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()).

Code-

```
import java.util.*;
public class SumUsingAutoboxing {
      public static int calculateSum(List<String> numberStrings) {
      int sum = 0;
      for (String numStr : numberStrings) {
         Integer num = Integer.parseInt(numStr);
         sum += num;
      }
      return sum;
   }
    public static void main(String[] args) {
      List<String> numberStrings = Arrays.asList("10", "20", "30", "40", "50");
      int result = calculateSum(numberStrings);
      System.out.println("Sum of numbers: " + result);
   }
}
```

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.

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;
   }
   public void display() {
      System.out.println("Student ID: " + id);
      System.out.println("Name: " + name);
      System.out.println("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 (FileNotFoundException e) {
         System.out.println("File not found: " + e.getMessage());
      } catch (IOException e) {
         System.out.println("IO Exception during serialization: " + e.getMessage());
      }
   }
   public static Student deserializeStudent() {
      try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(FILE_NAME))) {
         return (Student) ois.readObject();
      } catch (FileNotFoundException e) {
         System.out.println("File not found: " + e.getMessage());
      } catch (IOException e) {
         System.out.println("IO Exception during deserialization: " + e.getMessage());
      } catch (ClassNotFoundException e) {
         System.out.println("Class not found: " + e.getMessage());
      }
      return null;
   }
   public static void main(String[] args) {
      Student student = new Student(101, "Alice", 3.8);
      serializeStudent(student);
      Student deserializedStudent = deserializeStudent();
      if (deserializedStudent != null) {
         System.out.println("\nDeserialized Student Details:");
         deserializedStudent.display();
      }
   }
}
```

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.

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 EmployeeManagementSystem {
    private static final String FILE_NAME = "employees.dat";

    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!\n");
        } catch (IOException e) {
            System.out.println("Error saving employee: " + e.getMessage());
        }
    }
    @SuppressWarnings("unchecked")
    public static List<Employee> getEmployees() {
        File file = new File(FILE_NAME);
        if (!file.exists()) return new ArrayList<>();
        try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(FILE_NAME))) {
            return (List<Employee>) ois.readObject();
        } catch (IOException | ClassNotFoundException e) {
            System.out.println("Error reading employees: " + e.getMessage());
            return new ArrayList<>();
        }
    }
    public static void displayEmployees() {
        List<Employee> employees = getEmployees();
        if (employees.isEmpty()) {
            System.out.println("No employees found.\n");
        } else {
            System.out.println("\nEmployee Details:");
```

```
        for (Employee emp : employees) {
            System.out.println(emp);
        }
        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("Choose an option: ");
      int choice = scanner.nextInt();
      scanner.nextLine();
      switch (choice) {
        case 1:
          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 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 application...");
          scanner.close();
          return;
        default:
          System.out.println("Invalid choice! Please try again.\n");
      }
    }
  }
}
```