DMA and Classes with resources

Workshop 9

In this workshop, you are to complete the code for the Transcript class containing several Subjects.

LEARNING OUTCOMES

Upon successful completion of this workshop, you will have demonstrated the abilities to

- Dynamic Memory reallocation and resizing
- Working with classes with resources

SUBMISSION POLICY

The “in-lab” section is to be completed **during your assigned lab section**. It is to be completed and submitted by the end of the workshop. If you do not attend the workshop, you can submit the “in-lab” section along with your “at-home” section (a 20% late deduction will be assessed). The “at-home” portion of the lab is **due the day before your next scheduled workshop**

All your work (all the files you create or modify) must contain your name, Seneca email and student number.

You are responsible to regularly backup your work.

IN LAB:

Download or clone workshop 8 from https://github.com/Seneca-244200/OOP-Workshop9

TRANSCRIPT CLASS:

Create a class called transcript that can hold and unknown number of Subjects and print them as a transcript, showing the average in GPA format.

Member Variables:

```
    char stName_[81];
```

Holds the name of the student.

```
    char stNo_[10];
```
Holds the student number of the student.

```
    Subject* subs_;
```
A dynamic array of Subjects.

```
    int noOfSubs_;
```
Number of Subjects in the dynamic array pointed by subs_.

Constructor and Member Functions

```
    Transcript(const char* stname, const char* stno);
```
Copies the name and student number to the corresponding member variables and then sets the subs_ pointer to null and noOfSubs_ to zero.

```
    Transcript& operator+=(const Subject& S);
```
Adds one to the elements of the dynamic array of Subjects pointed by subs_ and then sets the last element to the Subject argument "S". Make sure noOfSubs_ is increased by one.

```
    Subject& operator[](int index);
```
Returns the reference of the Subject kept in the dynamic array of Subjects pointed by subs_. If the index passed the noOfSubs_, it will go back to the beginning of the array. (i.e if there are 5 subjects, index 5 will be the same as index 0 and index 11 same as 1).
If noOfSubs_ is zero, then the result of invoking this operator is unknown.

```
    std::ostream& display(std::ostream& os)const;
```
Displays the transcript in the following format:

```
Student Name: John Doe
Student Number: 042942088
OOP244: ..................................................................... C
EAC150: ..................................................................... B
IBC233: ..................................................................... A
INT222: ..................................................................... B
DBS201: ..................................................................... C
                                                                    GPA: 2.8
0123456789012345678901234567890123456789012345678901234567890123456789012345678
         1         2         3         4         5         6         7
```

Note: if the displayType of a Mark object is GPA, then the operator+= automatically adds the values as GPA and not raw numbers.

```
    virtual ~Transcript();
Deallocates the memory allocated by subs_;

    int noOfSubjects()const;
```

Returns the number of Subjects pointed by sub_ pointer.

SUBMISSION

To test and demonstrate execution of your program use w9_in_lab.cpp.

If not on matrix already, upload **Mark.cpp, Mark.h, Subject.cpp, Subject.h, Transcript.cpp, Transcript.h** and **w9_in_lab.cpp** to your matrix account. Compile and run your code and make sure everything works properly.

Then run the following script from your account:

```
Section SCC and SDD:
~fardad.soleimanloo/submit w9_in_lab <ENTER>
```

and follow the instructions.

AT-HOME

Create copy constructor and overload operator= for Transcript class.

Add

ostream& read(ostream& ostr, Transcript& T).

and overload operator>> to be able to enter the information for a transcript from console using cin.

Then write an application that receives the information for a transcript and saves the printout in a file.

(User decides what the name of the file should be.)

Save your application in w9_at_home.cpp.

SUBMISSION

To test and demonstrate execution of your program use w9_at_home.cpp.

If not on matrix already, upload **Mark.cpp, Mark.h, Subject.cpp, Subject.h, Transcript.cpp, Transcript.h** and **w9_at_home.cpp** to your matrix account. Compile and run your code and make sure everything works properly.

Then run the following script from your account:

```
Section SCC and SDD:
~fardad.soleimanloo/submit w9_at_home <ENTER>
```

and follow the instructions.