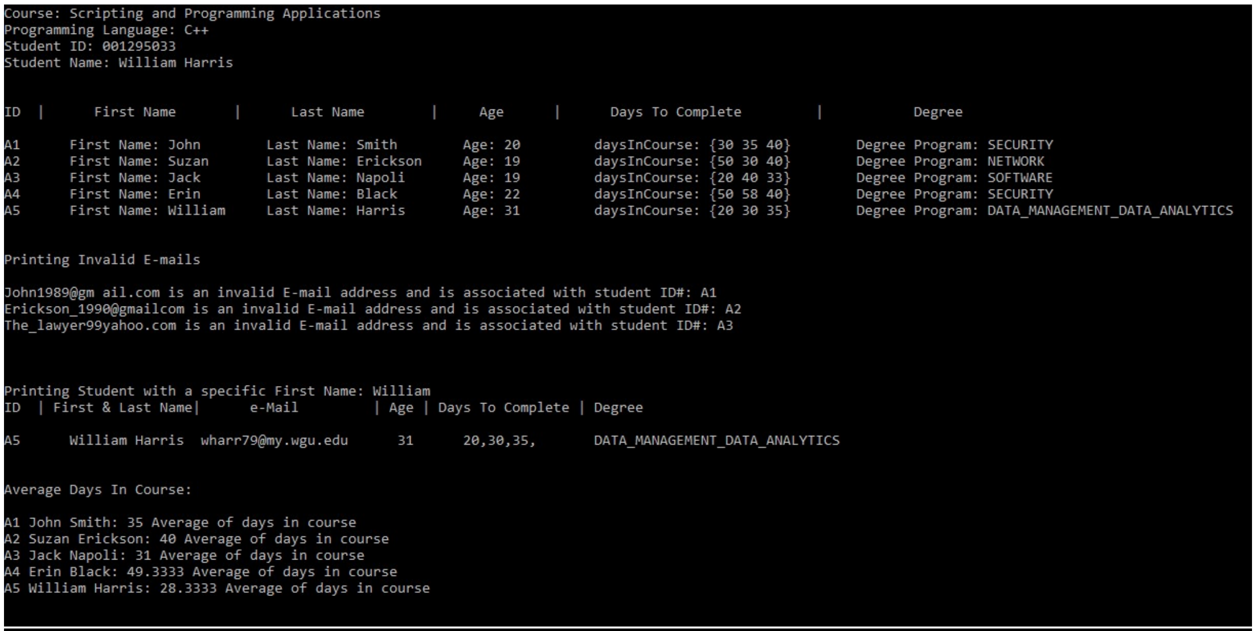
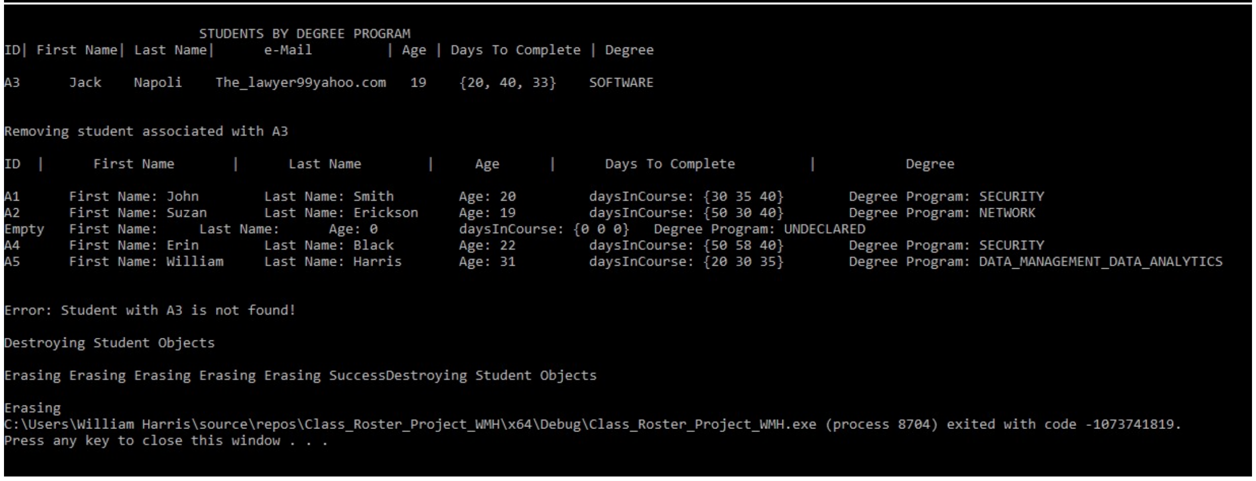
William Harris
Student ID:001295033
Wednesday, October 21, 2020

The following is screenshots of what should appear when application `Class_Roster_Project_WMH has executed properly`:



```
Course: Scripting and Programming Applications
Programming Language: C++
Student ID: 001295033
Student Name: William Harris


ID  |      First Name       |      Last Name        |     Age      |      Days To Complete         |           Degree

A1      First Name: John        Last Name: Smith        Age: 20         daysInCourse: {30 35 40}        Degree Program: SECURITY
A2      First Name: Suzan       Last Name: Erickson     Age: 19         daysInCourse: {50 30 40}        Degree Program: NETWORK
A3      First Name: Jack        Last Name: Napoli       Age: 19         daysInCourse: {20 40 33}        Degree Program: SOFTWARE
A4      First Name: Erin        Last Name: Black        Age: 22         daysInCourse: {50 58 40}        Degree Program: SECURITY
A5      First Name: William     Last Name: Harris       Age: 31         daysInCourse: {20 30 35}        Degree Program: DATA_MANAGEMENT_DATA_ANALYTICS


Printing Invalid E-mails

John1989@gm ail.com is an invalid E-mail address and is associated with student ID#: A1
Erickson_1990@gmailcom is an invalid E-mail address and is associated with student ID#: A2
The_lawyer99yahoo.com is an invalid E-mail address and is associated with student ID#: A3



Printing Student with a specific First Name: William
ID  | First & Last Name|      e-Mail         | Age | Days To Complete | Degree

A5      William Harris  wharr79@my.wgu.edu      31      20,30,35,       DATA_MANAGEMENT_DATA_ANALYTICS


Average Days In Course:

A1 John Smith: 35 Average of days in course
A2 Suzan Erickson: 40 Average of days in course
A3 Jack Napoli: 31 Average of days in course
A4 Erin Black: 49.3333 Average of days in course
A5 William Harris: 28.3333 Average of days in course
```



```
                        STUDENTS BY DEGREE PROGRAM
ID| First Name| Last Name|      e-Mail         | Age | Days To Complete | Degree

A3      Jack    Napoli    The_lawyer99yahoo.com   19    {20, 40, 33}    SOFTWARE


Removing student associated with A3

ID  |      First Name       |      Last Name        |     Age      |      Days To Complete         |           Degree

A1      First Name: John        Last Name: Smith        Age: 20         daysInCourse: {30 35 40}        Degree Program: SECURITY
A2      First Name: Suzan       Last Name: Erickson     Age: 19         daysInCourse: {50 30 40}        Degree Program: NETWORK
Empty   First Name:     Last Name:      Age: 0          daysInCourse: {0 0 0}   Degree Program: UNDECLARED
A4      First Name: Erin        Last Name: Black        Age: 22         daysInCourse: {50 58 40}        Degree Program: SECURITY
A5      First Name: William     Last Name: Harris       Age: 31         daysInCourse: {20 30 35}        Degree Program: DATA_MANAGEMENT_DATA_ANALYTICS


Error: Student with A3 is not found!

Destroying Student Objects

Erasing Erasing Erasing Erasing Erasing SuccessDestroying Student Objects

Erasing
C:\Users\William Harris\source\repos\Class_Roster_Project_WMH\x64\Debug\Class_Roster_Project_WMH.exe (process 8704) exited with code -1073741819.
Press any key to close this window . . .
```

Project Requirements:

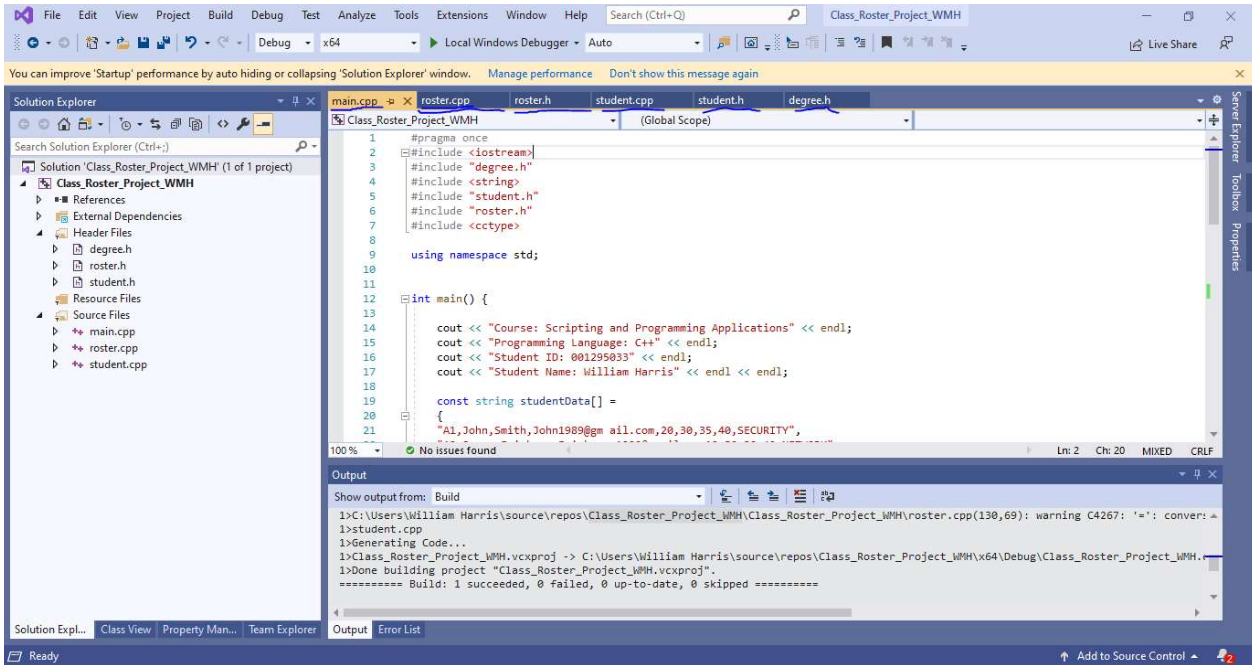
A. Modify the “studentData Table” to include your personal information as the last item.

My data is appended to the student data table.

William Harris
Student ID:001295033
Wednesday, October 21, 2020

B. Create a C++ project in your integrated development environment (IDE) with the following files:
- degree.h
- student.h and student.cpp
- roster.h and roster.cpp
- main.cpp

Note: There must be a total of six source code files.



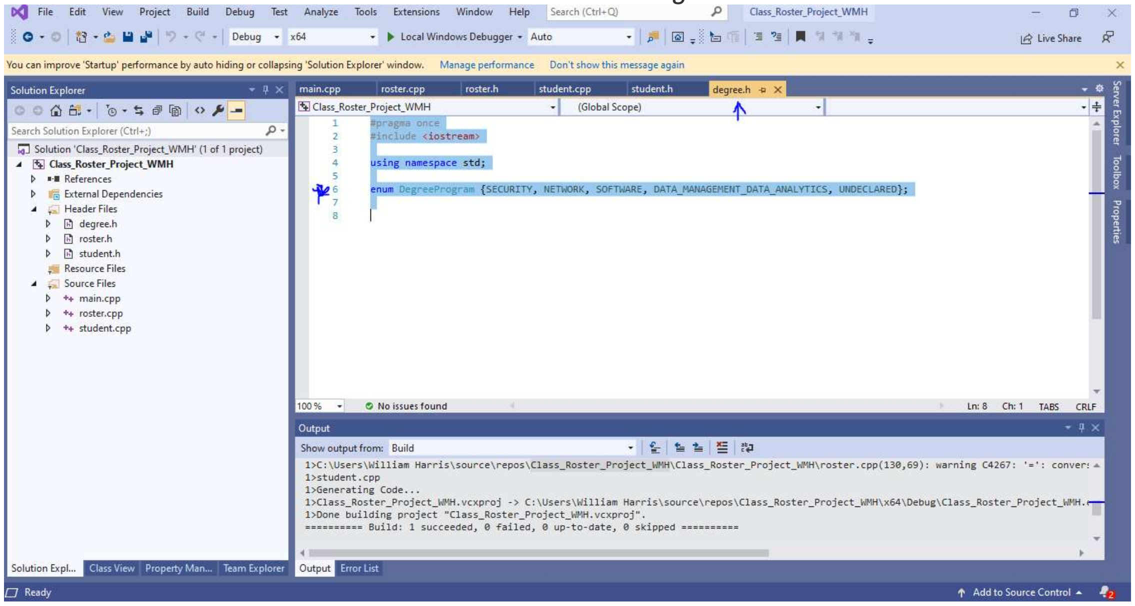
You can improve 'Startup' performance by auto hiding or collapsing 'Solution Explorer' window. Manage performance Don't show this message again

```
#pragma once
#include <iostream>
#include "degree.h"
#include <string>
#include "student.h"
#include "roster.h"
#include <cctype>

using namespace std;


int main() {

    cout << "Course: Scripting and Programming Applications" << endl;
    cout << "Programming Language: C++" << endl;
    cout << "Student ID: 001295033" << endl;
    cout << "Student Name: William Harris" << endl << endl;

    const string studentData[] =
    {
    "A1,John,Smith,John1989@gm ail.com,20,30,35,40,SECURITY",
```

```
1>C:\Users\William Harris\source\repos\Class_Roster_Project_WMH\Class_Roster_Project_WMH\roster.cpp(130,69): warning C4267: '=': conver:
1>student.cpp
1>Generating Code...
1>Class_Roster_Project_WMH.vcxproj -> C:\Users\William Harris\source\repos\Class_Roster_Project_WMH\x64\Debug\Class_Roster_Project_WMH.
1>Done building project "Class_Roster_Project_WMH.vcxproj".
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

C. Define an enumerated data type *DegreeProgram* for the degree programs containing the data type values *SECURITY*, *NETWORK*, and *SOFTWARE*.

Note: This information should be included in the degree.h file.



You can improve 'Startup' performance by auto hiding or collapsing 'Solution Explorer' window. Manage performance Don't show this message again

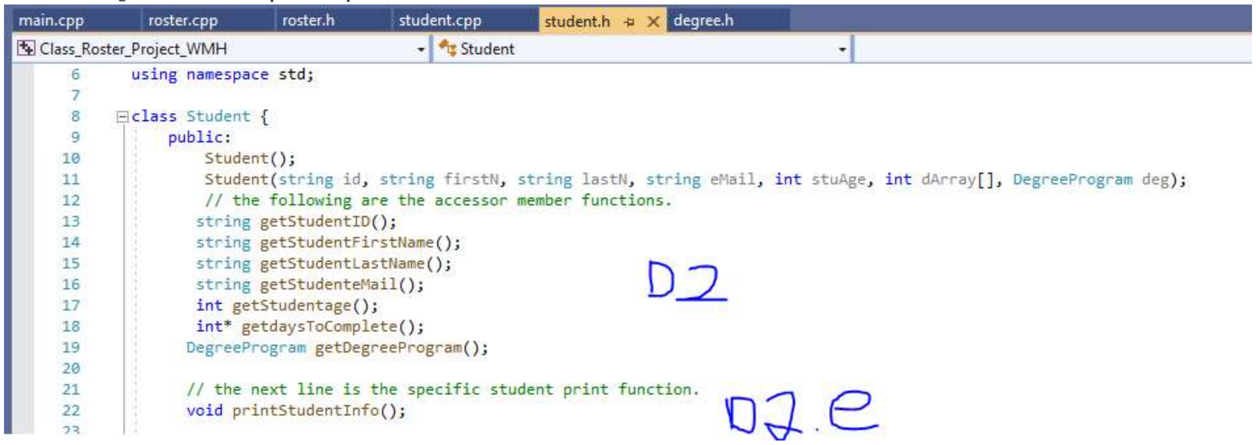
```
#pragma once
#include <iostream>

using namespace std;

enum DegreeProgram {SECURITY, NETWORK, SOFTWARE, DATA_MANAGEMENT_DATA_ANALYTICS, UNDECLARED};

```

```
1>C:\Users\William Harris\source\repos\Class_Roster_Project_WMH\Class_Roster_Project_WMH\roster.cpp(130,69): warning C4267: '=': conver:
1>student.cpp
1>Generating Code...
1>Class_Roster_Project_WMH.vcxproj -> C:\Users\William Harris\source\repos\Class_Roster_Project_WMH\x64\Debug\Class_Roster_Project_WMH.
1>Done building project "Class_Roster_Project_WMH.vcxproj".
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

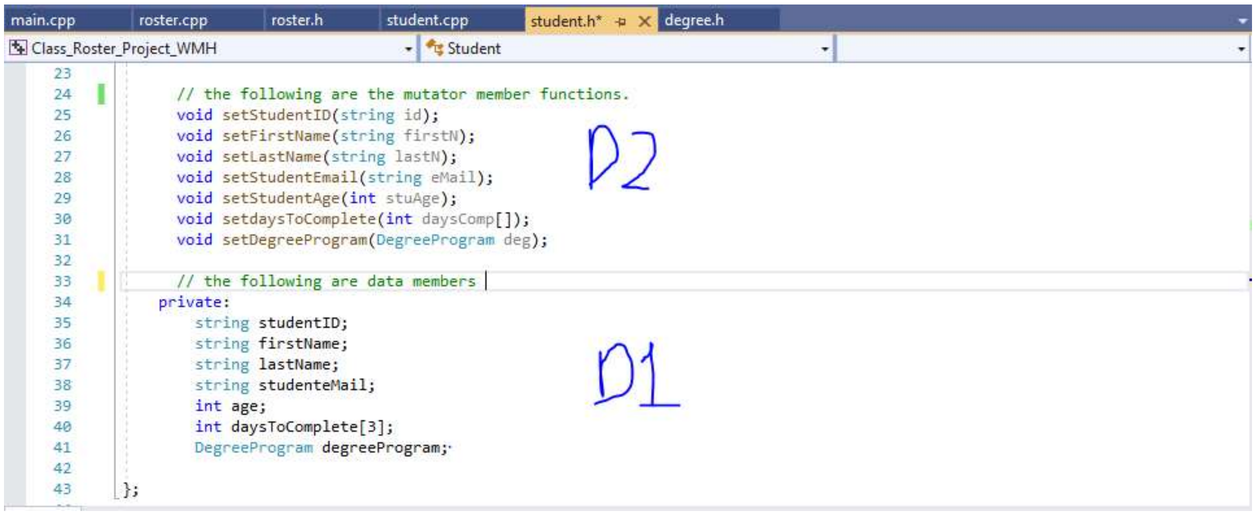
William Harris
Student ID:001295033
Wednesday, October 21, 2020

D. For the `Student` class, do the following:

1. Create the class `Student` in the files student.h and student.cpp, which includes *each* of the following variables:
   - student ID
   - first name
   - last name
   - email address
   - age
   - array of number of days to complete each course
   - degree program
2. Create *each* of the following functions in the `Student` class:
   a. an accessor (i.e., getter) for each instance variable from part D1
   b. a mutator (i.e., setter) for each instance variable from part D1
   c. All external access and changes to any instance variables of the `Student` class must be done using accessor and mutator functions.
   d. constructor using *all* of the input parameters provided in the table
   e. `print()` to print specific student data



```
using namespace std;

class Student {
    public:
        Student();
        Student(string id, string firstN, string lastN, string eMail, int stuAge, int dArray[], DegreeProgram deg);
        // the following are the accessor member functions.
       string getStudentID();
       string getStudentFirstName();
       string getStudentLastName();
       string getStudenteMail();
       int getStudentage();
       int* getdaysToComplete();
      DegreeProgram getDegreeProgram();

      // the next line is the specific student print function.
      void printStudentInfo();
```



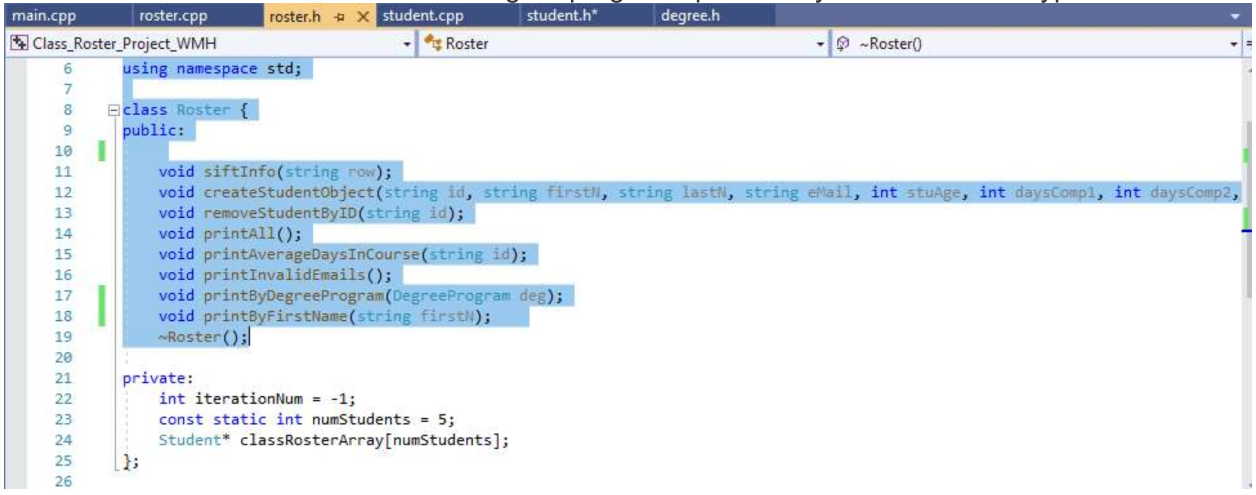
```
      // the following are the mutator member functions.
      void setStudentID(string id);
      void setFirstName(string firstN);
      void setLastName(string lastN);
      void setStudentEmail(string eMail);
      void setStudentAge(int stuAge);
      void setdaysToComplete(int daysComp[]);
      void setDegreeProgram(DegreeProgram deg);

      // the following are data members
    private:
        string studentID;
        string firstName;
        string lastName;
        string studenteMail;
        int age;
        int daysToComplete[3];
        DegreeProgram degreeProgram;

};
```

E. Create a `Roster` class (roster.cpp) by doing the following:

1. Create an array of pointers, `classRosterArray`, to hold the data provided in the “studentData Table.”
2. Create a student object for *each* student in the data table and populate `classRosterArray`.
    a. Parse *each* set of data identified in the “studentData Table.”
    b. Add *each* student object to `classRosterArray`.
3. Define the following functions:
    a. `public void add(string studentID, string firstName, string lastName, string emailAddress, int age, int daysInCourse1, int daysInCourse2, int daysInCourse3, DegreeProgram degreeprogram)` that sets the instance variables from part D1 and updates the roster.
    b. `public void remove(string studentID)` that removes students from the roster by student ID. If the student ID does not exist, the function prints an error message indicating that the student was not found.
    c. `public void printAll()` that prints a complete tab-separated list of student data in the provided format: `A1 [tab] First Name: John [tab] Last Name: Smith [tab] Age: 20 [tab]daysInCourse: {35, 40, 55} Degree Program: Security`. The `printAll()` function should loop through *all* the students in `classRosterArray` and call the print() function for *each* student.
    d. `public void printAverageDaysInCourse(string studentID)` that correctly prints a student’s average number of days in the three courses. The student is identified by the studentID parameter.
    e. `public void printInvalidEmails()` that verifies student email addresses and displays all invalid email addresses to the user.

    Note: A valid email should include an at sign ('@') and period ('.') and should not include a space (' ').

    f. `public void printByDegreeProgram(DegreeProgram degreeProgram)` that prints out student information for a degree program specified by an enumerated type.



```
using namespace std;

class Roster {
public:

    void siftInfo(string row);
    void createStudentObject(string id, string firstN, string lastN, string eMail, int stuAge, int daysComp1, int daysComp2,
    void removeStudentByID(string id);
    void printAll();
    void printAverageDaysInCourse(string id);
    void printInvalidEmails();
    void printByDegreeProgram(DegreeProgram deg);
    void printByFirstName(string firstN);
    ~Roster();

private:
    int iterationNum = -1;
    const static int numStudents = 5;
    Student* classRosterArray[numStudents];
};
```

F. Demonstrate the program's required functionality by adding a `main()` function in main.cpp, which will contain the required function calls to achieve the following results:

1. Print out to the screen, via your application, the course title, the programming language used, your WGU student ID, and your name.
2. Create an instance of the `Roster` class called `classRoster`.
3. Add *each* student to `classRoster`.
4. Convert the following pseudo code to complete the rest of the `main()` function:

```
classRoster.printAll();
classRoster.printInvalidEmails();

//loop through classRosterArray and for each element:
classRoster.printAverageDaysInCourse(/*current_object's student id*/);

classRoster.printByDegreeProgram(SOFTWARE);
classRoster.remove("A3");
classRoster.printAll();
classRoster.remove("A3");
//expected: the above line should print a message saying such a student
with this ID was not found.
```

5. Implement the destructor to release the memory that was allocated dynamically in `Roster`.

G. Demonstrate professional communication in the content and presentation of your submission.