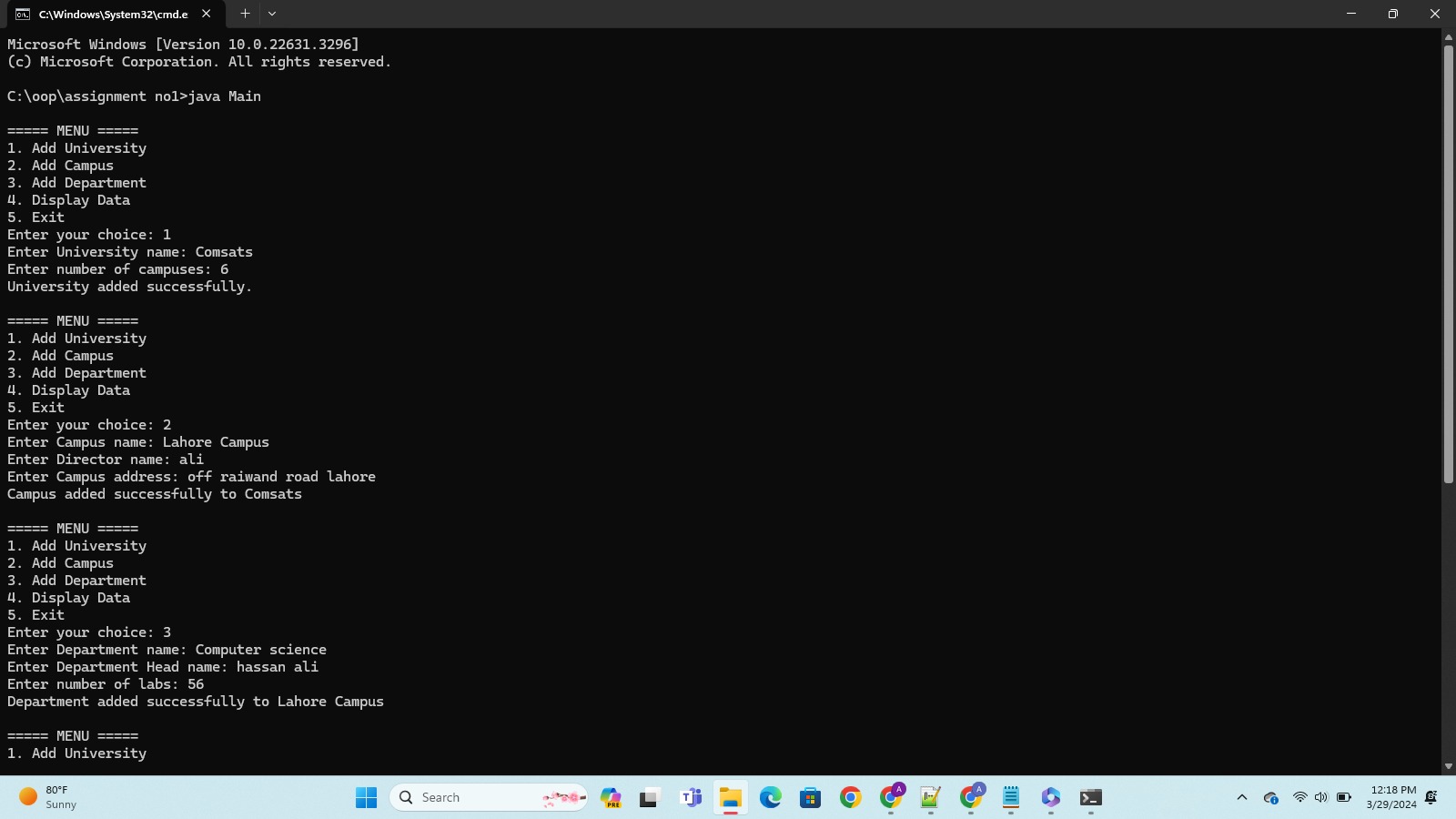
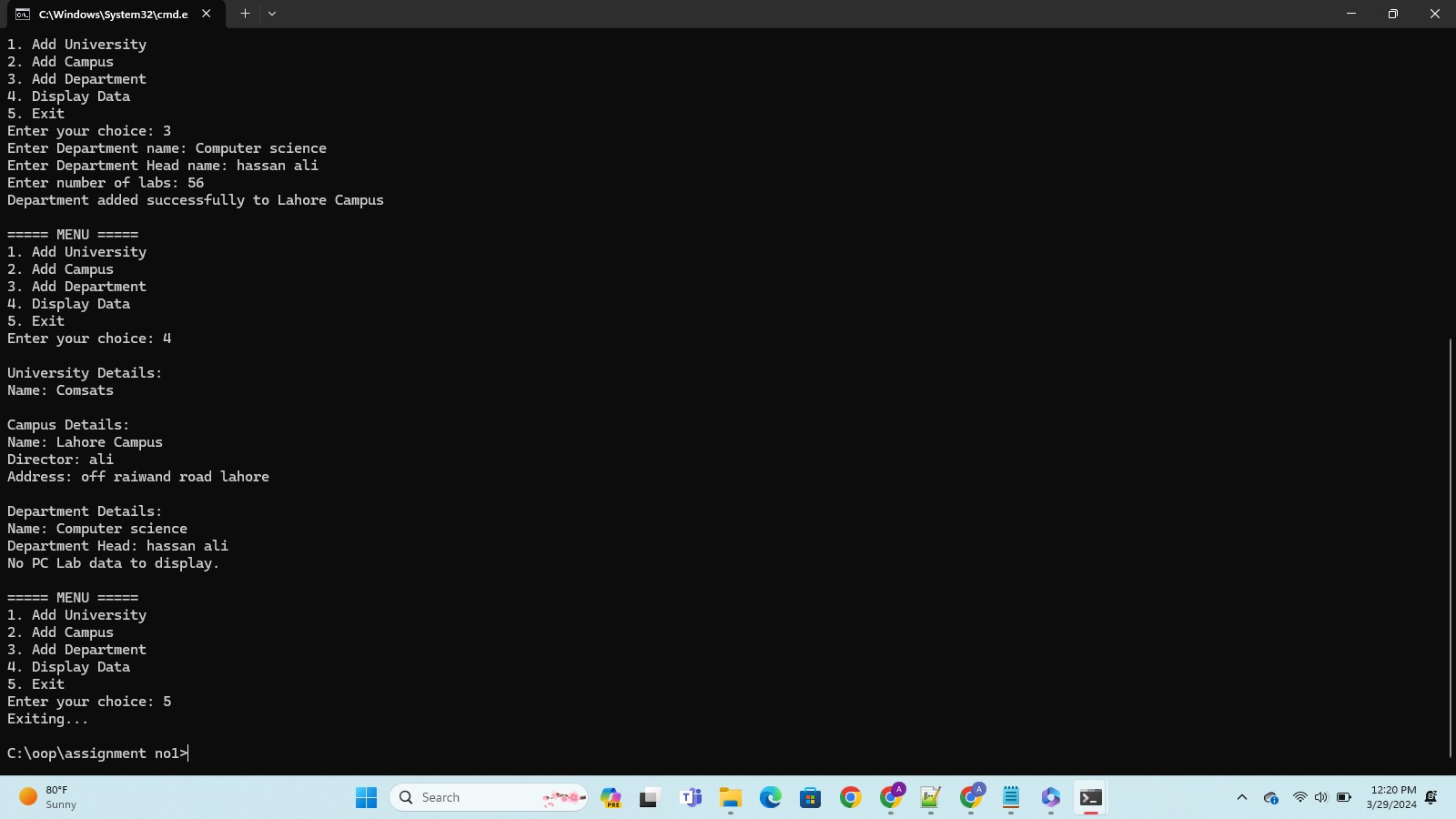
**CODE OUTPUT:**





**CAMPUS:**

public class Campus {

private String name;

private String directorName; // Corrected variable name

private String address;

private Department[] departments;

private int count = 0;

public Campus(String name, int departmentCount, String directorName, String address) {

this.directorName = directorName;

this.address = address;

this.name = name;

this.departments = new Department[departmentCount];

}

public void addNewDepartment(Department department) { // Corrected method name

departments[count++] = department;

}

public String getName() {

return name;

}

public String getDirectorName() { // Corrected method name

return directorName;

}

public String getAddress() {

return address;

}

public Department[] getDepartments() { // Corrected method name

return departments;

}

@Override

public String toString() {

StringBuilder departmentDetails = new StringBuilder("\n\n Department Details\n");

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

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

}

return String.format("Campus Name: %s\nDirector Name: %s\nAddress: %s ", name, directorName, address) + departmentDetails;

}

@Override

public Object clone(){

return new Campus(this.name,this.count,this.directorName ,this.address);

}

@Override

public boolean equals(Object o)

{

Campus p=(Campus)o;

return this.name.equals(p.name)&&this.address.equals(p.address);

}

}

**DEPARTMENT**

class Department {

private String dName;

private String dHead;

private PcLab[] pcLabs;

private int counter = 0;

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

this.dName = dName;

this.dHead = dHead;

this.pcLabs = new PcLab[labsCount];

}

public void addNewLab(PcLab lab) {

pcLabs[counter++] = lab;

}

public String getDName() {

return dName;

}

public String getDHead() {

return dHead;

}

@Override

public String toString() {

StringBuilder labDetails = new StringBuilder("\n\nPcLab details\n");

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

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

}

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

}

public PcLab[] getLabs() {

return pcLabs;

}

@Override

public Object clone(){

return new Department(this.dName,this.dHead,this.counter);

}

@Override

public boolean equals(Object o)

{

Department p=(Department)o;

return this.dName.equals(p.dName)&&this.dHead.equals(p.dHead);

}

public PcLab[] getpcLabs(){

return pcLabs;}

}

**UNI**

public class Uni {

private String name;

private Campus camp[];

private int count;

public Uni(String name, int count) {

this.name=name;

this.camp=new Campus[count];

}

public String getName() {

return name;

}

public void addNewCam(Campus c) {

camp[count++]=c;

}

public Campus[] getCampuses() {

return camp;

}

@Override

public String toString(){

StringBuilder campdetails = new StringBuilder("\n\n Campus Details\n");

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

campdetails.append("\n").append(camp[i]);

}

return String.format("University name: %s\n",name) + campdetails;

}

@Override

public Object clone(){

return new Uni(name,count);

}

@Override

public boolean equals(Object o)

{

Uni p=(Uni)o;

return this.name.equals(p.name);

}

}

**PCLAB:**

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 Object clone(){

return new PcLab(this.labName,this.pcCount,this.labAssistant ,this.labInCharge);

}

@Override

public boolean equals(Object o)

{

PcLab p=(PcLab)o;

return this.labAssistant.equals(p.labAssistant)&&this.labInCharge.equals(p.labInCharge);

}

}

**PC:**

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);

}

public Object clone(){

return new Pc(this.id ,this.ram,this.lcdMaker,this.storage,this.cpu,this.hasGpu);

}

public boolean equals(Object o)

{

Pc p=(Pc)o;

return this.cpu.equals(p.cpu);

}

**MENU:**

import java.util.Scanner;

public class Menu {

private Scanner scanner = new Scanner(System.in);

private Uni university;

public void startMenu() {

int choice;

do {

System.out.println("\n MENU ");

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

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

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

System.out.println("4. Display Data");

System.out.println("5. Exit");

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

choice = scanner.nextInt();

scanner.nextLine(); // Consume newline

switch (choice) {

case 1:

addUniversity();

break;

case 2:

addCampus();

break;

case 3:

addDepartment();

break;

case 4:

displayData();

break;

case 5:

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

break;

default:

System.out.println("Invalid choice. Please enter a valid option.");

}

} while (choice != 5);

scanner.close();

}

public void addUniversity() {

System.out.print("Enter University name: ");

String name = scanner.nextLine();

System.out.print("Enter number of campuses: ");

int numCampuses = scanner.nextInt();

scanner.nextLine(); // Consume newline

university = new Uni(name, numCampuses);

System.out.println("University added successfully.");

}

public void addCampus() {

if (university == null) {

System.out.println("Please add a university first.");

return;

}

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

String campusName = scanner.nextLine();

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

String directorName = scanner.nextLine();

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

String address = scanner.nextLine();

Campus campus = new Campus(campusName, 1, directorName, address);

university.addNewCam(campus);

System.out.println("Campus added successfully to " + university.getName());

}

public void addDepartment() {

if (university == null) {

System.out.println("Please add a university first.");

return;

}

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

String deptName = scanner.nextLine();

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

String deptHead = scanner.nextLine();

System.out.print("Enter number of labs: ");

int numLabs = scanner.nextInt();

scanner.nextLine(); // Consume newline

Department department = new Department(deptName, deptHead, numLabs);

Campus[] campuses = university.getCampuses();

if (campuses != null && campuses.length > 0 && campuses[0] != null) {

campuses[0].addNewDepartment(department);

System.out.println("Department added successfully to " + campuses[0].getName());

} else {

System.out.println("No campus found to add department.");

}

}

public void displayData() {

if (university == null) {

System.out.println("No University data to display.");

return;

}

System.out.println("\nUniversity Details:");

System.out.println("Name: " + university.getName());

Campus[] campuses = university.getCampuses();

for (Campus campus : campuses) {

if (campus == null) {

System.out.println("No Campus data to display.");

return;

}

System.out.println("\nCampus Details:");

System.out.println("Name: " + campus.getName());

System.out.println("Director: " + campus.getDirectorName());

System.out.println("Address: " + campus.getAddress());

Department[] departments = campus.getDepartments();

for (Department department : departments) {

if (department == null) {

System.out.println("No Department data to display.");

return;

}

System.out.println("\nDepartment Details:");

System.out.println("Name: " + department.getDName());

System.out.println("Department Head: " + department.getDHead());

PcLab[] pcLabs = department.getpcLabs();

for (PcLab pcLab : pcLabs) {

if (pcLab == null) {

System.out.println("No PC Lab data to display.");

return;

}

System.out.println("\nPC Lab Details:");

System.out.println("Lab Name: " + pcLab.getLabName());

System.out.println("Lab Assistant: " + pcLab.getLabAssistant());

System.out.println("Lab InCharge: " + pcLab.getLabInCharge());

Pc[] pcs = pcLab.getPcList();

for (Pc pc : pcs) {

if (pc == null) {

System.out.println("No PC data to display.");

return;

}

System.out.println("\nPC Details:");

System.out.println(pc);

}

}

}

}

}

}

**MAIN:**

public class Main {

public static void main(String[] args) {

Menu menu = new Menu();

menu.startMenu();

}

}

**}**