Semester Class:

term: String(Spring Fall Summer)
Year: String (four characters min/max)
LinkedList<Course> courses

createSemester()-> void
    print What term are you in? (Spring or Fall):
    accept correct input and update term
    print What year is it? (Spring or Fall):
    accept correct input and update year
    semesterMenu()

semesterMenu()-> void
        prints semester header
        listSemesterCourses()
        Prints
            1. Add Course to Semester
            2. Edit Course
            3. Delete Course
            4. Wipe Semester
        Prompts user to select course based on index displayed
        Invokes corresponding function

listSemesterCourses()-> void
    for course in Courses
        Print course index
        Print course

addCoursetoSemester(courses) -> void
    createCourse()
    Adds course to courses

editCourseInSemester()-> void
    courseNum: int
    Which course would you like to edit (ie 1…..N):
    accept correct input and update courseNum
    editCourse(courses[courseNum-1])

// TODO: create sorting algorithm to reduce search time

deleteCourse()-> void

prompts user of course number
accepts correct input and saves to courseNum
Remove respective course from courses

Course Class:

name: String
overall_score: float
LinkedList<Field> fields

createCourse() -> void

Prints What is the name of the course you would like to add?:
Accepts correct input
Saves to name
courseMenu()

courseMenu() -> void

listCourseFields()

prints
1. Edit Name
2. Add Field to Course
3. Edit Field in Course
4. Delete Field

updateCourseScore()

overallScore = current grade for course
for all fields
for all scores

sum scores
sum / length of scores
overall_score += (sum)*fields[i].weight

addFieldToCourse() -> void

createField()
Adds field to fields
updateCourseScore()

editCourseField() -> void

courseNum: int
Which field would you like to edit (ie 1…..N):
accept correct input and update courseNum
courses[courseNum-1].editField()
updateCourseScore()

deleteCourseField()-> void

prompts user of fieldnumber
accepts correct input and saves to fieldNum
Remove respective field from fields
updateCourseScore

Field Class:

name: String
weight: float
dropPolicy: int (default = 0)

LinkedList<int> scores

getName
setName
getWeight
setWeight
getDropPolicy
setDropPolicy

createField() -> void

//prompt user to select field
Accepts correct input and setName()
prompt user to enter weight
Accepts correct input and setWeight()
fieldMenu()

calcTotalScore()

fieldMenu()-> void

listFieldScores()

prints
1. Edit Name
2. Edit Weight
3. Edit Drop Policy
4. Add Score to Field
5. Delete Score
6. DisplayField
7. Exit

Prompts user to select course based on index displayed
Invokes corresponding function