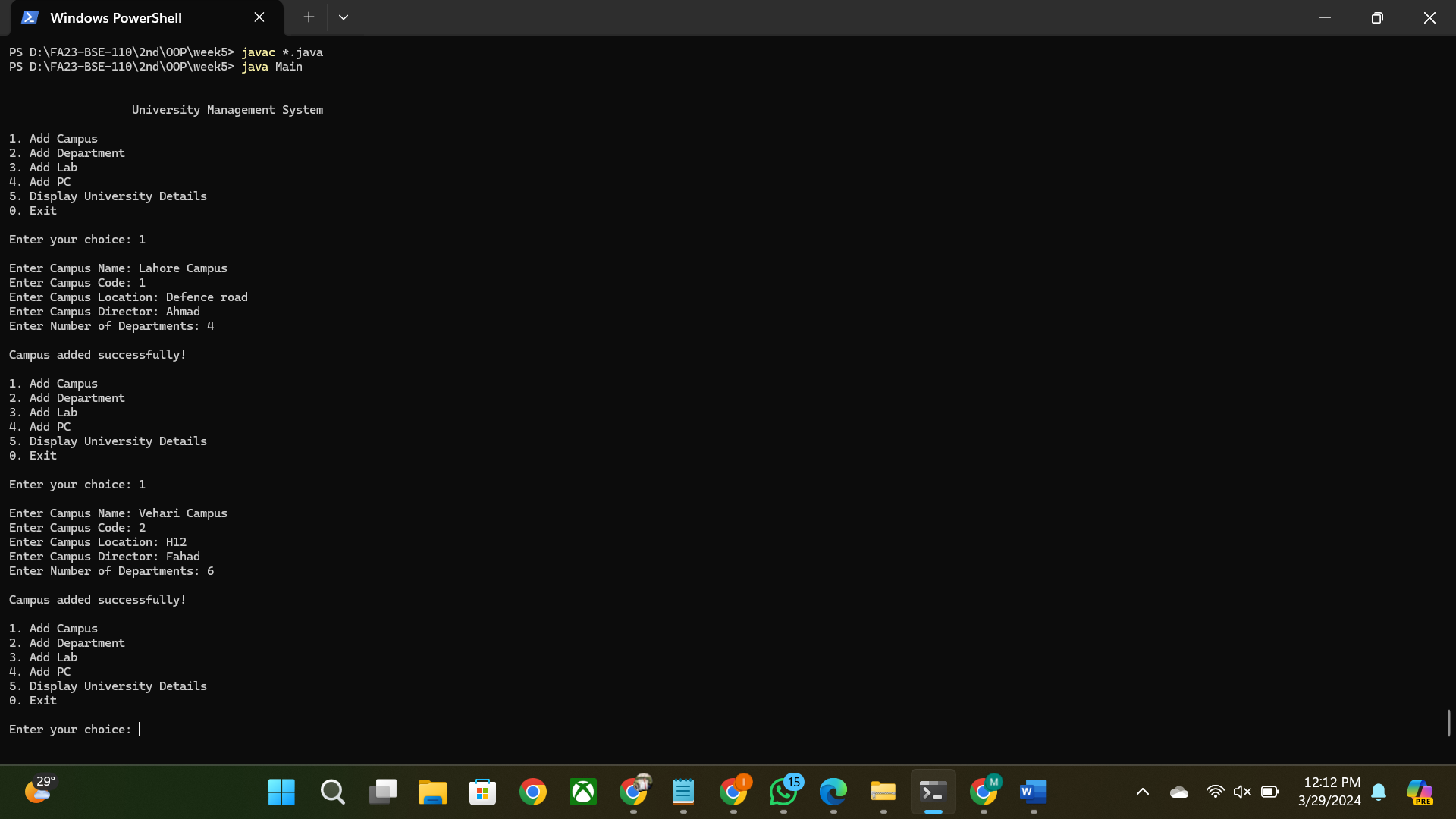
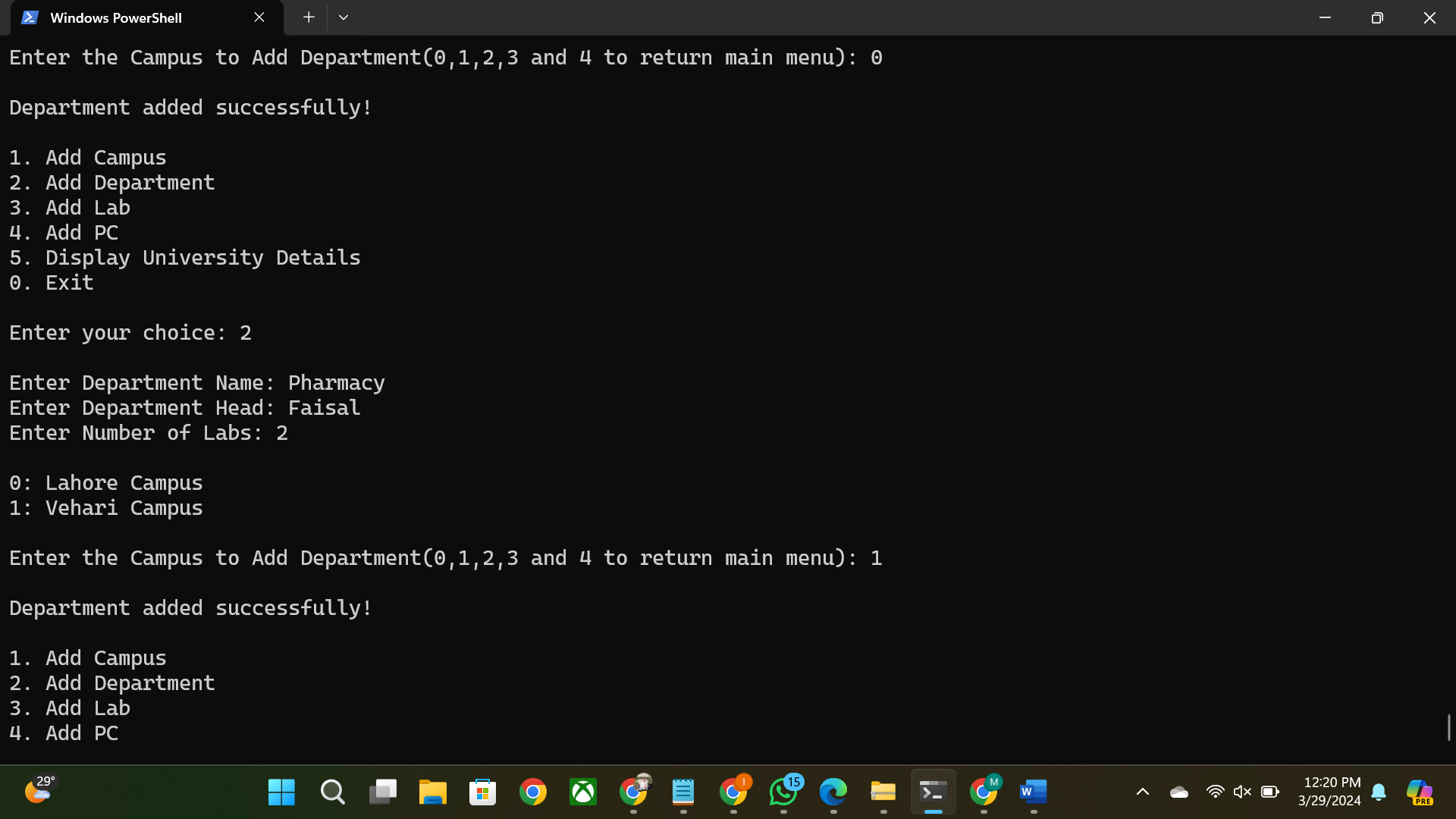
### FIRST I ADDED TWO CAMPUS:



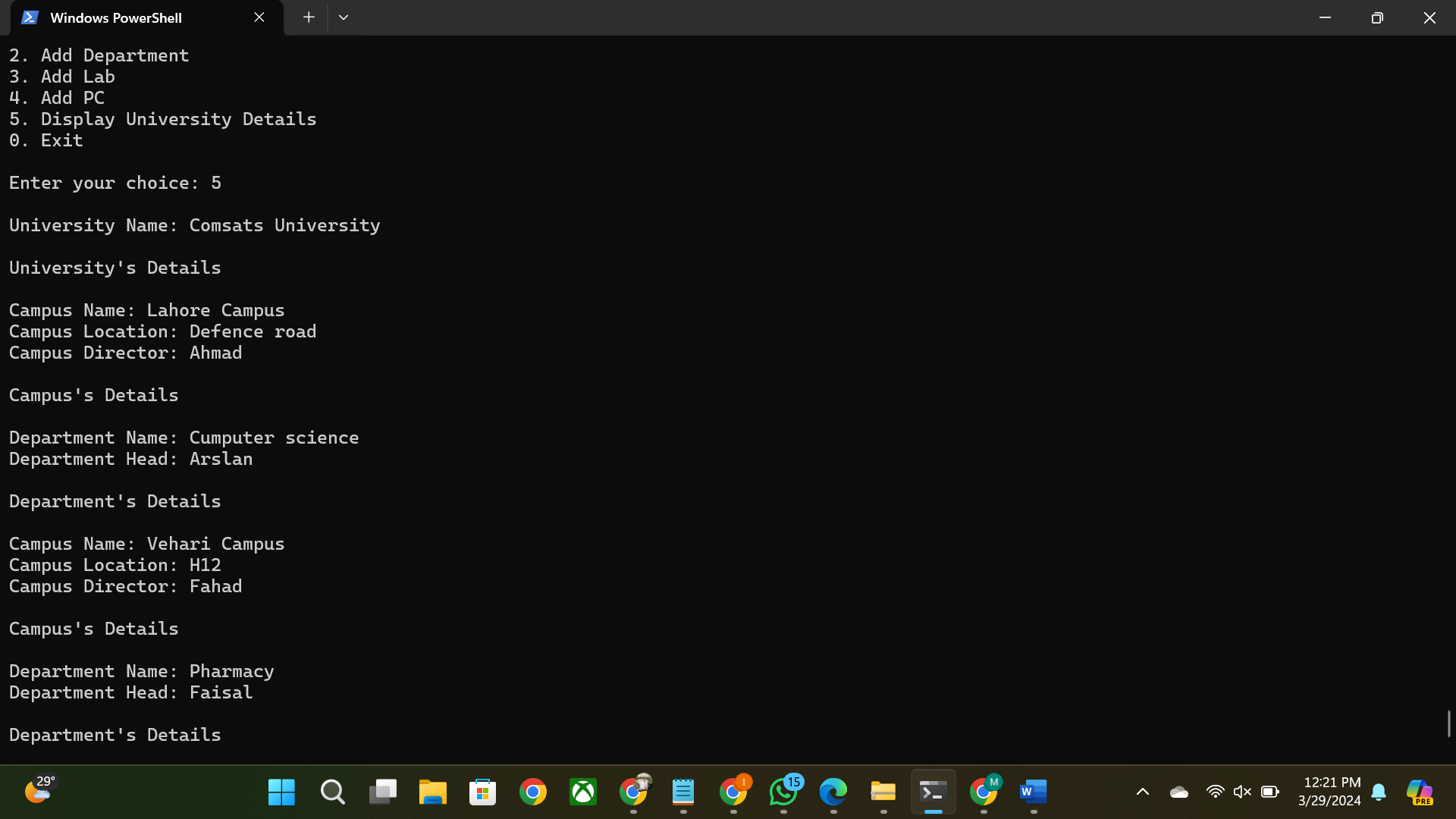
## Then I added department which asked me to select campus to add this department:

## 

## Then added another department by choosing other campus:



## Then displayed university:



## **SOURCE CODE**

public class University{

private String universityName;

private Campus campusList[];

private int counter = 0;

public University(String name, int counter){

this.universityName = name;

this.campusList = new Campus[counter];

}

public String getUniversityName(){

return universityName;

}

public void setUniversityName(String name){

this.universityName = name;

}

public void addNewCampus(Campus campus){

campusList[counter++] = campus;

}

public Campus[] getCampus(){

return campusList;

}

public void displayCampus(){

System.out.println();

for (int i = 0 ; i<campusList.length ; i++){

if (campusList[i] != null) {

System.out.printf("%d: ",i);

System.out.println(this.getCampus()[i].getCampusName());

}

}

}

@Override

public boolean equals(Object o){

University u = (University)o;

return this.universityName.equals(u.universityName) ;

}

@Override

public String toString() {

StringBuilder uniDetails = new StringBuilder("\n\nUniversity's Details\n");

for (int i = 0; i < counter; i++) {

uniDetails.append("\n").append(campusList[i]);

}

return String.format("\nUniversity Name: %s",universityName) + uniDetails;

}

}

class Campus{

private String campusName;

private String campusCode;

private String camLocation;

private String camDirector;

private Department departmentList[];

private int counter = 0;

public Campus(String name , String code , String location , String director , int counter){

this.campusName = name;

this.campusCode = code;

this.camLocation = location;

this.camDirector = director;

this.departmentList = new Department[counter];

}

public void setCampusName(String name){

this.campusName = name;

}

public String getCampusName(){

return campusName;

}

public void setCampusCode(String code){

this.campusCode = code;

}

public String getCampusCode(){

return campusCode;

}

public void setCampusLocation(String location){

this.camLocation = location;

}

public String getCampusLocation(){

return camLocation;

}

public void setCampusDiector(String director){

this.camDirector = director;

}

public String getCampusDirector(){

return camDirector;

}

public void addNewDepartment(Department department){

departmentList[counter++] = department;

}

public Department[] getDepartments(){

return departmentList;

}

/\*public void displayDepartmentName(){

System.out.println();

for (int i = 0 ; i<departmentList.length ; i++){

if (departmentList[i] != null) {

System.out.printf("%d: ",i);

System.out.println(this.getCampus()[i].getCampusName());

}

}

}\*/

@Override

public boolean equals(Object o){

Campus c = (Campus)o;

return this.campusName.equals(c.campusName) ;

}

@Override

public String toString() {

StringBuilder campusDetails = new StringBuilder("\n\nCampus's Details\n");

for (int i = 0; i < counter; i++) {

campusDetails.append("\n").append(departmentList[i]);

}

return String.format("Campus Name: %s\nCampus Location: %s\nCampus Director: %s",campusName, camLocation, camDirector) + campusDetails;

}

}

public class Department {

private String dName;

private String dHead;

private PcLab pcLabs[];

private int counter = 0;

public Department(String dName, String dHead, int labsCount) {

this.dName = dName;

this.dHead = dHead;

this.pcLabs = new PcLab[labsCount];

}

public void setDName(String dName) {

this.dName = dName;

}

public void setDHead(String dHead) {

this.dHead = dHead;

}

public String getDName() {

return dName;

}

public String getDHead() {

return dHead;

}

public void addNewLab(PcLab lab) {

pcLabs[counter++] = lab;

}

public PcLab[] getLabs() {

return pcLabs;

}

@Override

public boolean equals(Object o){

Department d = (Department)o;

return this.dName.equals(d.dName) ;

}

@Override

public String toString() {

StringBuilder departmentDetails = new StringBuilder("\n\nDepartment's Details\n");

for (int i = 0; i < counter; i++) {

departmentDetails.append("\n").append(pcLabs[i]);

}

return String.format("Department Name: %s\nDepartment Head: %s", dName, dHead) + departmentDetails;

}

}

class PcLab {

private String labName;

private Pc[] pcList;

private String labAssistant;

private String labInCharge;

private int pcCount = 0;

PcLab(String labName, int pcCount, String labAssistant, String labInCharge) {

this.labName = labName;

this.pcList = new Pc[pcCount];

this.labAssistant = labAssistant;

this.labInCharge = labInCharge;

}

// Getters and Setters

public String getLabName() {

return labName;

}

public void setLabName(String labName) {

this.labName = labName;

}

public String getLabAssistant() {

return labAssistant;

}

public void setLabAssistant(String labAssistant) {

this.labAssistant = labAssistant;

}

public String getLabInCharge() {

return labInCharge;

}

public void setLabInCharge(String labInCharge) {

this.labInCharge = labInCharge;

}

public void addPc(int id, String cpu, String ram, String storage, String lcdMaker, boolean hasGpu) {

if (pcCount < pcList.length) {

pcList[pcCount++] = new Pc(id, cpu, ram, storage, lcdMaker, hasGpu);

}

}

@Override

public String toString() {

StringBuilder pcDetails = new StringBuilder("\n\nPC's Details\n");

for (int i = 0; i < pcCount; i++) {

pcDetails.append("\n").append(pcList[i]);

}

return String.format("Lab Name: %s\nLab InCharge: %s\nLab Assistant: %s", labName, labInCharge, labAssistant) + pcDetails;

}

public Pc[] getPcList(){

return pcList;

}

public void setPcList(Pc newList[]){

pcList=newList;

}

@Override

public boolean equals(Object o){

PcLab lab = (PcLab)o;

return this.labName.equals(lab.labName);

}

}

class Pc {

private final int id;

private String cpu;

private String ram;

private String storage;

private String lcdMaker;

private boolean hasGpu;

Pc(int id, String cpu, String ram, String storage, String lcdMaker, boolean hasGpu) {

this.id = id;

this.cpu = cpu;

this.ram = ram;

this.storage = storage;

this.lcdMaker = lcdMaker;

this.hasGpu = hasGpu;

}

// Getters and Setters

public int getId() {

return id;

}

public String getCpu() {

return cpu;

}

public void setCpu(String cpu) {

this.cpu = cpu;

}

public String getRam() {

return ram;

}

public void setRam(String ram) {

this.ram = ram;

}

public String getStorage() {

return storage;

}

public void setStorage(String storage) {

this.storage = storage;

}

public String getLcdMaker() {

return lcdMaker;

}

public void setLcdMaker(String lcdMaker) {

this.lcdMaker = lcdMaker;

}

public boolean hasGpu() {

return hasGpu;

}

public void setGpu(boolean hasGpu) {

this.hasGpu = hasGpu;

}

@Override

public String toString() {

return String.format("Id: %d\nCPU: %s\nRAM: %s\nStorage: %s\nLCD Maker: %s\nGPU: %b\n", id, cpu, ram, storage, lcdMaker, hasGpu);

}

@Override

public boolean equals(Object o){

Pc p = (Pc)o;

return this.id == p.id && this.cpu.equals(p.cpu);

}

}

import java.util.Scanner;

public class Menu{

    University myUniversity = new University("Comsats University", 4);

    Scanner input = new Scanner(System.in);

    public void start(){

        int choice;

        System.out.println("\n\n                  University Management System");

        do {

            System.out.println("\n1. Add Campus");

            System.out.println("2. Add Department");

            System.out.println("3. Add Lab");

        System.out.println("4. Add PC");

            System.out.println("5. Display University Details");

            System.out.println("0. Exit");

            System.out.print("\nEnter your choice: ");

        choice = input.nextInt();

        input.nextLine();

            switch(choice){

                case 1:

                    addCampus();

                    break;

                case 2:

                    addDepartment();

                    break;

                case 3:

                    //addLab();

                    break;

                case 4:

                    //addPc();

                    break;

                case 5:

                    System.out.println(myUniversity);

                    break;

                case 0:

                    System.out.println("\nExiting program...\n");

                                break;

                        default:

                                System.out.println("Invalid choice. Please try again.");

                    break;

            }

        } while(choice != 0);

    }

    private void addCampus() {

            System.out.print("\nEnter Campus Name: ");

            String name = input.nextLine();

            System.out.print("Enter Campus Code: ");

            String code = input.nextLine();

            System.out.print("Enter Campus Location: ");

            String location = input.nextLine();

            System.out.print("Enter Campus Director: ");

            String director = input.nextLine();

            System.out.print("Enter Number of Departments: ");

            int numDepartments = input.nextInt();

            Campus campus = new Campus(name, code, location, director, numDepartments);

            myUniversity.addNewCampus(campus);

            System.out.println("\nCampus added successfully!");

    }

    private void addDepartment(){

        int camChoice;

            System.out.print("\nEnter Department Name: ");

                String dName = input.nextLine();

                System.out.print("Enter Department Head: ");

                String dHead = input.nextLine();

                System.out.print("Enter Number of Labs: ");

                int labsCount = input.nextInt();

            myUniversity.displayCampus();

            System.out.print("\nEnter the Campus to Add Department(0,1,2,3 and 4 to return main menu): ");

            camChoice = input.nextInt();

            input.nextLine();

            switch(camChoice){

                case 0:

                    myUniversity.getCampus()[0].addNewDepartment(new Department(dName, dHead, labsCount));

                    break;

                case 1:

                    myUniversity.getCampus()[1].addNewDepartment(new Department(dName, dHead, labsCount));

                    break;

                case 2:

                    myUniversity.getCampus()[2].addNewDepartment(new Department(dName, dHead, labsCount));

                    break;

                case 3:

                    myUniversity.getCampus()[3].addNewDepartment(new Department(dName, dHead, labsCount));

                    break;

                case 4:

                    System.out.print("\nReturning to Main Menu");

                    break;

                        default:

                                System.out.println("Invalid choice. Please try again.");

                    break;

            }

                    System.out.println("\nDepartment added successfully!");

    }

    /\*private void addLab(){

        int camChoice;

        int depChoice;

            System.out.print("\nEnter Lab Name: ");

                String lName = input.nextLine();

                System.out.print("Enter Lab Incharge Name: ");

                String InCharge = input.nextLine();

                System.out.print("Enter Lab Assistant Name: ");

                String lAssistant = input.nextLine();

                System.out.print("Enter Number of PCs: ");

                int pcCount = input.nextInt();

            myUniversity.displayCampus();

            System.out.print("\nEnter the Campus to Add Department(0,1,2,3 and 4 to return main menu): ");

            camChoice = input.nextInt();

            input.nextLine();

            switch(camChoice){

                case 0:{

                    myUniversity.getCampus()[0].addNewDepartment(new Department(dName, dHead, labsCount));

                    break;

                }

                case 1:

                    myUniversity.getCampus()[1].addNewDepartment(new Department(dName, dHead, labsCount));

                    break;

                case 2:

                    myUniversity.getCampus()[2].addNewDepartment(new Department(dName, dHead, labsCount));

                    break;

                case 3:

                    myUniversity.getCampus()[3].addNewDepartment(new Department(dName, dHead, labsCount));

                    break;

                case 4:

                    System.out.print("\nReturning to Main Menu");

                    break;

                        default:

                                System.out.println("Invalid choice. Please try again.");

                    break;

            }

                    System.out.println("\nDepartment added successfully!");

    }\*/

}

public class Main {

public static void main(String[] args) {

Menu m = new Menu();

m.start();

//PcLab pcLab1 = new PcLab("OOP Lab", 10, "Ali Raza", "Muhammad Danish");

//pcLab1.addPc(123, "Core i7", "16 GB", "512 SSD", "HP", true);

//pcLab1.addPc(124, "Core i7", "16 GB", "512 SSD", "HP", true);

// System.out.println(pcLab);

//cs.getLabs()[0].getPcList()[0]=new Pc(124, "Core i7", "16 GB", "512 SSD", "HP", true);

//System.out.println(cs.getLabs()[0].getPcList()[0]);

//System.out.println(cs);

/\* PcLab lab2 = new PcLab("PF Lab", 10, "Ali Ahmad", "Talha Iqbal");

lab2.addPc(124, "Core i7", "16 GB", "512 SSD", "HP", true);

Department cs = new Department("Computer Science", "Ali Attah", 10);

cs.addNewLab(lab2);

Department it = new Department("Information Technology", "Fazal", 8);

it.addNewLab(pcLab1);

Campus mainCampus = new Campus("Main Campus","123","Defence road Lahore", "Saad", 2);

mainCampus.addNewDepartment(cs);

mainCampus.addNewDepartment(it);

University myUniversity = new University("Comsats University", 1);

myUniversity.addNewCampus(mainCampus);

//System.out.println(myUniversity);

myUniversity.getCampus()[0].getDepartments()[0].getLabs()[0] = new PcLab("Physics Lab", 10, "faizan", " Danish");

System.out.println(myUniversity.getCampus()[0].getDepartments()[0].getLabs()[0]); \*/

}

}