

Student
<ul style="list-style-type: none"> - name: String - logString: String - studentID: StudentID - registrationOrder: int - grades: ArrayList<PassedCourseGrade> - currentCourse: ArrayList<Course> - advisor: Advisor - schedule: Schedule - transcript: Transcript - registrationController: RegistrationController
<ul style="list-style-type: none"> + Student(name: String, registrationYear: int, registrationOrder: int, registrationController: RegistrationController) - getPreviousCourses(): ArrayList<Course> + isPassed(course: Course): boolean + newCourses(courseSection: CourseSection): void + getGrade(course: Course): void + selectAndSendToAdvisor(): void

CourseSection
<ul style="list-style-type: none"> - course: Course - full: boolean - sectionHour: int - students: ArrayList<Student> - courseSchedule[][]: boolean - quota: int - numberOfQuotaFail: int - numberOfPrereqFail: int - numberOfCollisionFail: int - numberOfCreditFail: int - failedQuota: String - failedPrereq: String - failedCollision: String - failedCredit: String
<ul style="list-style-type: none"> + CourseSection(course: Course) - setCourseSchedule(): void + addStudent(student: Student): void + isFull(): boolean + setSectionHour(): void

Advisor
<ul style="list-style-type: none"> - name: String
<ul style="list-style-type: none"> + approveCourse(student: Student, courseSection: CourseSection): void

RegistrationController
<ul style="list-style-type: none"> - semester: String - year: int - courses: ArrayList<Course> - courseSections: ArrayList<CourseSection> - numberOfStudents: int - numberOfAdvisors: int - advisors: ArrayList<Advisor> - students: ArrayList<Student> - currentYear: int
<ul style="list-style-type: none"> + RegistrationController() - startRegistration(): void + getNamesList(): ArrayList<String> - createAdvisors(): void - createStudents(): void - assignAdvisorsToStudents(): void - addPreviousCourses(): void - selectCourses(): void - showOutput(): void - showStatistics(): void + createNewCourseList(student: Student): ArrayList<CourseSection> - findCourse(courseCode: String): Course

WeeklySchedule
<ul style="list-style-type: none"> - stuSchedule: CourseSection[][] - student: Student
<ul style="list-style-type: none"> + WeeklySchedule(student: Student) + addToSchedule(courseSection: CourseSection): void + isCollision(courseSection: CourseSection): boolean

PreviousCourseGrade
<ul style="list-style-type: none"> - decimalGrade: int - course: Course
<ul style="list-style-type: none"> + PassedCourseCode(course: Course, decimalGrade: int) + convertLetterGrade(): String

Transcript
<ul style="list-style-type: none"> - student: Student
<ul style="list-style-type: none"> + Transcript() + calculateCredits(): int + calculateGPA(): double

StudentID
<ul style="list-style-type: none"> - student: Student - department: String= "1501" {readOnly}
<ul style="list-style-type: none"> + StudentID(year: int, rank: int) - registrationYear(): String + getStudentID(): String + toString(): String

Course
<ul style="list-style-type: none"> - courseCode: String - courseName: String - semester: String - quota: int - credits: int - year: int - sectionHours[2]: int - requiredCredits: int - prerequisite: Course - passingCourseProb: double
<ul style="list-style-type: none"> + Course(courseCode: String, courseName: String, semester String, quota: int, credits: int, theoric: int, lab: int, year: int, requiredCredits: int, preRequisite: Course, passingCourseProb: double) + setPrerequisite(prerequisite: Course): void + getSectionHours(): int + setSectionHours(theoric: int, lab: int): void