**What to do (who will be doing)**

1. Report
   1. UML class diagram
   2. UML sequence diagram
   3. Design Considerations (Approach taken, Assumptions made, Principles used)
   4. Test Cases and Results
2. Code
3. Demonstration (Audio and Video)
   1. Functionalities
   2. Test Cases and Results
   3. (Optional) Essential parts in implementation
4. Compile disc

**UML Diagram**

ClassName: StudentManager (Control) - calls the Students methods if we need any info from the Students Class. In charge of creating/deleting/updating the students

Attributes:

Methods: deregisterStudents(), registerStudents(), getStudents(), setStudents(),

ClassName: Students (entity)

Attributes: (int) StudentID, (String) StudentName,

Methods: getStudentID(), getStudentName(), setStudentID(id: int): int, setStudentName(name: String): String,

ClassName: CourseManager (Control) - calls the Courses methods if we need any info from the Courses class.

Attributes:

Methods: deleteCourse(), getCourse(), setCourse(), createCourse()

ClassName: Courses (entity)

JY - Is it facultyName or the facultyMember (Course coordinator) as attribute?

Attributes: (int) CourseID, (String) CourseName, (int) vacancyNumber, (String) facultyName, (boolean) lecture, (boolean) tutorial, (boolean) laboratory

JY - Set exceptions for lecture, tutorial, laboratory combinations?

Methods: registerCourse(), displayVacancy(), displayClasses(), getCourseName(), getCourseID()

JY - Should we have getter and setter for each attribute?

ClassName: FacultyMembersManager (control)

Attributes:

Methods:

ClassName: FacultyMembers (entity)

Attributes: (int) facultyID, (String) facultyName,

Methods: getFacultyID(), getFacultyName(), setFacultyID(), setFacultyName()

ClassName: Grades (includes coursework component etc too) - (entity)

JY - Should include class participation and assignment?

Attributes: (double) exam, (double) coursework

Methods: getExamMarks(), getCourseworkMarks(), setExamMarks(), setCourseworkMarks(), computeOverallGrade(),

ClassName: ReadFile (to read in the datasets)

Attributes:

Methods: readFile()

ClassName: WriteFile (change datasets)

Attributes:

Methods: writeFile()

ClassName: Main (Boundary)

Attributes:

Methods:

Workload:

UML diagram: Josephine

Coding:

* Entity: Jimmy
* Create dataset with text documents, write and read Class: Elroy
* Controller: Jia Ren
* Boundary (main menu, similar to planeapp): Kevin