Experiment5

StudentName: Harshpal Singh **UID:**22BCS10869
Branch:BECSE **Section/Group:**EPAM-801(B)
Semester: 06 **DateofPerformance:**23-2-25
SubjectName: ProjectBasedLearninginJava **SubjectCode:**22CSH-359

1. **Aim-**

Easy: WriteaJavaprogramtocalculatethesumofalistofintegersusingautoboxingand unboxing. Include methods to parse strings into their respective wrapper classes (e.g., Integer.parseInt()).

Medium: CreateaJavaprogramtoserializeanddeserializeaStudentobject.Theprogram should:Serialize a Student object (containing id, name, and GPA) and save it to a file. Deserializetheobjectfromthefileanddisplaythestudentdetails.

Handle FileNotFoundException, IOException, and ClassNotFoundException using exception handling.

Hard: Create a menu-based Java application with the following options. 1.Add an Employee2.DisplayAll3.ExitIfoption1isselected,theapplicationshouldgatherdetails oftheemployee likeemployeename, employee id, designationandsalaryandstoreit ina file.Ifoption2isselected,theapplicationshoulddisplayalltheemployeedetails.Ifoption 3 is selected the application should exit.

2. **Procedure-**

EasyLevel:SumofIntegers

1. Initializeanempty list.
2. Takeuserinputsuntil"end"is entered.
3. Converteachinputtoaninteger(autoboxing)andaddtothe list.
4. CalculatethesumbyunboxingeachInteger.
5. Displaythesum.

MediumLevel:Serializationand Deserialization

1. CreateaStudentclassimplementingSerializable.
2. Serialize:
   - CreateaStudentobject.
   - Save ittoafileusing ObjectOutputStream.
3. Deserialize:
   - Readtheobjectfromthefileusing ObjectInputStream.
   - Displaytheobjectdata.

HardLevel:EmployeeManagement

1. Displayamenu:

- AddEmployee
- DisplayAllEmployees
- Exit

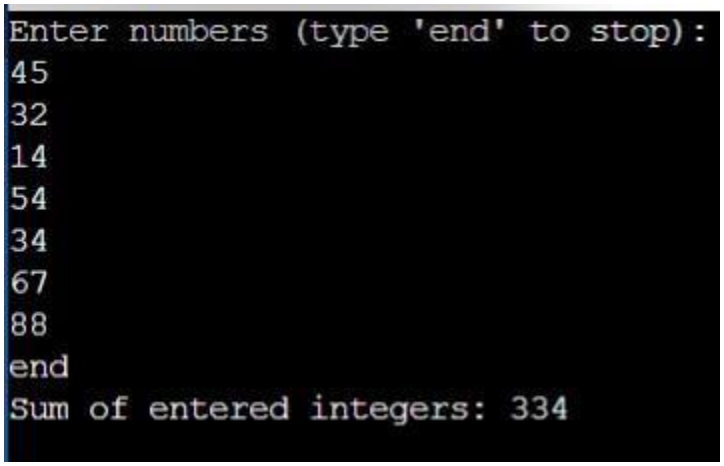
2. For AddEmployee:
   - Takeinput forID,Name,Designation, andSalary.
   - SaveitasanEmployeeobjectinalist.
   - Serializethelisttoafile.
3. ForDisplayAllEmployees:
   - Deserializethe listfromthe file.
   - Displayeachemployee's details.
4. Exittheprogramonuserchoice.

3. Code-

EASY:

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

publicclassSumOfIntegers{
   public static void main(String[] args) {
      ArrayList<Integer>numbers=newArrayList<>();
      Scanner sc = new Scanner(System.in);

      System.out.println("Enternumbers(type'end'tostop):"); while

      (true) {
         Stringinput= sc.next();
         if(input.equalsIgnoreCase("end")){
            break;
         }
         try{
            //Autoboxing:ConvertingprimitiveinttoInteger
            numbers.add(Integer.parseInt(input));
         } catch (NumberFormatException e) {
            System.out.println("Invalidinput,pleaseenteraninteger.");
         }
      }

      intsum=0;
      for(Integernum: numbers){
         //Unboxing:ConvertingIntegertoprimitiveint
         sum += num;
      }
```

```
        System.out.println("Sumofenteredintegers:"+sum); sc.close();
    }
}
```



MEDIUM:

```
importjava.io.*;

//SerializableClass
classStudentimplementsSerializable{
    privatestaticfinallongserialVersionUID=1L; int
    id;
    Stringname;
    double gpa;

    publicStudent(intid,Stringname,doublegpa){ this.id
        = id;
        this.name=name;
        this.gpa = gpa;
    }

    @Override
    publicStringtoString() {
        return"ID: "+ id +",Name:"+name+",GPA:"+ gpa;
    }
}

publicclassStudentSerialization{

    //SerializetheStudentObject
    publicstaticvoidserializeStudent(Student student,Stringfilename) {
```

```
        try(ObjectOutputStreamout=newObjectOutputStream(new
FileOutputStream(filename))) {
            out.writeObject(student);
            System.out.println("Studentserializedsuccessfully!");
        } catch (FileNotFoundException e) {
            System.out.println("Filenotfound:"+e.getMessage());
        }catch(IOExceptione){
            System.out.println("IOException:"+e.getMessage());
        }
    }

    //DeserializetheStudentObject
    publicstaticvoiddeserializeStudent(Stringfilename){
        try(ObjectInputStreamin=newObjectInputStream(new
FileInputStream(filename))) {
            Student student = (Student) in.readObject();
            System.out.println("DeserializedStudent:"+student);
        } catch (FileNotFoundException e) {
            System.out.println("Filenotfound:"+e.getMessage());
        }catch(IOExceptione){
            System.out.println("IOException:"+e.getMessage());
        } catch (ClassNotFoundException e) {
            System.out.println("Classnotfound:"+e.getMessage());
        }
    }

    publicstatic voidmain(String[]args){
        Studentstudent=newStudent(101,"Dipesh",8.5); String
        filename = "student.ser";

        //
        SerializeserializeStudent(student,fil
        ename);

        // Deserialize
        deserializeStudent(filename);
    }
}
```

```
Student serialized successfully!
Deserialized Student: ID: 101, Name: Dipesh, GPA: 8.5
```

HARD:

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

//SerializableClass
classEmployeeimplementsSerializable{
    privatestaticfinallongserialVersionUID=1L; int
    id;
    String name;
    Stringdesignation;
    double salary;

    publicEmployee(intid,Stringname,Stringdesignation,doublesalary){ this.id =
        id;
        this.name = name;
        this.designation=designation;
        this.salary = salary;
    }

    @Override
    publicStringtoString() {
        return"ID:"+id+",Name:"+name+",Designation:"+designation+",Salary: " + salary;
    }
}

publicclassEmployeeManagement{
    privatestaticfinalStringFILE_NAME ="employees.dat";

    // Method to add an employee
    publicstaticvoidaddEmployee(){
        Scanner sc = new Scanner(System.in);
        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 Designation: ");
        String designation = sc.nextLine();
        System.out.print("Enter Salary: ");
        doublesalary=sc.nextDouble();
```

```
        Employeeemployee=newEmployee(id,name,designation,salary);
        ArrayList<Employee> employees = readEmployees();
        employees.add(employee);
        writeEmployees(employees);

        System.out.println("Employeeaddedsuccessfully!");
    }

    //Methodtodisplayallemployees
    public static void displayAllEmployees() {
        ArrayList<Employee> employees = readEmployees();
        if (employees.isEmpty()) {
            System.out.println("Noemployeesfound.");
        }else{
            for(Employeeemp:employees){
                System.out.println(emp);
            }
        }
    }

    // Methodto reademployeesfromthefile
    public static ArrayList<Employee> readEmployees() {
        ArrayList<Employee>employees=new ArrayList<>();
        try(ObjectInputStreamin=newObjectInputStream(new
FileInputStream(FILE_NAME))){
            employees=(ArrayList<Employee>)in.readObject();
        } catch (FileNotFoundException e) {
            System.out.println("Noexistingrecordsfound.");
        } catch (IOException | ClassNotFoundException e) {
            System.out.println("Errorreadingemployeedata:"+e.getMessage());
        }
        returnemployees;
    }

    //Methodto writeemployeestothe file
    publicstaticvoidwriteEmployees(ArrayList<Employee>employees){ try
        (ObjectOutputStream out = new ObjectOutputStream(new
FileOutputStream(FILE_NAME))) {
            out.writeObject(employees);
        }catch(IOExceptione){
            System.out.println("Errorwritingemployeedata:"+e.getMessage());
        }
    }
```

```
    //MainMenu
    public static void main(String[] args) {
        Scannersc=newScanner(System.in);

        while(true){
            System.out.println("\n---EmployeeManagementSystem---");
            System.out.println("1. Add an Employee");
            System.out.println("2. Display All Employees");
            System.out.println("3. Exit");
            System.out.print("Chooseanoption:");
            int choice = sc.nextInt();

            switch(choice){ case
                1:
                    addEmployee();
                    break;
                case2:
                    displayAllEmployees();
                    break;
                case3:
                    System.out.println("Exiting...");
                    sc.close();
                    System.exit(0);
                    break;
                default:
                    System.out.println("Invalidchoice.Pleasetryagain.");
            }
        }
    }
}
```

```
2. Display All Employees
3. Exit
Choose an option: 1
Enter Employee ID: 101
Enter Employee Name: Garisha
Enter Designation: Software Manager
Enter Salary: 1000000
No existing records found.
Employee added successfully!

--- Employee Management System ---
1. Add an Employee
2. Display All Employees
3. Exit
Choose an option: 1
Enter Employee ID: 105
Enter Employee Name: Yashika
Enter Designation: Director
Enter Salary: 1500000
Employee added successfully!

--- Employee Management System ---
1. Add an Employee
2. Display All Employees
3. Exit
Choose an option: 2
ID: 101, Name: Garisha, Designation: Software Manager, Salary: 1000000.0
ID: 105, Name: Yashika, Designation: Director, Salary: 1500000.0

--- Employee Management System ---
1. Add an Employee
2. Display All Employees
3. Exit
Choose an option: 3
Exiting...
```

4. LearningOutcomes-

- Autoboxing**&Unboxing:**Efficientlyconvertbetweenprimitivetypesandtheir wrapper classes in Java.
- Serialization**&Deserialization:**Storeandretrieveobjectstatesusingfilehandling.
- Object-**OrientedDesign:**Implementclasseswithattributesandmethods, demonstrating encapsulation.
- File**I/OOperations:**Readfromandwritetofilesfor persistentdatastorage.
- Menu-**DrivenProgramming:**Buildinteractiveconsoleapplicationswithdynamicuser input handling.