5. Design a course registration platform.
Wenchao Ma

```
/**
 * neu-sep18 Assignment1
 * 5. Design a course registration platform.
 * @author Wenchao Ma
 */
```

Problem : Design a course registration platform.

Student:
Data: UserName, Passcode, FirstName, LastName, Pictures, Birthday, StudentNumber, Address, PhoneNumber, Email, Course, Grade, College, Major, RegisteredCourse, GPA
Behaviors : SignIn, SignOut, SearchForCourse, Setting, ViewCourseInfo, RegisterForCourse, ViewRegisteredCourseInfo, GetNotify

Professor:
Data: UserName, Passcode, FirstName, Pictures, LastName, Birthday, IDNumber, Address, PhoneNumber, Email
Behaviors : SignIn, SignOut, EditCourseInfo, SubmitCourseInfo, DeleteCourseInfo, Setting, ViewCourseInfo, GetNotify

Course:
Data: Title, CourseNumber, CreditHours, Level, Term, CourseDescription, Syllabus, Restriction, DateAndTime, Enrollment, Corequisites, Prerequisites, MutualExclusion, Subject, EnrollmentSeatsAvailable

Connections:
Data: IpAddressData, DomainData, LocalIpAddressData
Behaviors: EstablishConnection

Platform:
Data : Name, PhoneNumber, Email, Student, Professor, Course
Behaviors : RegisterForCourse, SignIn, SignOut, SearchForCourse, SendNotify, Recommendation, CheckStudentQualification, ShowErrorInfo, CheckStudentMaxCreditHour, CheckCourseAvailable

Sequence of invoking behaviors on Objects:

Design a course registration platform.

Student student

```
Professor professor
Courses course
Connections connection
Platform nuplatform

        if connection is established
      if this student did not sign in
         student.SignIn -> student.UserName, student.Passcode, nuplatform, :
signResult
         if signResult is fail
               student.tryAgainLater
      if this professor did not sign in
         professor.SignIn -> professor.UserName, professor.Passcode, nuplatform, :
signResult
         if signResult is fail
               professor.tryAgainLater

      professor.EditCourseInfo -> course.Title, course.CourseNumber,
course.CreditHours, course.Level, course.Term, course.CourseDescription,
course.Syllabus, course.Restriction, course.DateAndTime, course.Enrollment,
course.Corequisites, course.Prerequisites, course.MutualExclusion, course.Subject
      professor.SubmitCourseInfo -> course, nuplatform : submitResult
      if submitResult is fail
        professor.tryAgainLater

      student.ViewRegisteredCourseInfo

      while student wants to choose a new course
        student.SearchForCourse -> Title, CourseNumber, Level, Term, Subject,
College : courseList
        Initialize pageNumber = 1
        Loop in courseList:
               student.ViewCourseInfo
               if student.findTheCourse
               course = findTheCourse
               break
               if courseList.hasNextPage
                  pageNumber = pageNumber +1
               else if courseList.notHasNextPage
                        break
        student.RegisterForCourse -> course, nuplatform
        nuplatform.CheckStudentQualification -> student.Grade, student.College,
student.Major, student.RegisteredCourse, course : StudentQualification
        nuplatform.CheckCourseAvailable -> course.EnrollmentSeatsAvailable :
CourseAvailable
        nuplatform.CheckStudentMaxCreditHour -> student.RegisteredCourse :
MaxCreditHour
```

```
        if StudentQualification or CourseAvailable or MaxCreditHour is fail
                nuplatform.ShowErrorInfo -> student, course
                student.tryAgainLater
        nuplatform.SendNotify -> student, course, professor
        professor.GetNotify
        student.GetNotify
        course.EnrollmentSeatsAvailable minus 1


else if connection is null
    student.tryAgainLater
    professor.tryAgainLater
```