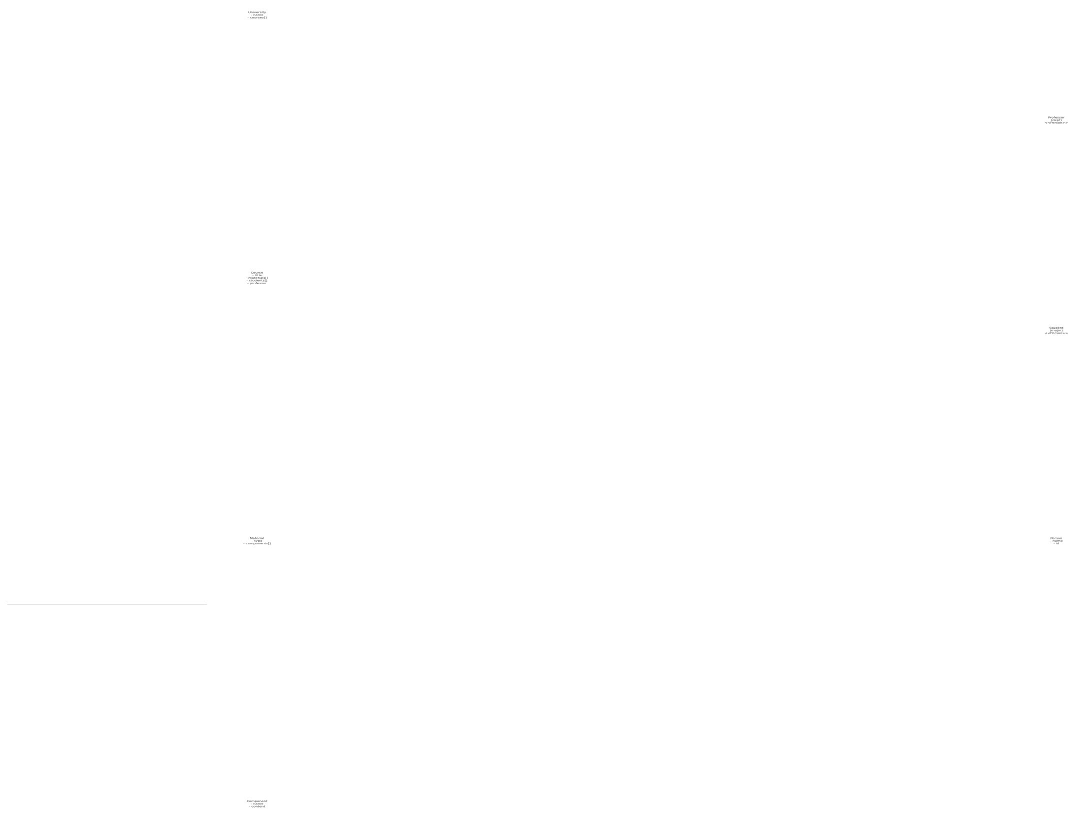
University Management System - OOP Relationships

This document presents the solution for the COS3612/S3632DO_FM_2025_S2_F Unit 1: OOP Paradigm Main Concepts, Week 2 Practical Session 1. The problem involves designing a university's internal management system that models relationships between classes such as people, courses, and enrollments.

Identified Classes:
1. University
2. Person
3. Student
4. Professor
5. Course
6. Material
7. Component

Relationships and Types:
- Student → Person: Inheritance
- Professor → Person: Inheritance
- Course ↔ Professor: Association (1 course has 1 professor)
- Course ↔ Student: Association (many-to-many)
- University → Course: Aggregation (1 university can have many courses)
- Course → Material: Composition (materials depend on the course)
- Material → Component: Composition (components depend on the material)

UML Class Diagram:



Relationship	Classes Involved	Type	Multiplicity
Inheritance	Student → Person	Inheritance	1-to-1
Inheritance	Professor → Person	Inheritance	1-to-1
Association	Course ↔ Professor	Association	1→1
Association	Course ↔ Student	Association	Many-to-many
Aggregation	University → Course	Aggregation	1→many
Composition	Course → Material	Composition	1→many
Composition	Material → Component	Composition	1→many

Python Implementation: class Person: def __init__(self, name, person_id): self.name = name self.person_id = person_id class Student(Person): def __init__(self, name, person_id, major): super().__init__(name, person_id) self.major = major class Professor(Person): def __init__(self, name, person_id, department): super().__init__(name, person_id) self.department = department class Component: def __init__(self, name, content): self.name = name self.content = content class Material: def __init__(self, material_type): self.material_type = material_type self.components = [] def add_component(self, component): self.components.append(component) class Course: def __init__(self, title, professor): self.title = title self.professor = professor self.materials = [] self.students = [] def add_material(self, material): self.materials.append(material) def enroll_student(self, student): self.students.append(student) class University: def __init__(self, name): self.name = name self.courses = [] def add_course(self, course): self.courses.append(course) def list_courses(self): return [course.title for course in self.courses]