Experiment 5.1

Student Name: Shiwani Agarwal	**UID: 22BCS15973**
Branch: CSE	**Section/Group:22BCS_IOT-640/A**
Semester: 6th	**Date of Performance:24/02/25**
Subject Name: PBLJ	**Subject Code: 22CSH-359**

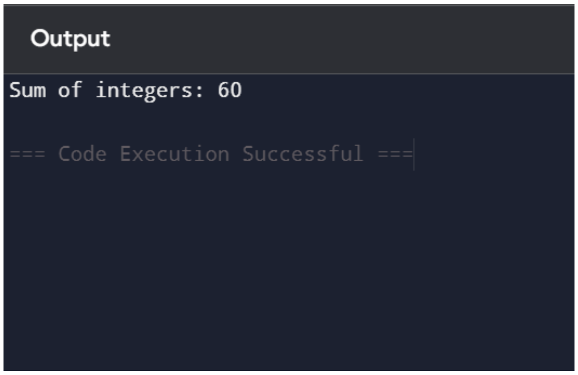
1. **Aim:** Write a Java program to calculate the sum of a list of integers using autoboxing and unboxing.

2. **Objective:** 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())

3. **Implementation/Code:**

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

public class SumOfIntegers {

    public static void main(String[] args) {
        // Example list of integers as strings
        List<String> integerStrings = new ArrayList<>();
        integerStrings.add("10");
        integerStrings.add("20");
        integerStrings.add("30");

        // Calculate sum of integers in the list
        int sum = calculateSum(integerStrings);
        System.out.println("Sum of integers: " + sum);
    }

    public static int calculateSum(List<String> integerStrings) {
        int sum = 0;
        for (String str : integerStrings) {
            // Parse each string to Integer using Integer.parseInt()
            Integer num = Integer.parseInt(str); // Autoboxing
            sum += num; // Unboxing
        }
        return sum;
    }
```

4. **Output:**



```
Sum of integers: 60

=== Code Execution Successful ===
```

Experiment 5.2

1. **Aim :** Create a Java program to serialize and deserialize a Student object.

2. **Objective :**

   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.

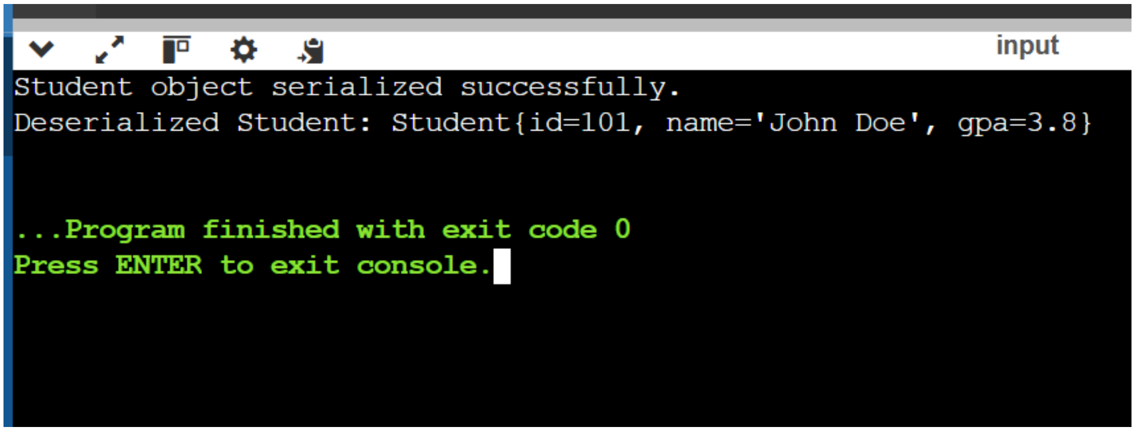
   Deserialize the object from the file and display the student details.

   Handle FileNotFoundException, IOException, and ClassNotFoundException using exception handling.

3. **Implementation /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 + ", name='" + name + '\'' + ", gpa=" + 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 (IOException e) {
            e.printStackTrace();
        }
    }

    public static void deserializeStudent() {
        try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream(FILE_NAME))) {
            Student student = (Student) ois.readObject();
            System.out.println("Deserialized Student: " + student);
        } catch (FileNotFoundException e) {
            System.err.println("Error: File not found!");
        } catch (IOException e) {
            System.err.println("Error: IO Exception occurred!");
        } catch (ClassNotFoundException e) {
            System.err.println("Error: Class not found!");
        }
    }
    public static void main(String[] args) {
        Student student = new Student(101, "John Doe", 3.8);
        serializeStudent(student);
        deserializeStudent();
    }
}
```

4. **Output :**



```
Student object serialized successfully.
Deserialized Student: Student{id=101, name='John Doe', gpa=3.8}


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

Experiment 5.3

1. **Aim** : Create a menu-based Java application

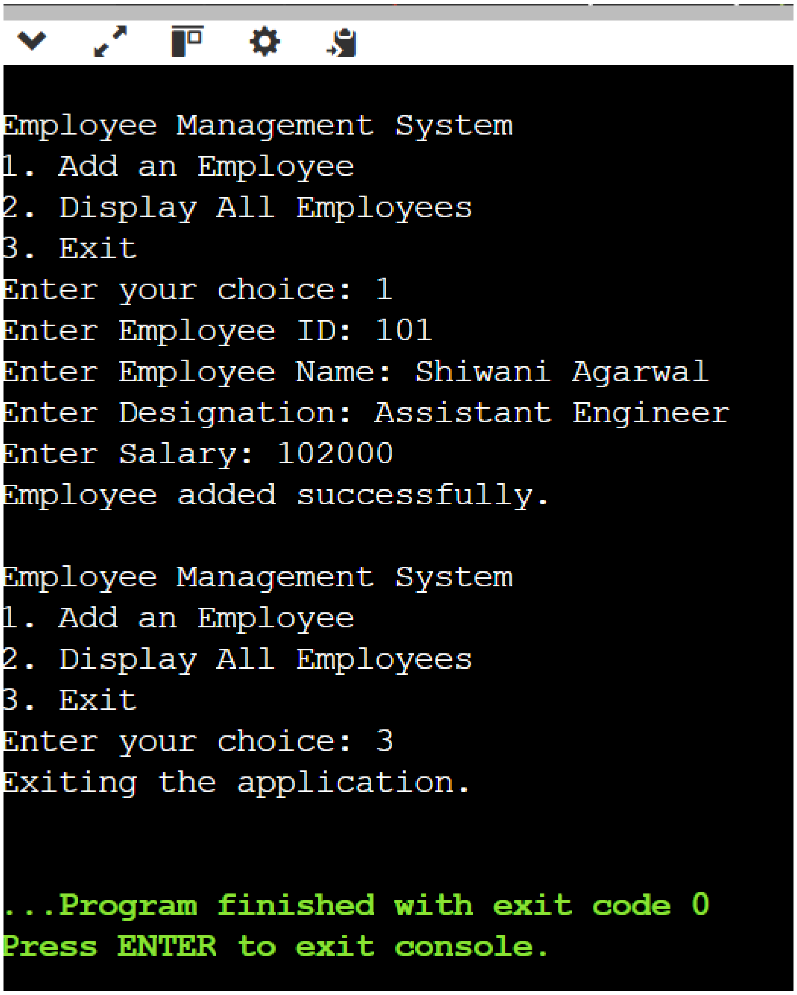
2. **Objective :**

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.

3. **Implementation /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 + ", name='" + name + '\'' + ", designation='"
+ designation + '\'' + ", salary=" + salary + '}';
    }
}

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

    public static void addEmployee() {
        try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(FILE_NAME, true))) {
            Scanner scanner = new Scanner(System.in);
            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 employee = new Employee(id, name, designation, salary);
            oos.writeObject(employee);
            System.out.println("Employee added successfully.");
        } catch (IOException e) {
            System.err.println("Error: Unable to add employee.");
        }
    }

    public static void displayEmployees() {
        try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream(FILE_NAME))) {
            while (true) {
                try {
                    Employee employee = (Employee) ois.readObject();
                    System.out.println(employee);
                } catch (EOFException e) {
                    break;
                }
            }
        } catch (FileNotFoundException e) {
            System.err.println("Error: No employee records found.");
        } catch (IOException | ClassNotFoundException e) {
            System.err.println("Error: Unable to display employees.");
        }
    }

    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 an Employee");
            System.out.println("2. Display All Employees");
            System.out.println("3. Exit");
            System.out.print("Enter your choice: ");
            int choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    addEmployee();
                    break;
                case 2:
                    displayEmployees();
                    break;
                case 3:
                    System.out.println("Exiting the application.");
                    return;
                default:
                    System.out.println("Invalid choice. Please try again.");
            }
        }
    }
}
```

4. **Output :**



```
Employee Management System
1. Add an Employee
2. Display All Employees
3. Exit
Enter your choice: 1
Enter Employee ID: 101
Enter Employee Name: Shiwani Agarwal
Enter Designation: Assistant Engineer
Enter Salary: 102000
Employee added successfully.

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


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

5. **Learning Outcomes :**

- Learn about Serialization and Deserialization.
- Understanding File Handling in Java.
- Exceptional Handling in Java.
- Enhancing Problem-Solving Skills.