Name: Muhammad Mudassar Raza	**Class :** ADP-CS	**Section :** M
Teacher : Sir Amjad Khan	**Assignment no 1**	

Resubmission Of Assignment

1.Overview:

The Java-based University Management System simulates a basic academic setup that includes universities, courses, lecturers, students and classes that are given modules. The system displays object-oriented programming (OOP) such as inheritance, capsules, abstraction , polymorphism.

2. Class and Working:

2.1: Person Class:

-**Purpose:** Represents an abstract person with general attributes.
-**Inherited:** Expanded by "students" and lecturers.
The code for Person class is:

```
public class Person{
private String name;
private int age;
private String gender;
private Address address;

public Person(String pName, int pAge, String pGender){
name = pName;
age = pAge;
gender = pGender;
}
```

```
public String getname(){
return name;
}
public void setname(String pName){
name = pName;
}
public int getage(){
return age;
}
public void setage(int pAge){
age = pAge;
}
public String getgender(){
return gender;
}
public void setgender(String pGender){
gender = pGender;
}
public Address getaddress(){
return address;
}
public void setaddress(Address addr){
address = addr;
}
}
```

2.2 Lecturer Class:

- **Purpose:** Represents a university lecturer.
- **Attributes:**
- `employeenumber`(string)
- `nlnumber`(string) National License Number.
- `jobtitle`(string)
- `salary`(double)

The code of Lecturer class is :

```
public class Lecturer extends Person{
private String employeeNumber;
private String Nlnumber;
private String jobTitle;
private double salary;

public Lecturer(String pName, String pGender, int pAge, String empNum, String nl, String job, double sal){
super(pName, pAge, pGender);
employeeNumber = empNum;
Nlnumber = nl;
jobTitle = job;
salary = sal;
}
```

```
public String getemployeeNumber(){
return employeeNumber;
}
public void setemployeeNumber(String empNum){
employeeNumber = empNum;
}
public String getNlnumber(){
return Nlnumber;
}
public void setNlnumber(String nl){
Nlnumber = nl;
}
public String getjobTitle(){
return jobTitle;
}
public void setjobTitle(String job){
jobTitle = job;
}
public double getsalary(){
return salary;
}
public void setsalary(double sal){
salary = sal;
}
}
```

2.3 Students (Extended Class):

- **Purpose:** Represents a student with academic details.
- **Marketing** (boolean) Marketing Registration-email.

The code for this class is:

```
public class Student extends Person{
private String studentNumber;
private boolean marketing;

public Student(String pName, String pGender, int pAge, String studNum, boolean market){
super(pName, pAge, pGender);
studentNumber = studNum;
marketing = market;
}
public String getstudentNumber(){
return studentNumber;
}
public void setstudentNumber(String studNum){
studentNumber = studNum;
}
public boolean getmarketing(){
return marketing;
}
public void setmarketing(boolean market){
marketing = market;
}
}
```

2.4 : Address Class:

- **Purpose**: Saves address details for one person or institution.
- Attribute:
- `streetnumber`(int)
- `cityorcountry`(string)
- `addresslines`(string)
- `postcode`(string)
- `country`(string)

Code for Address class is :

```
public class Address{
    private int streetNumber;
    private String cityorcountry;
    private String addresslines;
    private String postcode;
    private String country;

    public Address(int strNumber, String city, String addrLines, String pcode, String ctry){
        streetNumber = strNumber;
        cityorcountry = city;
        addresslines = addrLines;
        postcode = pcode;
        country = ctry;
}
```

```
public int getstreetNumber(){
return streetNumber;
}
public void setstreetNumber(int strNumber){
streetNumber = strNumber;
}
public String getcityorcountry(){
return cityorcountry;
}
public void setcityorcountry(String city){
cityorcountry = city;
}
public String getaddresslines(){
return addresslines;
}
public void setaddresslines(String addrLines){
addresslines = addrLines;
}
public String getpostcode(){
return postcode;
}
public void setpostcode(String pcode){
postcode = pcode;
}
public String getcountry(){
return country;
}
public void setcountry(String ctry){
country = ctry;
}
}
```

2.5 : Course Class:

- Represents an academic course in a university system
- Stores course details: name, department, duration, lecturer, and student list
- Implements getter/setter methods for all class attributes
- Follows OOP encapsulation principles

The code for it is:

```
public class Course{
private String coursename;
private String department;
private int durationInYear;
private Lecturer lecturer;
private Student[] students;
private Course[] course;

public Course(String cName, String dept, int duration, Lecturer lect, Student[] studs, Course[] courses){
coursename = cName;
department = dept;
durationInYear = duration;
lecturer = lect;
students = studs;
course = courses;
}
```

```java
public String getcoursename(){
return coursename;
}
public void setcoursename(String cName){
coursename = cName;
}
public String getdepartment(){
return department;
}
public void setdepartment(String dept){
department = dept;
}
public int getdurationInYear(){
return durationInYear;
}
public void setdurationInYear(int duration){
durationInYear = duration;
}
public Lecturer getlecturer(){
return lecturer;
}
public void lecturer(Lecturer lect){
lecturer = lect;
}
```

```java
public Student[] getstudents(){
return students;
}
public void setstudent(Student[] studs){
students = studs;
}
public Course[] getcourse(){
return course;
}
public void setcourse(Course[] courses){
course = courses;
}
}
```

2.6 University Class:

- **Purpose**: A class that represents the university that has the location.
- **Important Way:**
- Set up with getter for university details. 2.6.
- **Attribute:**
- `coursename`(string)
- 'department`(string)
- ` durationinyear`(int)
- 'instructor (instructor) - Course instructor.
- "Student" (student[]) Registered students.
- "course" (course []) Potential similar course (not used in existing implementations)

The code for it is:

```java
public class University{
private String uniname;
private Address address;
public University(String uniName, Address addr){
uniname = uniName;
address = addr;
}
public String getuniname(){
return uniname;
}
public void setuniname(String uniName){
uniname = uniName;
}
public Address getaddress(){
return address;
}
public void setaddress(Address addr){
address = addr;
}
}
```

2.7 Module Class:

Purpose: Represents an individual subject module in a course.

- **Attributes:**
- `modulename '(string)
- 'lescevenue`(string)
- `dayofweek`(string)
- `time`(string)
- `Lecturer (Lecturer) - A person who teaches modules.

The code for it is:

```java
public class Module{
private String modulename;
private String lectureVenue;
private String dayOfWeek;
private String time;
private Lecturer lecturer;

public Module(String mName, String venue, String day, String mTime, Lecturer lect){
modulename = mName;
lectureVenue = venue;
dayOfWeek = day;
time = mTime;
lecturer = lect;
}
```

```
public String getmodulename(){
return modulename;
}
public void setmodulename(String mName){
modulename = mName;
}
public String getlectureVenue(){
return lectureVenue;
}
public void setlectureVenue(String venue){
lectureVenue = venue;
}
public String getdayOfWeek(){
return dayOfWeek;
}
public void setdayOfWeek(String day){
dayOfWeek = day;
}
public String gettime(){
return time;
}
public void settime(String mTime){
time = mTime;
}
public Lecturer getlecturer(){
return lecturer;
}
public void setlecturer(Lecturer lect){
lecturer = lect;
}
}
```

2.8 The Main Class:

1. **Initialization:**

- Create a university "address".
- Create an instance of "University" ("Minhaji University"). (Programming Java").
- Create a course Problem:
- Print university information (`name`, `location`).
- Printing course information (`name`, `lecturers').
- Print registered students.
- Print module information (`name`, `day`, `time`).

The code for it is:

```
public class Main{
    public static void main(String[] args){
        Address uniaddress = new Address(1, "Lahore", "Township", "54000", "Pakistan");

        University myuni = new University("Minhaj Univeristy", uniaddress);
        Lecturer teacher = new Lecturer("Hasham Haider", "Male", 42, "E111", "Nl100", "Professor", 80000);
        Student student1 = new Student("Mudassir", "Male", 22, "001", false);
        Student student2 = new Student("Hamid", "Male", 22, "003", false);
```

```
Student[] studentlist = {student1, student2};
Module javamodule = new Module("Java programming", "Room no 6402", "Tuesday", "10.30 Am", teacher);
Course[] courselist = new Course[1];
Course CScourse = new Course("Mobile app development", "Computer Science", 4, teacher, studentlist, courselist);
courselist[0] = CScourse;
```

```
System.out.println("University: " + myuni.getuniname());
System.out.println("Location: " + myuni.getaddress().getcityorcountry());
System.out.println("\nCourse: " + CScourse.getcoursename());
System.out.println("Lecturer: " + CScourse.getlecturer().getname());
System.out.println("Students: ");
```

```
        for (Student s : studentlist){
            System.out.println("- " + s.getname() + " (" + s.getstudentNumber() + ")");
        }
        System.out.println("Module: " + javamodule.getmodulename());
        System.out.println("Day: " + javamodule.getdayOfWeek() + ", Time: " + javamodule.gettime());
    }
}
```

3. Conclusion:

This system simulates a simple university organization with encapsulation of classes, relationships and data. This can be further developed to include a registration system, classification, and complex course management.