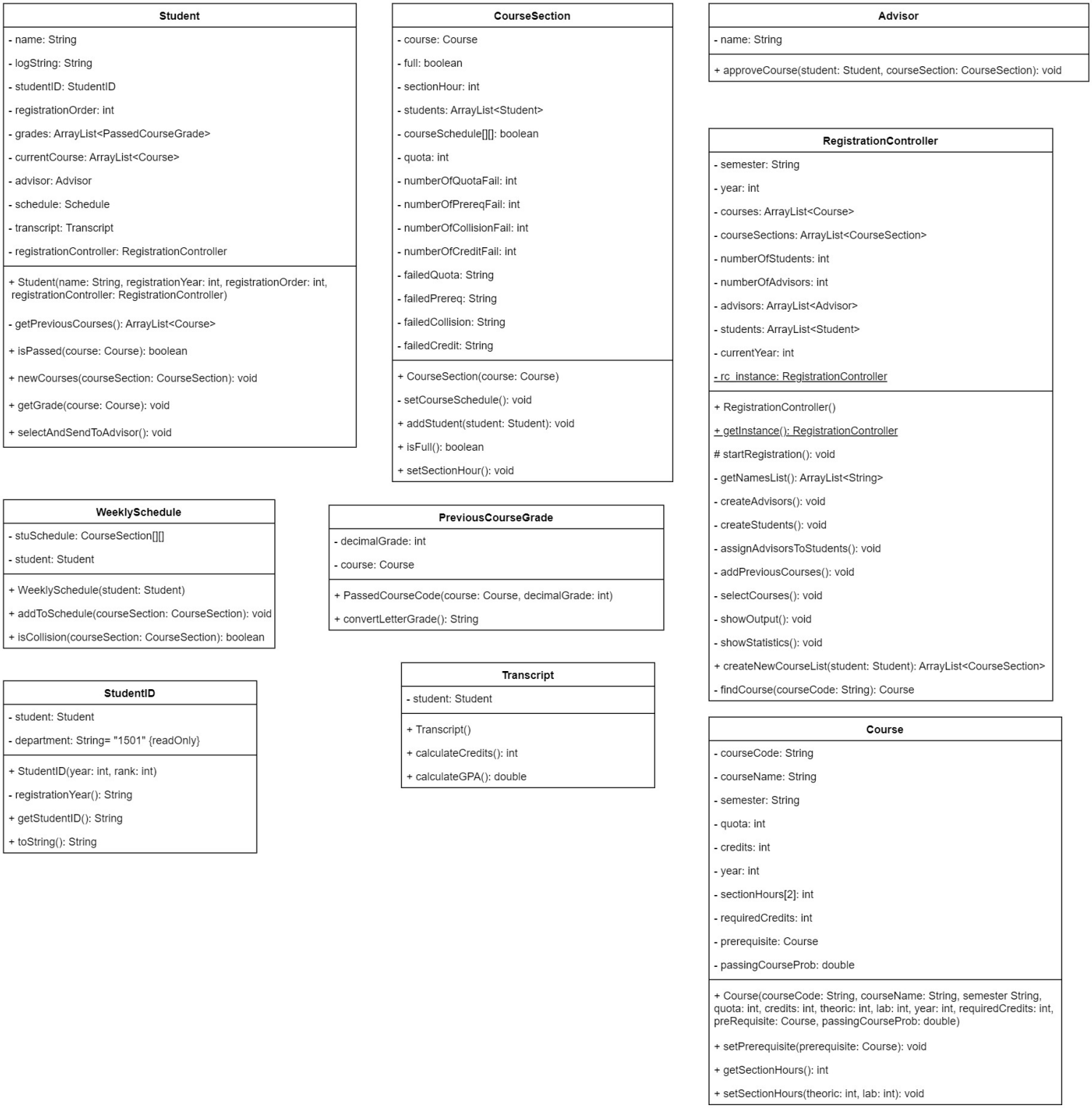
Student
- name: String
- logString: String
- studentID: StudentID
- registrationOrder: int
- grades: ArrayList<PassedCourseGrade>
- currentCourse: ArrayList<Course>
- advisor: Advisor
- schedule: Schedule
- transcript: Transcript
- registrationController: RegistrationController
+ Student(name: String, registrationYear: int, registrationOrder: int, registrationController: RegistrationController)
- getPreviousCourses(): ArrayList<Course>
+ isPassed(course: Course): boolean
+ newCourses(courseSection: CourseSection): void
+ getGrade(course: Course): void
+ selectAndSendToAdvisor(): void
CourseSection
- 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
+ CourseSection(course: Course)
- setCourseSchedule(): void
+ addStudent(student: Student): void
+ isFull(): boolean
+ setSectionHour(): void
Advisor
- name: String
+ approveCourse(student: Student, courseSection: CourseSection): void
RegistrationController
- semester: String
- year: int
- courses: ArrayList<Course>
- courseSections: ArrayList<CourseSection>
- numberOfStudents: int
- numberOfAdvisors: int
- advisors: ArrayList<Advisor>
- students: ArrayList<Student>
- currentYear: int
- rc_instance: RegistrationController
+ RegistrationController()
+ getInstance(): 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
- stuSchedule: CourseSection[][]
- student: Student
+ WeeklySchedule(student: Student)
+ addToSchedule(courseSection: CourseSection): void
+ isCollision(courseSection: CourseSection): boolean
PreviousCourseGrade
- decimalGrade: int
- course: Course
+ PassedCourseCode(course: Course, decimalGrade: int)
+ convertLetterGrade(): String
Transcript
- student: Student
+ Transcript()
+ calculateCredits(): int
+ calculateGPA(): double
StudentID
- student: Student
- department: String= "1501" {readOnly}
+ StudentID(year: int, rank: int)
- registrationYear(): String
+ getStudentID(): String
+ toString(): String
Course
- courseCode: String
- courseName: String
- semester: String
- quota: int
- credits: int
- year: int
- sectionHours[2]: int
- requiredCredits: int
- prerequisite: Course
- passingCourseProb: double
+ 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