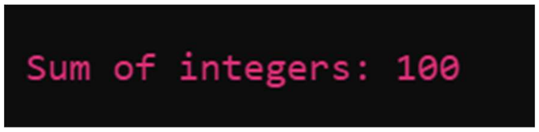
Part A: Autoboxing, Unboxing, and Sum Calculation

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

public class AutoboxingUnboxing {
  public static void main(String[] args) {
    String[] numberStrings = {"10", "20",
"30", "40"};
    List<Integer> integerList = new
ArrayList<>();

    // Autoboxing: Converting primitive int to
Integer
    for (String number : numberStrings) {

integerList.add(Integer.parseInt(number)); //
Parsing string to Integer
    }

    int sum = 0;
    // Unboxing: Converting Integer to int
    for (Integer num : integerList) {
      sum += num;
    }

    System.out.println("Sum of integers: " +
sum);
  }
}
```

OUTPUT-

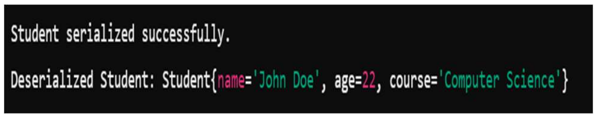


```
Sum of integers: 100
```

Part B: Serialization and Deserialization of a Student Object

```
import java.io.*;

class Student implements Serializable {
  private static final long serialVersionUID =
1L;
  private String name;
  private int age;
  private String course;

  public Student(String name, int age, String
course) {
    this.name = name;
    this.age = age;
    this.course = course;
  }

  @Override
  public String toString() {
    return "Student{name='" + name + "',
age=" + age + ", course='" + course + "'}";
  }
}

public class StudentSerialization {
  public static void main(String[] args) {
    Student student = new Student("John
Doe", 22, "Computer Science");

    // Serialization
    try (ObjectOutputStream out = new
ObjectOutputStream(new
FileOutputStream("student.ser"))) {
      out.writeObject(student);
      System.out.println("Student serialized
successfully.");
    } catch (IOException e) {
      e.printStackTrace();
    }

    // Deserialization
    try (ObjectInputStream in = new
ObjectInputStream(new
FileInputStream("student.ser"))) {
      Student deserializedStudent = (Student)
in.readObject();
      System.out.println("Deserialized
Student: " + deserializedStudent);
    } catch (IOException |
ClassNotFoundException e) {
      e.printStackTrace();
    }
  }
}
```

OUTPUT-



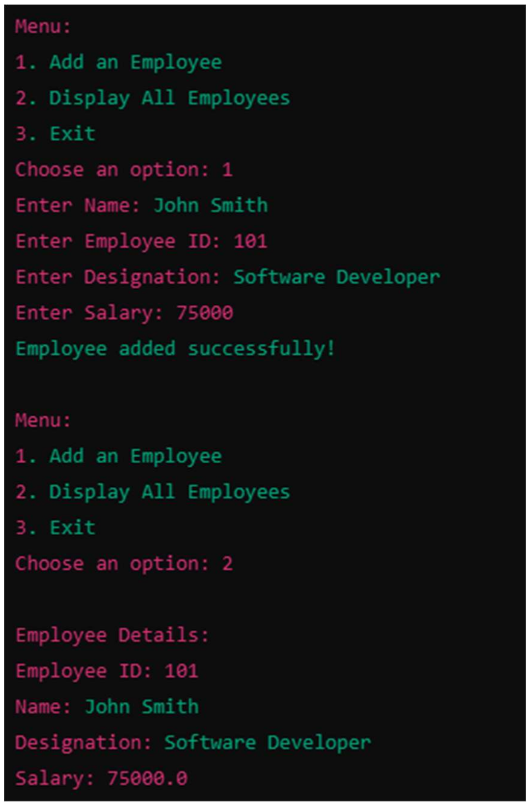
```
Student serialized successfully.

Deserialized Student: Student{name='John Doe', age=22, course='Computer Science'}
```

Part C: Menu-Based Employee Management System

```
import java.io.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

class Employee implements Serializable {
  private static final long serialVersionUID =
1L;
  private String name;
  private int employeeId;
  private String designation;
  private double salary;

  public Employee(String name, int
employeeId, String designation, double salary)
{
    this.name = name;
    this.employeeId = employeeId;
    this.designation = designation;
    this.salary = salary;
  }

  @Override
  public String toString() {
    return "Employee ID: " + employeeId +
"\nName: " + name + "\nDesignation: " +
designation + "\nSalary: " + salary + "\n";
  }
}

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

  public static void main(String[] args) {
    Scanner scanner = new
Scanner(System.in);
    List<Employee> employees =
loadEmployees();

    while (true) {
      System.out.println("\nMenu:");
      System.out.println("1. Add an
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(); // Consume newline

      switch (choice) {
        case 1:
          addEmployee(scanner, employees);
          saveEmployees(employees);
          break;
        case 2:
          displayEmployees(employees);
          break;
        case 3:
          System.out.println("Exiting
program.");
          scanner.close();
          return;
        default:
          System.out.println("Invalid choice!
Please select a valid option.");
      }
    }
  }

  private static void addEmployee(Scanner
scanner, List<Employee> employees) {
    System.out.print("Enter Name: ");
    String name = scanner.nextLine();
    System.out.print("Enter Employee ID: ");
    int id = scanner.nextInt();
    scanner.nextLine(); // Consume newline
    System.out.print("Enter Designation: ");
    String designation = scanner.nextLine();
    System.out.print("Enter Salary: ");
    double salary = scanner.nextDouble();

    employees.add(new Employee(name, id,
designation, salary));
    System.out.println("Employee added
successfully!");
  }

  private static void
displayEmployees(List<Employee> employees)
{
    if (employees.isEmpty()) {
      System.out.println("No employees
found.");
    } else {
      System.out.println("\nEmployee
Details:");
      for (Employee employee : employees) {
        System.out.println(employee);
      }
    }
  }
```

```
    private static void
saveEmployees(List<Employee> employees) {
        try (ObjectOutputStream out = new
ObjectOutputStream(new
FileOutputStream(FILE_NAME))) {
            out.writeObject(employees);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    private static List<Employee>
loadEmployees() {
        try (ObjectInputStream in = new
ObjectInputStream(new
FileInputStream(FILE_NAME))) {
            return (List<Employee>)
in.readObject();
        } catch (IOException |
ClassNotFoundException e) {
            return new ArrayList<>();
        }
    }
}
```

OUTPUT-



```
Menu:
1. Add an Employee
2. Display All Employees
3. Exit
Choose an option: 1
Enter Name: John Smith
Enter Employee ID: 101
Enter Designation: Software Developer
Enter Salary: 75000
Employee added successfully!

Menu:
1. Add an Employee
2. Display All Employees
3. Exit
Choose an option: 2

Employee Details:
Employee ID: 101
Name: John Smith
Designation: Software Developer
Salary: 75000.0
```