Experiment5

Student Name:Yasar Alam **UID:22BCS15545**
Branch:CSE **Section:22BCS_EPAM-802(B)**
Semester:6th **DOP:26/02/25**
Subject:Project Based Learning with Java **SubjectCode:22CSH-359**

Aim:Develop Java programs using autoboxing,serialization,filehandling,and efficient data processing and management.

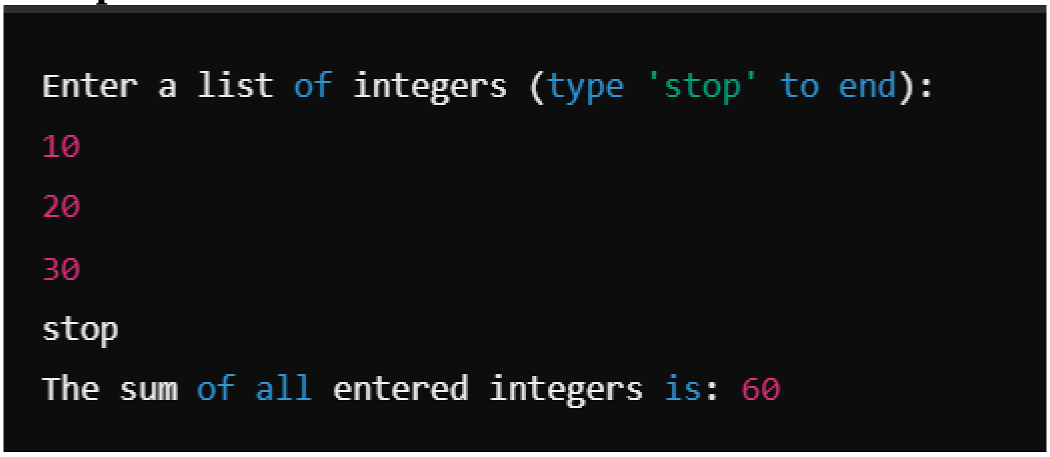
(A) WriteaJavaprogramtocalculatethesumofalistofintegersusingautoboxingand unboxing. Include methods to parse strings into their respective wrapper classes (e.g., Integer.parseInt()).

Objective: The objective ofthisprogramisto:
UnderstandtheconceptofAutoboxingandUnboxingin Java.
ImplementaJavaprogramtocalculatethe sumofalistofintegersusingAutoboxingand Unboxing.
DemonstratetheusageofWrapperClasses(Integer)toconvertStringtoIntegerusing Integer.parseInt().
ShowhowJavaautomaticallyconverts primitivedatatypes(int) towrapperclassobjects (Integer) and vice versa.

Code:

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

publicclassSumOfIntegers{

    public staticvoidmain(String[]args){
        //CreatinganArrayListofInteger(WrapperClass)
        ArrayList<Integer>numbers=newArrayList<>();
        Scanner scanner = new Scanner(System.in);

        //Taking inputfromuser
        System.out.println("Enteralistof integers(type'stop'toend):"); while
        (true) {
            Stringinput=scanner.nextLine();

            // Check if user wants to stop
            inputif(input.equalsIgnoreCase("stop
            ")){
                break;
            }

            //ConvertStringtoIntegerusingparseInt()method try {
                intnumber=Integer.parseInt(input);//Autoboxingoccurshere
                numbers.add(number);
            }catch(NumberFormatExceptione){
                System.out.println("Invalidinput.Pleaseenteravalidinteger.");
            }
        }

        //Calculatethesumofintegers
        int sum = 0;
        for(Integernum:numbers){
            sum+=num;//Unboxingoccurshere
        }

        //Displaytheresult
        System.out.println("Thesumofall enteredintegersis:"+sum);
    }
}
```

Output:



```
Enter a list of integers (type 'stop' to end):
10
20
30
stop
The sum of all entered integers is: 60
```

(B) Create a java program serializer anddeserialize object. The program should: Serialize a Student object (containing id, name, andGPA) andsave it to a file. Deserialize the object fromthefileanddisplaythestudentdetails. HandleFileNotFoundException,IOException, and ClassNotFoundException using exception handling.

Objective:

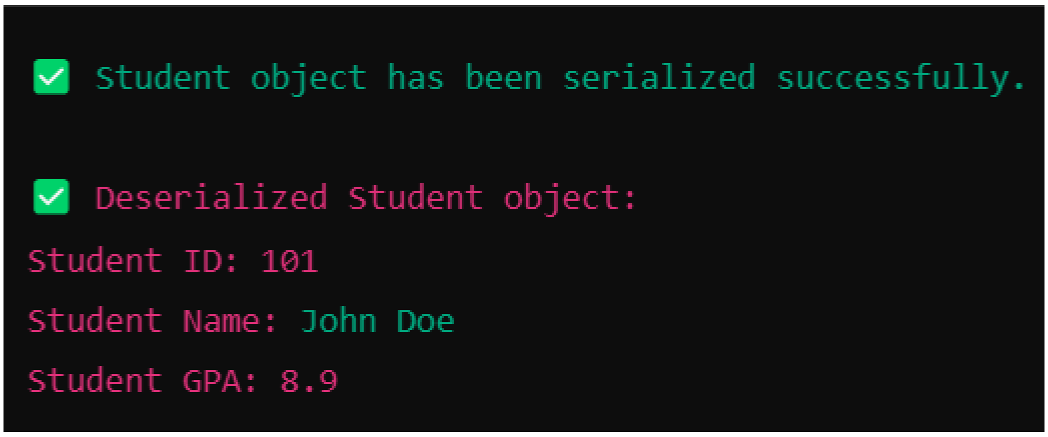
Theobjectiveofthisprogramisto:

1. **UnderstandtheconceptofSerializationandDeserialization**in Java.
2. ImplementaJavaprogramto **SerializeaStudentobject** (with id,name,and GPA)and **saveitto a file**.
3. Implementthefunctionalityto **DeserializetheStudentobject**fromthefileanddisplaythe student details.

Code:

```
importjava.io.*;

//Step1:CreateaSerializableStudentclass
class Student implements Serializable {
    privatestaticfinallongserialVersionUID=1L;//VersionIDforSerialization
    private int id;
    privateStringname;
    private double gpa;

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

    //Methodtodisplaystudentdetails
    public void displayDetails() {
        System.out.println("Student ID: " + id);
        System.out.println("StudentName:"+name);
        System.out.println("Student GPA: " + gpa);
    }
}

publicclassStudentSerialization{

      publicstaticvoidmain(String[]args){
          //Step2:CreateaStudent object
        Studentstudent=newStudent(101,"JohnDoe",8.9); String
        fileName = "student_data.ser";

        //Step3:Serializetheobjectand saveittoafile
        try(ObjectOutputStreamout=newObjectOutputStream(newFileOutputStream(fileName))){
            out.writeObject(student);// Serialize the object
            System.out.println(" Studentobjecthasbeenserializedsuccessfully.");
        }catch(FileNotFoundExceptione){
            System.out.println("+Filenotfound:"+e.getMessage());
        }catch(IOExceptione){
            System.out.println("+IOExceptionoccurred:"+e.getMessage());
        }

        //Step4:Deserializetheobjectfromthefile
        try(ObjectInputStreamin=newObjectInputStream(newFileInputStream(fileName))){ Student
            deserializedStudent = (Student) in.readObject();
            System.out.println("\n DeserializedStudentobject:");
            deserializedStudent.displayDetails();
        }catch(FileNotFoundExceptione){
            System.out.println("+Filenotfound:"+e.getMessage());
        }catch(IOExceptione){
            System.out.println("+IOExceptionoccurred:"+e.getMessage());
        }catch(ClassNotFoundExceptione){
            System.out.println("+ClassNotFoundException:"+e.getMessage());
        }
    }
}
```

Output:



```
✅ Student object has been serialized successfully.

✅ Deserialized Student object:
Student ID: 101
Student Name: John Doe
Student GPA: 8.9
```

(C) Createamenu-basedJavaapplicationwiththe followingoptions.1.AddanEmployee 2. Display All 3. Exit If option 1 is selected, the application should gather details of the employeelikeemployeename,employeeid,designationandsalaryandstoreitinafile.If option 2 is selected, the application should display all the employee details. If option 3 is selected the application should exit.

Objective:

TheobjectiveofthisJavaprogramisto:
Createamenu-drivenapplication inJavausingfilehandlingandserialization. Provide three options:
Option1:AddanEmployee→Gatheremployeedetailslike name,id,designation,and salary and save them to a file.
Option2:DisplayAllEmployees→Readanddisplayallemployeerecordsfromthefile. Option 3: Exit → Close the application.
UseSerializationtostoretheemployeeobjectinafile.

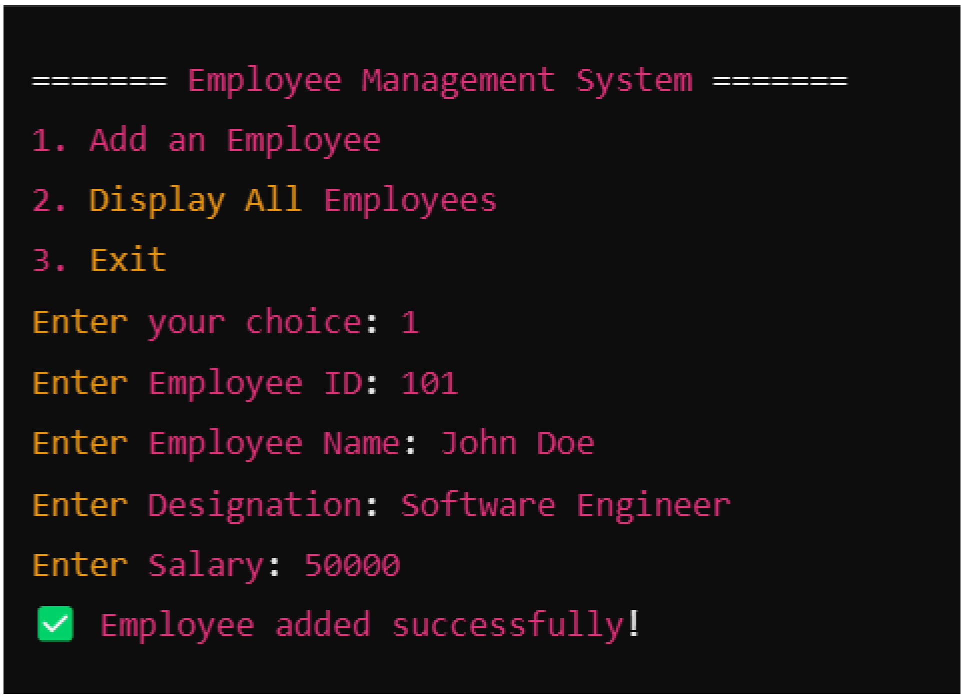
Code:

```
importjava.io.*;

importjava.util.ArrayList;

import java.util.List;

import java.util.Scanner;


//Step1:CreateanEmployeeclassimplementingSerializable class

Employee implements Serializable {

    privatestaticfinallongserialVersionUID=1L;
    private int id;

    privateStringname;

    privateStringdesignation;

    private double salary;


    // Constructor

    publicEmployee(intid,Stringname,Stringdesignation,doublesalary){ this.id =

        id;

        this.name = name;

        this.designation=designation;

        this.salary = salary;

    }


    // Display Employee Details

    publicvoiddisplayEmployee(){

        System.out.println("Employee ID    : " + id);

        System.out.println("Employee Name: " + name);

        System.out.println("Designation : " + designation);

        System.out.println("Salary         : $" + salary);

        System.out.println("----------------------");

    }

}


publicclassEmployeeManagement{


    private static final String FILE_NAME = "employee_data.ser";

    privatestaticList<Employee>employeeList=newArrayList<>();
```

```
public static void main(String[] args) {

    Scannerscanner=newScanner(System.in);


    //Step2:Loadexistingemployees(ifany)

    loadEmployees();


    while(true){

        //Step3:DisplayMenuOptions

        System.out.println("\n=======EmployeeManagementSystem=======");

        System.out.println("1. Add an Employee");

        System.out.println("2.DisplayAllEmployees");

        System.out.println("3. Exit");

        System.out.print("Enter your choice: ");

        intchoice= scanner.nextInt();


        switch(choice){ case

            1:

                addEmployee(scanner);

                break;

            case2:

                displayAllEmployees();

                break;

            case3:

                System.out.println("Exitingtheapplication.Thankyou!"); saveEmployees();

                System.exit(0);

                break;

            default:
                System.out.println("+Invalidchoice.Pleasetryagain.");
        }
    }
}

//Method toAdd anEmployee
publicstaticvoidaddEmployee(Scannerscanner){
    System.out.print("Enter Employee ID: ");
    int id = scanner.nextInt();
    scanner.nextLine();//Consumethenewline

    System.out.print("EnterEmployeeName:"); String
    name = scanner.nextLine();

    System.out.print("EnterDesignation:");
    Stringdesignation=scanner.nextLine();

    System.out.print("Enter Salary: ");
    doublesalary=scanner.nextDouble();

    //Step4:CreateEmployeeObject
    Employeeemp=newEmployee(id,name,designation,salary);
    employeeList.add(emp);
    System.out.println(" Employeeaddedsuccessfully!");
}

// Method to Display All
EmployeespublicstaticvoiddisplayAllEmp
```

```
loyees(){
        if(employeeList.isEmpty()){
            System.out.println("+ No employees found.");
            return;
        }
        System.out.println("\n=======EmployeeDetails=======");
        for (Employee emp : employeeList) {
            emp.displayEmployee();
        }
    }
    //MethodtoSaveEmployeestoFile
    publicstaticvoidsaveEmployees(){
        try(ObjectOutputStreamout=newObjectOutputStream(new
FileOutputStream(FILE_NAME))) {
            out.writeObject(employeeList);
            System.out.println(" Employeedatasavedsuccessfully.");
        }catch(IOExceptione){
            System.out.println("+Errorsavingdata:"+e.getMessage());
        }
    }
    //MethodtoLoadEmployeesfromFile
    public static void loadEmployees() {
        Filefile=newFile(FILE_NAME); if
        (!file.exists()) return;
        try(ObjectInputStreamin=newObjectInputStream(new
FileInputStream(FILE_NAME))) {
            employeeList=(List<Employee>)in.readObject();
        } catch (FileNotFoundException e) {
            System.out.println("+ File not found.");
        }catch(IOExceptione){
          System.out.println("+Errorreadingfile:"+e.getMessage());
        }catch(ClassNotFoundExceptione){
          System.out.println("+Classnotfound:"+e.getMessage());
        }
     }
  }
```

Output:



```
======= Employee Management System =======
1. Add an Employee
2. Display All Employees
3. Exit
Enter your choice: 1
Enter Employee ID: 101
Enter Employee Name: John Doe
Enter Designation: Software Engineer
Enter Salary: 50000
✅ Employee added successfully!
```

LearningOutcomes:

- ThroughtheimplementationofthesethreeJavaprograms,learnerswillgainpractical knowledge in core Java concepts such as autoboxing, unboxing, serialization, file handling, and exception handling.
- They will understand how autoboxing and unboxing facilitate seamless conversion betweenprimitivedatatypesandtheircorrespondingwrapperclasses,enhancingdata processing capabilities.
- Byworkingwithserialization,learnerswillcomprehendhowtoconvertobjectsintoa byte stream and store them in files, enabling persistent storage of objects.
- Additionally,theywillgraspthedeserializationprocesstoretrieveobjectsfromfilesand display their information.
- The implementation of file handling using classes like FileOutputStream, FileInputStream,ObjectOutputStream,andObjectInputStreamwilldeveloptheir understanding of reading from and writing to files.