }
                    System.out.println("Employee added.");
                    break;
                case 2:
                    if (employees.isEmpty()) {
                        System.out.println("No employees found.");
                    } else {
                        for (Employee emp : employees) {
                            System.out.println(emp);
                        }
                    }
                    break;
                case 3:
                    System.out.println("Exiting...");
                    sc.close();
                    return;

                default:
                    System.out.println("Invalid choice.");
            }
        }
```

```
    }
}
```

OUTPUT:


input
1. Add Employee
2. Display All
3. Exit
Enter choice: 1
Enter ID: 11
Enter Name: Diksha
Enter Designation: Manager
Enter Salary: 50000
Employee added.
1. Add Employee
2. Display All
3. Exit
Enter choice: 2
ID: 11, Name: Diksha, Designation: Manager, Salary: 50000.0
1. Add Employee
2. Display All
3. Exit
Enter choice: 3
Exiting...
...Program finished with exit code 0
Press ENTER to exit console.