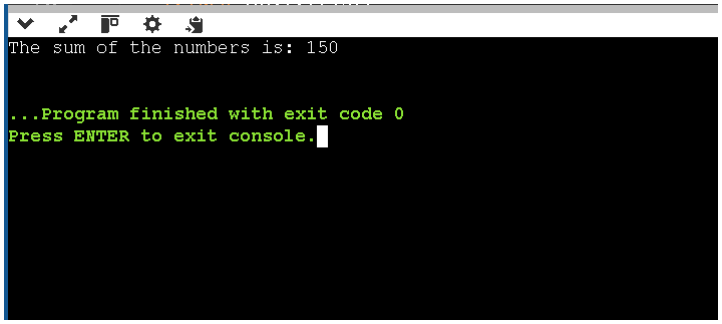
EXPERIMENT-5.1

CODE

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

public class AutoboxingExample {
    public static void main(String[] args) {
        String[] numberStrings = {"10", "20", "30", "40", "50"};

        List<Integer> numbers = parseStringArrayToIntegers(numberStrings);

        int sum = calculateSum(numbers);

        System.out.println("The sum of the numbers is: " + sum);
    }

    public static List<Integer> parseStringArrayToIntegers(String[] strings) {
        List<Integer> integerList = new ArrayList<>();
        for (String str : strings) {
            integerList.add(Integer.parseInt(str));
        }
        return integerList;
    }

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

OUTPUT



Experiment 5.2

1.Aim: 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.

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 main(String[] args) {
        Student student = new Student(1, "Preeti", 7.8);
        serializeStudent(student);
        deserializeStudent();
    }

    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("File not found: " + e.getMessage());
        } catch (IOException e) {
            System.err.println("IOException occurred: " + e.getMessage());
        }
    }

    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("File not found: " + e.getMessage());
        } catch (IOException e) {
            System.err.println("IOException occurred: " + e.getMessage());
        } catch (ClassNotFoundException e) {
            System.err.println("Class not found: " + e.getMessage());
        }
    }
}
```

OUTPUT

```
Student object serialized successfully.
Deserialized Student: Student{id=1, name='Preeti', gpa=7.8}


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

Experiment 5.3

1. **Aim:** 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.

CODE

```
import java.util.ArrayList;
import java.util.Iterator;
import java.util.Scanner;

class Employee {
    int id;
    String name;
    double salary;

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

    @Override
    public String toString() {
        return "ID: " + id + ", Name: " + name + ", Salary: " + salary;
    }
}

public class EmployeeManagement {
    static ArrayList<Employee> employeeList = new ArrayList<>();
    static Scanner scanner = new Scanner(System.in);

    public static void addEmployee() {
        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 Employee Salary: ");
        double salary = scanner.nextDouble();

        employeeList.add(new Employee(id, name, salary));
        System.out.println("Employee added successfully!");
    }
 public static void updateEmployee() {
        System.out.print("Enter Employee ID to update: ");
        int id = scanner.nextInt();

        for (Employee emp : employeeList) {
            if (emp.id == id) {
                scanner.nextLine(); // Consume newline
                System.out.print("Enter new Name: ");
                emp.name = scanner.nextLine();

                System.out.print("Enter new Salary: ");
                emp.salary = scanner.nextDouble();

                System.out.println("Employee details updated successfully!");
                return;
            }
        }
        System.out.println("Employee not found!");
    }
public static void removeEmployee() {
        System.out.print("Enter Employee ID to remove: ");
        int id = scanner.nextInt();

        Iterator<Employee> iterator = employeeList.iterator();
        while (iterator.hasNext()) {
            Employee emp = iterator.next();
            if (emp.id == id) {
                iterator.remove();
                System.out.println("Employee removed successfully!");
                return;
            }
        }
        System.out.println("Employee not found!");
    }
```

```java
public static void searchEmployee() {
        System.out.print("Enter Employee ID to search: ");
        int id = scanner.nextInt();

        for (Employee emp : employeeList) {
            if (emp.id == id) {
                System.out.println(emp);
                return;
            }
        }
        System.out.println("Employee not found!");
    }

    public static void displayEmployees() {
        if (employeeList.isEmpty()) {
            System.out.println("No employees found.");
        } else {
            for (Employee emp : employeeList) {
                System.out.println(emp);
            }
        }
public static void main(String[] args) {
        while (true) {
            System.out.println("\nEmployee Management System");
            System.out.println("1. Add Employee");
            System.out.println("2. Update Employee");
            System.out.println("3. Remove Employee");
            System.out.println("4. Search Employee");
            System.out.println("5. Display All Employees");
            System.out.println("6. Exit");
            System.out.print("Choose an option: ");

            int choice;
            try {
                choice = scanner.nextInt();
            } catch (Exception e) {
                System.out.println("Invalid input! Please enter a number.");
                scanner.nextLine(); // Clear buffer
                continue;
            }
switch (choice) {
                case 1 -> addEmployee();
                case 2 -> updateEmployee();
                case 3 -> removeEmployee();
                case 4 -> searchEmployee();
                case 5 -> displayEmployees();
                case 6 -> {
                    System.out.println("Exiting... Goodbye!");
                    scanner.close();
                    return;
                }
                default -> System.out.println("Invalid choice! Please try again.");
            }
        }
    }
}
```

Run EmployeeManagement

```
1. Add Employee
2. Update Employee
3. Remove Employee
4. Search Employee
5. Display All Employees
6. Exit
Choose an option: 1
Enter Employee ID: 11248
Enter Employee Name: preeti verma
Enter Employee Salary: 100000
Employee added successfully!

Employee Management System
1. Add Employee
2. Update Employee
3. Remove Employee
4. Search Employee
5. Display All Employees
6. Exit
Choose an option: 4
Enter Employee ID to search: 11248
ID: 11248, Name: preeti verma, Salary: 100000.0

Employee Management System
1. Add Employee
2. Update Employee
3. Remove Employee
4. Search Employee
5. Display All Employees
6. Exit
Choose an option:
```