Programming Assignment 01
Full Points: 100
Submission Deadline: 01/30/2017 (Monday, 11:59 pm)

Define a class in C++ that has the following properties:

```
class student{

     string name, id;
     float gpa;  // 0 <= gpa <= 4


     public:

     student(); //cons_0 ==> set default values with empty string and 0.00 as
gpa
     student(string, string, float); //cons_1
     student(const student&); //cost_2

     string get_name();
     string get_id();
     float get_gpa();
     void set_name(string);
     void set_id(string);
     void set_gpa(float);

 }
```

In your main function (should be declared in a file named
'program_01_yourUNTID.cpp')
you will create:

i) three objects of type: student (say, a, b, c).
ii) initialize 'a' with constructor cons_0,
initialize 'c' with constructor cons_1,
initialize 'b' with constructor cons_2, you can pass either a or c

iii) print all the properties of each object
iv) create an array of 5 objects. You can ask the user either
a) to insert the properties while constructing the new object
b) to use a predefined object (say, b) to construct the new object
(we will prefer a menu to handle such scenario)
v) print the name and id of the student with highest gpa out of the 5 students;

vi) you can add more member functions to the student class if you need to!
vii) your code should be able to detect invalid input errors (e.g. negative gpa, etc.)

Instructions::
a) Submit at least four files (student.h, student.cpp, program_01_yourUNTID.cpp, readme)

b) if you implement everything in a single file: -20 points

c) if your program does not compile/run in CSE machines: -100 points

d) if your program needs some special instructions to run/compile, add a readme file with
sufficient explanation and a few sample inputs and corresponding outputs. If no read me is found in such a case: -100 points

e) The grader should be able to input/change values of different properties of each object. So, create
a menu to do so. If you hard code the inputs: -50 points

f)

i) for implementing the class properly: +40 points
ii) for implementing the menu properly: +20 points
iii) add comments in proper places (at least before each function): +10 points
iv) for handling i/o errors (the user cannot insert numbers for names, negatives for gap, etc.): +20 points
v) for writing a README file with 3 sample input/output of the run-time scenario: +10 points

g) The rest of the points will be determined by the grader as he/she tests your code with different
input sets. Please remember that your code has to compile and run in CSE machines first to get the points.