Name: Muhammad Mudassar Raza	**Class:** ADP-CS	**Section:** (M)	**Subject:** Mobile Application dev
Teacher name: Sir Amjad Khan	Ass#1		

TOPIC : University Management System Documentation

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, and configuration.

Class 2 Description

2.1. `person` (base class)

-**Purpose:** Represents an abstract person with general attributes.

-**Inherited:** Expanded by "students" and lecturers.

2.2. address

- **Purpose**: Saves address details for one person or institution.
- Attribute:
- `streetnumber`(int)
- `cityorcountry`(string)
- `addresslines`(string)
- `postcode`(string)
- `country`(string)
- **Key Method**:
- Initialization.
- Change getter and setter.

2.3. Student (Extended Person)

- **Purpose:** Represents a student with academic details.
- **Marketing** (boolean) Marketing Registration-email.
- Important Method:
- Inherit the "people" method.
- Student-specific field getters and setters.

2.4. "Lecturer" (Expanded "People")

- **Purpose:** Represents a university lecturer.
- **Additional Attributes:**
- `equipheenumber`(string)
- `nlnumber`(string) National License Number.
- `jobtitle`(string)
- `salary`(double)
- **Key method:**
- Inheritance method.
- Getters and setters in the instructor field.

2.5. "University"
- **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).
- **Important Way:**
- Setzer and getter for course info.

2.7. `module`
- **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.
- **Important Way:**
- Getters and Setters for Modules.

Third main program workflow (`main.java`)
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`).

4. Conceptual illustration of ooop key
Inheritance:
- "student" and "lector" heirs are from "people."
Capsulation::
- All fields are "private" with "getters/setters".
Configuration:
- "University" includes "address".
- "course" includes "Lecturer" and "Student[]".

6. **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. Note: This document describes the architecture and logic of the system. See the specified Java file for the complete implementation.