JSONWriter

```
+ JSONWriter():

- department: Department
- objectMapper: ObjectMapper
- jsonNode: JsonNode

+ writeStudents(): void
+ writeJson(): void
+ start(Department): void
+ writeTranscripts(): void
+ writeRequests(): void
```

JSONReader

```
+ JSONReader():

- studentRegistrationMap: Map<Stud
- jsonNode: JsonNode
- mapper: ObjectMapper
- courseSectionCodesMap: Map<Co
- department: Department
- coursePrerequisiteCourseCodesMa
- lecturerIDCoursesMap: Map<Intege
- studentIDDraftMap: Map<Integer, L
- advisorIDCoursesMap: Map<Intege
- studentIDAdvisorIDMap: Map<Integ

+ readStudents(): void
+ start(Department): void
+ readCourseSections(): void
+ syncObjects(): void
+ readCourses(): void
+ readLecturers(): void
+ readRequests(): void
+ readAdvisors(): void
+ readJson(): void
+ readTranscript(Student): void
```

main

```
+ main():

+ main(String[]): void
```

Transcript

```
+ Transcript(Student):

- takenCredits: int
- student: Student
- courseGradeMap: Map<Course, Grade>
- cgpa: double
- studentCourses: List<Course>
- gradeLevel: byte
- completedCredits: int

+ getCourseGradeMap(): Map<Course, Grade
+ calculateCgpa(): double
+ setStudentCourses(List<Course>): void
+ setStudent(Student): void
+ getCgpa(): double
+ showTranscript(): void
+ getCompletedCredits(): int
+ setCourseGradeMap(Map<Course, Grade>):
+ getStudentCourses(): List<Course>
+ setTakenCredits(int): void
+ getTakenCredits(): int
```

Person {abstract}

```
+ Person(int, String, String, String, String):

- ID: int
- name: String
- userName: String
- password: String
- surname: String
- department: Department

+ getName(): String
+ setDepartment(Department): void
+ getID(): int
+ setUserName(String): void
+ getPassword(): String
+ login(String, String): boolean
+ setPassword(String): void
+ getSurname(): String
+ getUserName(): String
+ getDepartment(): Department
```

Department

```
+ Department(String):

- transcripts: List<Transcript>
- courseCodeCourseMap: Map<String, Course>
- sectionCodeCourseMap: Map<String, Course>
- sectionCourseMap: Map<CourseSection, Course>
- studentIDStudentMap: Map<Integer, Student>
- maxCourseNumber: byte
- advisors: List<Advisor>
- userNamePersonMap: Map<String, Person>
- advisorIDAdvisorMap: Map<Integer, Advisor>
- students: List<Student>
- departmentName: String
- courseSections: List<CourseSection>
- courses: List<Course>
- lecturers: List<Lecturer>

+ getAdvisorIDAdvisorMap(): Map<Integer, Advisor>
+ getStudents(): List<Student>
+ getTranscripts(): List<Transcript>
+ getCourseSections(): List<CourseSection>
+ getLecturers(): List<Lecturer>
+ getStudentIDStudentMap(): Map<Integer, Student>
+ getUserNamePersonMap(): Map<String, Person>
+ getMaxCourseNumber(): byte
+ getDepartmentName(): String
+ getSectionCourseMap(): Map<CourseSection, Course>
+ getCourses(): List<Course>
+ getAdvisors(): List<Advisor>
+ getCourseCodeCourseMap(): Map<String, Course>
+ getSectionCodeCourseMap(): Map<String, Course>
```

Course

```
+ Course(String, String, int, byte):

- courseCode: String
- lecturer: Lecturer
- courseName: String
- courseSections: List<CourseSection>
- preRequisiteCourses: List<Course>
- courseCredit: int
- gradeLevel: byte

+ getCourseSections(): List
+ getCourseCredit(): int
+ getGradeLevel(): byte
+ getCourseCode(): String
+ getCourseName(): String
+ setLecturer(Lecturer): void
+ getPreRequisiteCourses(): List<Course>
```

CourseSection

```
+ CourseSection(Course, String, String, int):

- students: List
- courseSectionCode: String
- hour: int
- day: String

+ getCourseSectionCode(): String
```

<<interface>> IDisplayMenu

```
+ printMenu(String): void
```

Lecturer

```
+ Lecturer(int, String, String):
+ Lecturer(int, String, String, String, String):

- lessonsTaught: List<Course>

+ getLessonsTaught(): List<Course>
+ login(String, String): boolean
```

Student

```
+ Student(int, String, String, String, String, byte):

- advisor: Advisor
- gradeLevel: byte
- transcript: Transcript
- availableCourses: List<Course>
- draft: List<Course>
- hasRequest: boolean

+ setUserName(String): void
+ getDraft(): List<Course>
+ showRequestStatus(): void
+ setPassword(String): void
+ getAdvisor(): Advisor
+ dropCourse(): void
+ setHasRequest(boolean): void
+ setDraft(List<Course>): void
+ getAvailableCourses(): List<Course>
+ login(String, String): boolean
- calculateNumberOfCourses(): int
+ addCourse(): void
+ isHasRequest(): boolean
+ setTranscript(Transcript): void
+ getDepartmentName(): String
+ setAdvisor(Advisor): void
+ getTranscript(): Transcript
+ viewAvailableCourses(): void
+ sendRequest(): void
+ getGradeLevel(): byte
+ printMenu(String): void
```

CourseRegistrationSystem

```
+ CourseRegistrationSystem():

- choice: int
- input: Scanner
- department: Department

+ start(): void
+ exitProgram(): void
+ courseSelectionMenu(Student): void
+ getInput(): int
+ printMenu(String): void
+ studentMenu(Student): void
+ loginMenu(Department): void
+ mainMenu(Department): void
+ advisorMenu(Advisor): void
```

Advisor

```
+ Advisor(int, String, String, String, String):

- studentsAdvised: List<Student>
- requestList: List<Registration>
- requestNumber: int

+ login(String, String): boolean
+ getStudentsAdvised(): List<Student>
+ getRequestList(): List<Registration>
+ printRequests(): void
+ printMenu(String): void
+ getRequests(): List<Registration>
+ replyRequests(): void
```

Registration

```
+ Registration(Student, List<Course>):

- courses: List<Course>
- student: Student

+ rejectRequest(): void
+ getCourses(): List<Course>
+ getStudent(): Student
+ approveRequest(): void
+ addRequest(Advisor): void
```
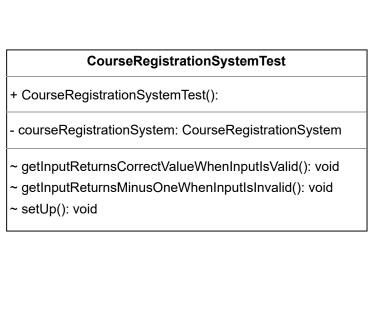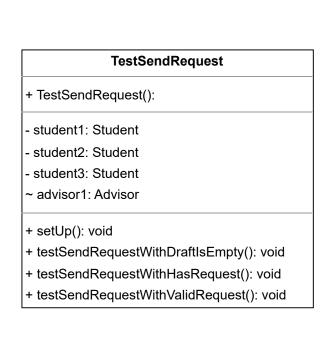
Grade

```
+ Grade(String):

- gradeOver4: double
- gradeOver100: int
- letterGrade: String

- setGradeOver4(): void
- setGradeOver100(): void
+ getLetterGrade(): String
+ getGradeOver4(): double
```

TranscriptTest

```
+ TranscriptTest():

- course2: Course
- course1: Course
- transcript: Transcript
- course3: Course

~ calculateCgpaWithCoursesAndGrades(): void
~ calculateCgpaWithCoursesNoGrades(): void
~ setUp(): void
~ calculateCgpaWithNoCourses(): void
```

CourseRegistrationSystemTest

```
+ CourseRegistrationSystemTest():

- courseRegistrationSystem: CourseRegistrationSystem

~ getInputReturnsCorrectValueWhenInputIsValid(): void
~ getInputReturnsMinusOneWhenInputIsInvalid(): void
~ setUp(): void
```

calculateCompletedCreditsTest

```
+ calculateCompletedCreditsTest():

- grade4: Grade
- course3: Course
- course2: Course
- grade1: Grade
- course4: Course
- transcript: Transcript
- grade2: Grade
- course1: Course
- grade3: Grade

~ calculateCompletedCredits(): void
~ setUp(): void
```

TestSendRequest

```
+ TestSendRequest():

- student1: Student
- student2: Student
- student3: Student
- advisor1: Advisor

~ setUp(): void
+ testSendRequestWithDraftIsEmpty(): void
+ testSendRequestWithHasRequest(): void
+ testSendRequestWithValidRequest(): void
```

RegistrationTest

```
+ RegistrationTest():

- courses: List<Course>
- course1: Course
- student: Student
- course2: Course
- registration: Registration

~ approveRequestTest(): void
~ rejectRequestTest(): void
~ addRequestTest(): void
~ setUp(): void
```

PersonTest

```
+ PersonTest():

- person: Person

+ setUp(): void
+ loginShouldReturnFalseWhenCredentialsAreInvalid(): void
+ loginShouldReturnTrueWhenCredentialsAreValid(): void
```