Progetto:

Classi :

- Professor :

Private:

- Name
- Surname
- Number

Pubblic:

- getName() const
- getSurname() const
- getNumber() const

- Student :
  - Name
  - Surname
  - Number

- Lesson :

Private:

- Course* course
- Date* Date
- Hour* Start
- Hour* Duration
- Classroom* classroom

Public:

- getCourse() const
- getDate() const
- getStart() const
- getDuration() const
- getClassroom() const
- printLesson()

- Classroom :

Private:

- String ID
- Int Seats

Public:

- getID() const
- getSeats() const

- Course :

Private:

- Professor* Lista professori
- Int year
- Int StudentsCounter
- Student* Lista studenti
- Lesson* Lista lezioni

Public:

- Course()
- ~Course()
- addProfessors(Professor* professor)
- addStudents(Student* student)
- addLessons(Lesson* lesson)
- getProfessor() const
- getYear() const
- getStudents const
- getLessons const
- SeatsController(const &Course) const

- DegreeCourse :

Private:

- Course* Lista corsi
- Professor* Lista docenti
- Classroom* Lista aule
- Lesson* Lista lezioni
- Student* Lista studenti

Public:

- CourseFinderByProfessor(const &Docente) const
- CourseFinderByClassroom(const &Aula) const

- ProfessorFinderByCourse(const& Corso) const
- ClassroomFinderByCourse (const &Corso) const
- FullCoursesOverview(const &CorsoDiLaurea) const
- NoOverlapController(const &DegreeCourse) const

Evoluzione :

1) Unica classe madre per docenti e studenti :
   - User :

   Protected:

   - String Name
   - String Surname
   - Int Number

   Public:

   - getName() const
   - getSurname() const
   - getNumber() const

   Docente e studente ereditano senza bisogno di metodi o attributi aggiuntivi

2) Tutti i metodi get fanno un return degli attributi privati delle classi

3) SeatsController(const &Corso) const -> prende un riferimento costante al corso e fa un assert dei posti a sedere delle aule col numero di studenti che devono partecipare al corso -> mi servirà un metodo per aggiungere studenti alla lista e una variabile che conteggi il numero degli studenti aggiunti -> StudentsCounter + AddStudents

4) Classe per la gestione della data e dell'ora:

   - Date:

   Private:
   - Int Day
   - String Month
   - Int Year

   Pubblic:

   - getDay() const

- getMonth() const
- getYear() const
- printDate() const

- Hour:

  Private:
  - Int Hour
  - Int Minutes

  Public:
  - getHour() const
  - getMinutes() const
  - printHour() const

5) Tutti i metodi finder :
   - CourseFinderByProfessor(const &Docente) const
   - CourseFinderByClassroom(const &Aula) const
   - ProfessorFinderByCourse(const& Corso) const
   - ClassroomFinderByCourse (const &Corso) const

   Ciclano nelle liste della classe DegreeCourse e fanno dei controlli per trovare uguaglianza con gli attributi dell'oggetto che gli si passa come riferimento costante. Nel caso dei docenti usiamo la matricola perché è univoca per ogni persona e risparmiamo codice. Per l'aula confrontiamo gli ID, per ProfessorFinderByCourse basta stampare tutti i corsi che hanno quel docente nella loro lista professori.
6) Implementazione deimetodi di print per classi lesson e course poi richiami i print nell'overview della classe DegreeCourses per fare un print più grande di tutti i corsi
7) NoOverlapController deve ciclare nella lista delle lezioni dei corsi della degreecourse class e con if che controlla l'anno del corso ciclare ancora per controllare che non ci siano coincidenze di data, e ora di lezioni di corsi diversi