Main

-

+main(): void

Advisor

-advisorID: int
-fName: String
-lName: String

+getters(): type
+setters(type: Type): void
+advisorControl(chosenClasses: List<Strings>, student: Student): void

Courses

-name: String
-courseCode: String
-prerequisite: List<String>
-credit: int
-courseType: int
-semester: int
-courseYear: int
-theoreticalCourseHour: int
-practicalLessonHour: int

+getters(): type
+setters(type: Type): void
+checkIfPrerequisite(courses: Courses):Boolean
+checkIfTwoPrerequisite(courses: Courses):Boolean
+getPreRequisiteName(): String
+getTwoPreRequisiteName(courses: Courses):ArrayList<String>

Student

-studentId: int
-fName: String
-lName: String
-totalCredits: int
-advisorId: int
-gpa: double
-currentYear: int
-currentSemester: int
-currentSelectedCourses: List<String>
-completedCourses: List<CompletedCourses>
-availableCourses: List<String>
-failedCourses: List<FailedCourses>

+getters(): type
+setters(type: Type): void
+selectFromAvailableCourses(): void
+checkIfCourseFailed(courseCode: String): boolean
+sendToAdvisorSelectedClasses(advisors: Advisor[]): void
+changeSelectedCourses(advisorApprovedCourses: ArrayList<String>,
advisorRejectedCoursesAndReasons:ArrayList<String>): void
+gpaCalculator(courses: Courses[]): void

GenerateStudent

-student: List<Student>
-courses: List<Courses>
-firstSemesterCourses: List<String>
-secondSemesterCoursesHash: HashMap<String, List<String>>
-thirdSemesterCoursesHash: HashMap<String, List<String>>
-fourthSemesterCoursesHash: HashMap<String, List<String>>
-fifthSemesterCourses: HashMap<String, List<String>>
-sixthSemesterCoursesHash: HashMap<String, List<String>>
-seventhSemesterCoursesHash: HashMap<String, List<String>>
-eighthSemesterCoursesHash: HashMap<String, List<String>>
-prerequisiteList: HashMap<String, List<String>>

<<constructer>> GenerateStudents(student: Student[],courses: Courses[])
+addCourseNames(): void
+generateYear(student: Student): void
+semesterSetter(s: Student, semester: String): void
+setCoursesList(s: Student): void
+assignFailedCourses(currentSemesterFailed: List<FailedCourses> , courseCode: String): void
+prerequisiteControlAndLock(courseCode: String, lockedCourses: HashMap<String, List<String>>): void
+addCompletedCourses(currentSemesterCompleted: List<CompletedCourses>,courseCode: String,
grade: String, finishedSemester: int): void
+simulateFailedCourses(s: Student, currentSemesterCompleted: List<CompletedCourses>,
currentSemester: int): void
+unlockLockedCoursesAndSetAvailable(s: Student, completedCourses: List<CompletedCourses>,
lockedCourses: HashMap<String, List<String>>): void
+checkAvailableCourse(s: Student, currentSemesterCompleted: List<CompletedCourses>,
currentSemesterFailed: List<FailedCourses>, lockedCourses: HashMap<String, List<String>>,
currentSemesterCourses: HashMap<String, List<String>>):void
+checkCourseGiven(s: Student): void
+removeDuplicates(s: Student): void
+checkCourseHasPrerequisite(courseCode: String): void
+checkPrerequisiteCourseIsGiven(s: Student, courseCode: String, semester: int): boolean
+courseIsGivenAlready(s: Student, courseCode: String): boolean
+setStudentAdvisor(s: Student): void
+simulateSemester(s: Student, semester: String): void
+simulate(): void
+assignRandomGrades(): String

Randomizer

-

+setAvailableCoursesForEachStudent(students: Student[], courses: Courses[], advisors: Advisor[]) : void
+addCourseNames(students: Student[], courses: Courses[]) : void
+generate(students: Student[]): ArrayList <Integer>
+assignRandomGrades(): String

Transcript

-completedCourses: List <CompletedCourses>
-failedCourses: List<FailedCourses>
-gpa: Double
-completedCredits: int

+getters(): type
+setters(type: Type): void
+printTranscript(students2: Student[])

CompletedCourses

-courseName: String
-courseGrade: String
-givenSemester: int

+getters(): type
+setters(type): void
+toString(): String

FailedCourses

-courseName: String
-courseGrade: String

+getters(): type
+setters(type: Type): void
+toString(): String