CECS 282 – Assignment 2

DYNAMIC ARRAY (so… only use Dynamic Array)

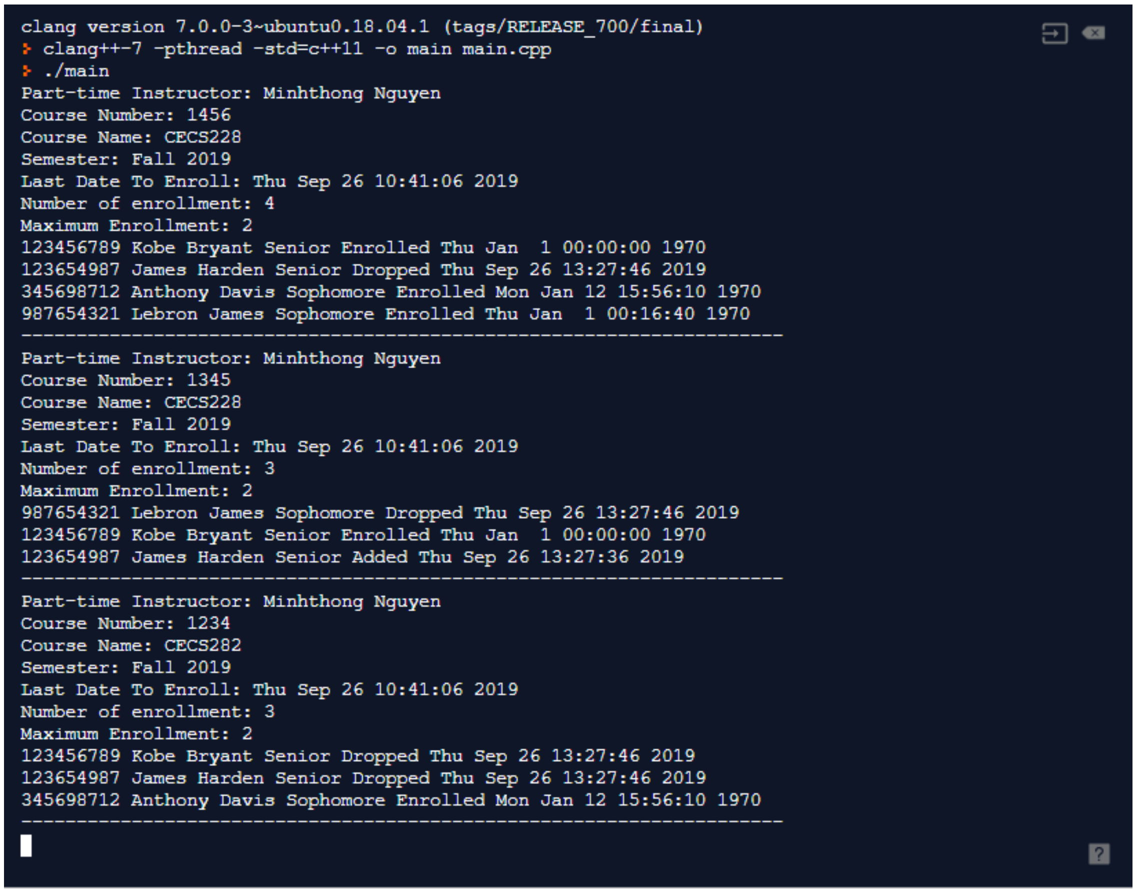
In this assignment, you will design and implement several classes.

```
Student Class:
private:
  string level;//Freshman, Sophomore,
Junior, Senior
  string id;
  string name; //formatted as First
Last (assume no middle name)
  string status; //Added, Enrolled,
Dropped
  Time date_of_action;//time in secs
when enroll, add or drop occurs
public:
  Student();
  Student(string id, string name,
string level, Time date_of_action);
  Student(const Student& s);
  Student& operator=(const Student&
s);
  string getLevel() const;
  string getId() const;
  string getName() const;
  string getDateOfAction() const;
//return as Www Mmm dd hh:mm:ss yyyy
  Time getTimeOfAction() const;
//return date_of_action in seconds
  string getStatus() const;
  void setStatus(string stat);
  void setTimeOfAction(Time act);

Course Class:
private:
  int numOfEnrolled;
  string courseNumber;//four-digit
integer
  string courseName;//e.g. CECS282
  string semester;// Fall, Winter,
Spring, Summer
  Time last_date_to_enroll;//time in
secs
  Students* students; //pointer to
array of students
public:
  Course();
  Course(string num, string name,
string sem, Time last_date, Student*
stdts, int numOfEnroll);
```

```
Time Class:
private:
  time_t secs; //time in seconds. if
secs = 0, it represents Jan 1st, 1970
public:
  Time();
  Time(time_t t);
  Time(const Time& t);
  string toString() const; //return
time in the following format Www Mmm
dd hh:mm:ss yyyy
  string getMMDDYYYY() const;//return
time in the following format
MM/DD/YYYY
  string getYear() const;//return
year of time.
  int compareTime(const Time& t);//
return -1 if less than, 0 if equal,
and 1 if more than

Instructor Class:
private:
  int numOfCoursesTaught;
  string name; //formatted as First
Last (assume no middle name)
  string status; //full-time or part-
time or tenured
  Course* courses; //pointer to array
of courses
public:
  Instructor();
  Instructor(string name, string
status, Course* crs, int num);
  Instructor(const Instructor& i);
  ~Instructor();
  Course* getCourse()  const;
  int getNumberOfCoursesTaught()
const;
  string getName() const;
  string getStatus() const; //Part-
time or Full-time or Tenured
  string getStudentStatus(const
Student& s, const Course& c) const;
//Enrolled, Added, or Dropped
```

```
  Course(const Course& c);
  Course& operator=(const Course& c);
  ~Course();
  Student* getStudent() const;
  string getCourseNumber() const;
  string getCourseName() const;
  string getSemester() const;
  string getYear() const; //return
the year of the semester (assume to
be the year of the last date to
enroll)
  int getNumberOfEnrollment()
const;
  string getLastDateToEnroll() const;
// return as Www Mmm dd hh:mm:ss yyyy
  Time getTimeLastDateToEnroll() const;
// return last_date_to_enroll in
seconds
   void setNumberOfEnrollment(int ne);
   void setRoster(Student* stdts);
```

```
  int addStudent(const Student& s,
Course& c); //return -1 if a student
already exists; return 0 if a student
is not on the roster. Otherwise,
return 1.
  int dropStudent(const Student& s,
Course& c, Time t); //return 0 if a
student is not on the roster.
Otherwise, return 1.
  int findStudent(const Student& s,
const Course& c);
//return 0 if a student is not found.
Otherwise, return 1.
  int addCourse(const Course& c);
//return -1 if course already exists;
return 0 if the numOfCoursesTaught
reaches MAXCOURSE. Otherwise, add the
course and return 1.
    int findCourse(const Course& c);
//return 0 if a course is not found.
Otherwise, return 1.
```



```
clang version 7.0.0-3~ubuntu0.18.04.1 (tags/RELEASE_700/final)
> clang++-7 -pthread -std=c++11 -o main main.cpp
> ./main
Part-time Instructor: Minhthong Nguyen
Course Number: 1456
Course Name: CECS228
Semester: Fall 2019
Last Date To Enroll: Thu Sep 26 10:41:06 2019
Number of enrollment: 4
Maximum Enrollment: 2
123456789 Kobe Bryant Senior Enrolled Thu Jan  1 00:00:00 1970
123654987 James Harden Senior Dropped Thu Sep 26 13:27:46 2019
345698712 Anthony Davis Sophomore Enrolled Mon Jan 12 15:56:10 1970
987654321 Lebron James Sophomore Enrolled Thu Jan  1 00:16:40 1970
--------------------------------------------------------------------
Part-time Instructor: Minhthong Nguyen
Course Number: 1345
Course Name: CECS228
Semester: Fall 2019
Last Date To Enroll: Thu Sep 26 10:41:06 2019
Number of enrollment: 3
Maximum Enrollment: 2
987654321 Lebron James Sophomore Dropped Thu Sep 26 13:27:46 2019
123456789 Kobe Bryant Senior Enrolled Thu Jan  1 00:00:00 1970
123654987 James Harden Senior Added Thu Sep 26 13:27:36 2019
--------------------------------------------------------------------
Part-time Instructor: Minhthong Nguyen
Course Number: 1234
Course Name: CECS282
Semester: Fall 2019
Last Date To Enroll: Thu Sep 26 10:41:06 2019
Number of enrollment: 3
Maximum Enrollment: 2
123456789 Kobe Bryant Senior Dropped Thu Sep 26 13:27:46 2019
123654987 James Harden Senior Dropped Thu Sep 26 13:27:46 2019
345698712 Anthony Davis Sophomore Enrolled Mon Jan 12 15:56:10 1970
--------------------------------------------------------------------
```

Input:
A test file called *roster.cpp* will be provided to test the correctness of your classes. It is important that you are successfully implement all classes since the output files depend on this tester. After that, you can change and add more codes to the *roster.cpp* to produce the output files. A sample output to test your classes is provided above. Note that the number of enrollment does not matchup with the status. However, this issue will be corrected in the *courseName_courseNumber.txt*.

Output:
A. Your raw output is *raw_data_courseNumber.txt*, which is formatted as:

```
123456789 Bryant Kobe Senior Enrolled Sat Aug 3 16:40:32 2019
456789012 James Lebron Sophomore Added Fri Aug 30 17:00:00 2019
987654321 Davis Anthony Freshman Dropped Thu Aug 15 13:05:57 2019
```

Note:
- There are 6 fields to each line. Each field is separated by a single space.
  - ID: has the exact length of 9 numerical (0-9) characters.
  - Last Name: has maximum length of 15 characters including white spaces.
  - First Name: has maximum length of 15 characters including white spaces.
  - Level: has maximum length of 9 characters including white spaces.
  - Status: has maximum length 8 characters including white spaces.
  - Date: has the exact length of 24 characters including white spaces.
- Any changes to this file (i.e. enroll, add, or drop) must update this file properly.

B. Your formatted output is *courseName_courseNumber.txt*, which is formatted as below.

```
Part-time Instructor: Minhthong Nguyen
Course Number: 1234
Course: CECS 282
Semester: Fall 2019
Enrollment: 2
Max Enrollment: 35
Last date to enroll: Tue Aug 20 23:59:59 2019
ID        Last Name       First Name      Level     Status   Date
123456789 Bryant          Kobe            Senior    Enrolled 08/03/2019
456789012 James           Lebron          Sophomore Added    08/30/2019
```

- The first line is the course number.
- The second line is name of instructor
- The third line is the current semester
- The fourth line is the number of total enrollment
- The fifth line is the maximum number of enrollment
- The sixth line is the last date to enroll (formatted as `Www Mmm dd hh:mm:ss yyyy`)

- The rest of the lines are the fields from the *raw_data_courseNumber.txt*. The width of each field is the same as above except that Date field now has length of 10 characters. All fields are left-aligned.
- Only students with status of Enrolled or Added appear in this file.
- This file must be updated properly as students are enrolled, added or dropped from a course.

Other guidlines:
- You can add more member functions to those classes but DO NOT add or change anything else.
- The day of the week must match with the date. In other words, DO NOT make this up (`ctime` library can help you with this).
- Maximum enrollment (called `CAPACITY`) is always 35.
- Maximum number of courses (called `MAXCOURSE`) of an instructor is 3;
- `CAPACITY` and `MAXCOURSE` must be made global for all cpp and header files to use (Hint: use `extern`).
- The number of enrollment may exceed the `CAPACITY` but the `CAPACITY` value remains 35.
- A student with "`Enrolled`" status if enrolling before or on the last date to enroll
- A student with "`Added`" status if enrolling after the last date to enroll
- A student with "`Dropped`" status if dropped from the course at any time after the initial add or enroll.
- All files should not have duplicates of students.
- Add the same course or student again will not affect the roster.
- Add or drop the same student again will not affect the roster.
- Each course has a unique course number.
- Each student has a unique id number.
- Any first name and last name with more than 15 characters will be cut off in both raw_data.txt and course.txt file.
- Assume that all students and instructors do not have middle name.
- Use `ctime` library to implement the Time class.
- Please include the following block at the beginning of your program

```
/*
Name:
Class: CECS 282
Instructor: Minhthong Nguyen
Purpose of the program:
Last updated:
*/
```

- Comment your code.
- Follow standard style for coding (refer to java docs).

Deliverables:
Turn-in all files (header, cpp, and txt) to Dropbox and bring a physical copy of all files (header, cpp, and txt) when you demo your program.