1.) Part a: Sum of Integers Using Autoboxing and Unboxing

```
// Part (a) - Autoboxing & Unboxing
public class AutoboxingDemo {
    public static void main(String[] args) {
        int a = 10;
        int b = 20;

        // Autoboxing: converting primitive to wrapper
        Integer objA = a;
        Integer objB = b;

        // Unboxing: converting wrapper to primitive
        int sum = objA + objB;

        System.out.println("Integer A: " + objA);
        System.out.println("Integer B: " + objB);
        System.out.println("Sum using Autoboxing and Unboxing: " + sum);
    }
}
```

2.) Part b: Serialization and Deserialization of a Student Object

```
import java.io.*;

// Serializable Student class
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;
    }

    void display() {
        System.out.println("ID: " + id + ", Name: " + name + ", GPA: " + gpa);
    }
}

public class SerializationDemo {
    public static void main(String[] args) {
        String filename = "student.ser";

        // Serialization
        try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(filename))) {
```

```
            Student s1 = new Student(101, "Saksham", 8.9);
            oos.writeObject(s1);
            System.out.println("Student object serialized successfully.");
        } catch (IOException e) {
            e.printStackTrace();
        }

        // Deserialization
        try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(filename))) {
            Student s2 = (Student) ois.readObject();
            System.out.println("Deserialized Student object:");
            s2.display();
        } catch (IOException | ClassNotFoundException e) {
            e.printStackTrace();
        }
    }
}
```

3.) Part c: Menu-Based Employee Management System Using File Handling

```
import java.io.*;
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;
    }

    public String toString() {
        return id + "," + name + "," + salary;
    }
}

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

    // Add employee
    static void addEmployee(Employee emp) {
        try (FileWriter fw = new FileWriter(FILE_NAME, true);
             BufferedWriter bw = new BufferedWriter(fw);
             PrintWriter pw = new PrintWriter(bw)) {
            pw.println(emp);
```

```
        System.out.println("Employee added successfully.");
    } catch (IOException e) {
        e.printStackTrace();
    }
}

// View all employees
static void viewEmployees() {
    try (BufferedReader br = new BufferedReader(new FileReader(FILE_NAME))) {
        String line;
        System.out.println("---- Employee Records ----");
        while ((line = br.readLine()) != null) {
            System.out.println(line);
        }
        System.out.println("--------------------------");
    } catch (IOException e) {
        System.out.println("No employees found.");
    }
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int choice;

    do {
        System.out.println("\nEmployee Management System");
        System.out.println("1. Add Employee");
        System.out.println("2. View Employees");
        System.out.println("3. Exit");
        System.out.print("Enter choice: ");
        choice = sc.nextInt();

        switch (choice) {
            case 1:
                System.out.print("Enter Employee ID: ");
                int id = sc.nextInt();
                sc.nextLine(); // consume newline
                System.out.print("Enter Employee Name: ");
                String name = sc.nextLine();
                System.out.print("Enter Salary: ");
                double salary = sc.nextDouble();
                Employee emp = new Employee(id, name, salary);
                addEmployee(emp);
                break;
            case 2:
                viewEmployees();
                break;
            case 3:
```

```
                System.out.println("Exiting...");
                break;
            default:
                System.out.println("Invalid choice!");
        }
    } while (choice != 3);

    sc.close();
  }
}
```

Outputs:-