|  |  |
| --- | --- |
| Name: | Aman kumar |
| UID: | 23BCS11038 |
| Session: | 609-A |

Experiment 2.2 –

Part A –

import java.util.ArrayList;

import java.util.Scanner;

public class SumUsingAutoboxing {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

ArrayList<Integer> numbers = new ArrayList<>();

System.out.println("Enter integers (type 'done' to finish):");

while (true) {

String input = sc.next();

if (input.equalsIgnoreCase("done")) {

break;

}

try {

int num = Integer.parseInt(input);

numbers.add(num);

} catch (NumberFormatException e) {

System.out.println("Invalid input. Please enter an integer.");

}

}

int sum = 0;

for (Integer n : numbers) {

sum += n; // unboxing

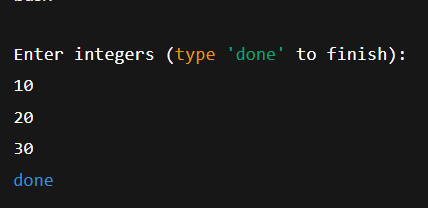
}

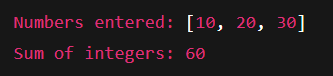
System.out.println("Numbers entered: " + numbers);

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

}

}





PART B –

import java.io.\*;

class Student implements Serializable {

private static final long serialVersionUID = 1L;

int studentID;

String name;

String grade;

public Student(int studentID, String name, String grade) {

this.studentID = studentID;

this.name = name;

this.grade = grade;

}

@Override

public String toString() {

return "Student [ID=" + studentID + ", Name=" + name + ", Grade=" + grade + "]";

}

}

public class StudentSerialization {

public static void main(String[] args) {

String filename = "student.ser";

Try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(filename))) {

Student s1 = new Student(101, "Navya", "A+");

oos.writeObject(s1);

System.out.println("Student object has been serialized: " + s1);

} catch (IOException e) {

e.printStackTrace();

}

try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(filename))) {

Student s2 = (Student) ois.readObject();

System.out.println("Student object has been deserialized: " + s2);

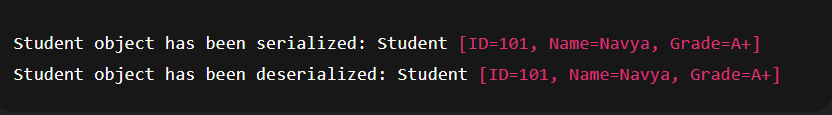
} catch (IOException | ClassNotFoundException e) {

e.printStackTrace();

}

}

}



PART C –

import java.io.\*;

import java.util.\*;

class Employee implements Serializable {

private static final long serialVersionUID = 1L;

int id;

String name;

String designation;

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 {

static final String FILE\_NAME = "employees.dat";

public static void addEmployee(Employee emp) {

try (ObjectOutputStream oos = new ObjectOutputStream(

new FileOutputStream(FILE\_NAME, true)) {

}) {

} catch (IOException e) {

}

try (AppendableObjectOutputStream oos = new AppendableObjectOutputStream(

new FileOutputStream(FILE\_NAME, true))) {

oos.writeObject(emp);

System.out.println("Employee added successfully!");

} catch (IOException e) {

e.printStackTrace();

}

}

public static void displayEmployees() {

try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(FILE\_NAME))) {

System.out.println("\nEmployee Records:");

while (true) {

Employee emp = (Employee) ois.readObject();

System.out.println(emp);

}

} catch (EOFException e) {

System.out.println("End of employee list.");

} catch (IOException | ClassNotFoundException e) {

System.out.println("No records found yet.");

}

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

while (true) {

System.out.println("\n===== Employee Management Menu =====");

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

System.out.println("2. Display All Employees");

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

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

int choice = sc.nextInt();

sc.nextLine();

switch (choice) {

case 1:

System.out.print("Enter Employee ID: ");

int id = sc.nextInt();

sc.nextLine();

System.out.print("Enter Employee Name: ");

String name = sc.nextLine();

System.out.print("Enter Designation: ");

String designation = sc.nextLine();

System.out.print("Enter Salary: ");

double salary = sc.nextDouble();

Employee emp = new Employee(id, name, designation, salary);

addEmployee(emp);

break;

case 2:

displayEmployees();

break;

case 3:

System.out.println("Exiting program...");

sc.close();

return;

default:

System.out.println("Invalid choice! Try again.");

}

}

}

}

class AppendableObjectOutputStream extends ObjectOutputStream {

public AppendableObjectOutputStream(OutputStream out) throws IOException {

super(out);

}

@Override

protected void writeStreamHeader() throws IOException {

reset(); // Prevents writing a new header

}

}

