



UNIVERSITY OF SANTO TOMAS

Engineering Pre-Major Year Collaborative (EPYC) Program

ENG 209:

Computer Fundamentals and Programming

Project

MACHINE PROBLEM: Project X

Students

Leader: Buscado, Son

Reviewer: Gokoico, Joshua

Developer: Santonia, Ivan Thaddeaus F

ENGR. MA. MADECHEEN S. PANGALIMAN, MSc. ECE

Professor

2nd Term

S.Y. 2018 – 2019

Table of Contents

A. MACHINE PROBLEM 1

INTRODUCTION3

PROGRAM SIMULATIONS (TESTING)4-7

B. MACHINE PROBLEM 2

INTRODUCTION8

PROGRAM SIMULATIONS (TESTING)8-9

C. APPENDIX

C++ CODES FOR MP110-18

C++ CODES FOR MP219-21

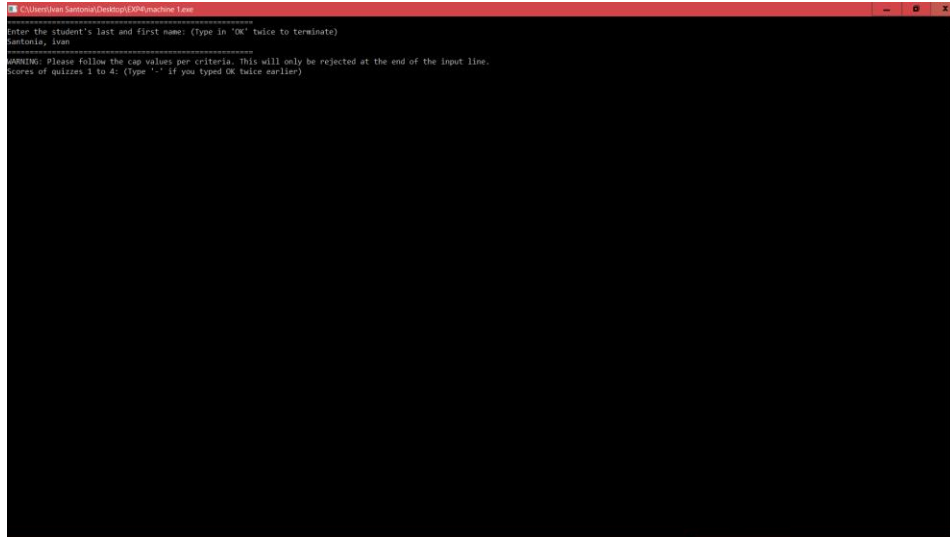
I. MACHINE PROBLEM 1

INTRODUCTION

- I. This program will let the user to input 50 student names and their respective scores for each grade component and students' raw grades will be computed according to the grade components
 - Quizzes = this component is composed of 4 quizzes.
 - Each quiz is equal to 100 points and has a weight of 15%. 60%
 - Major Examinations =
 - Preliminary and Final Examinations are also equal to 100 points each and has a weight of 20%. 40%
 - Class Standing =
 - This component is added on top of their final grades and is composed of the following:
 - *Notebook: = $\text{Grade}/100$ and is equivalent to 1% of class standing grade. 5%
 - Seat works: A total of 10 seat works were given, and each seat work is equivalent to 10 points each. Seat works are equivalent to 2% of class standing grade.
 - Board work: Board work points are given to those students who actively participate in recitations. The student who has the highest board work points will earn the 2% of the class standing grade.
- II. The program will output the students' final raw grades and their corresponding Grade Point Average based on the transmutation
- III. The program will determine if a student passed or failed the subject and will display the top 10 performing students in class.

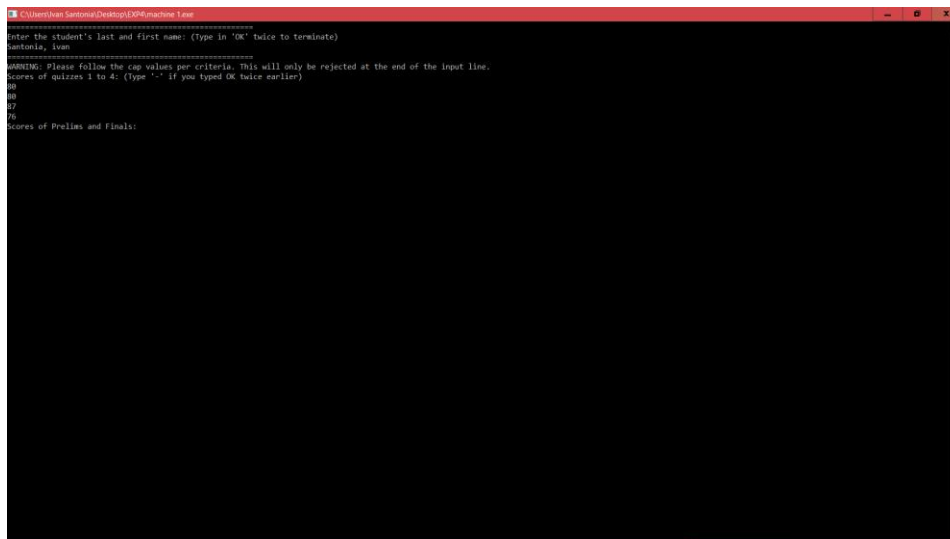
PROGRAM SIMULATIONS (TESTING)

- I. First the user will input the students full name



```
C:\Users\Ivan>python C:\Users\Ivan\Desktop\ENG209\machine 1.exe
=====
Enter the student's last and first name: (Type in 'OK' twice to terminate)
Sanjivla, Ivan
=====
WARNING: Please follow the cap values per criteria. This will only be rejected at the end of the input line.
Scores of quizzes 1 to 4: (Type '-' if you typed OK twice earlier)
```

- II. Second the user will input the raw scores of the student on the 4 quizzes and 1 quiz is equivalent to 100 points each



```
C:\Users\Ivan>python C:\Users\Ivan\Desktop\ENG209\machine 1.exe
=====
Enter the student's last and first name: (Type in 'OK' twice to terminate)
Sanjivla, Ivan
=====
WARNING: Please follow the cap values per criteria. This will only be rejected at the end of the input line.
Scores of quizzes 1 to 4: (Type '-' if you typed OK twice earlier)
88
89
87
76
Scores of Prelims and Finals:
```

Problem

- III. Third the user will Input the raw scores of the prelim and finals this is also over 100 points each

```

C:\Users\Ivan Santonia\Desktop\ENG209\machine1.exe
Enter the student's last and first name: (Type in 'OK' twice to terminate)
Santonia, Ivan
=====
WARNING: Please follow the cap values per criteria. This will only be rejected at the end of the input line.
Scores of quizzes 1 to 4: (Type '-' if you typed OK twice earlier)
88
87
76
Scores of Prelims and Finals:
87
85
Scores of Seatworks 1 to 10:

```

- IV. Next the user will input the raw scores of the 10 seatwork and each SW is equivalent to 10 points each

```

C:\Users\Ivan Santonia\Desktop\ENG209\machine1.exe
Enter the student's last and first name: (Type in 'OK' twice to terminate)
Santonia, Ivan
=====
WARNING: Please follow the cap values per criteria. This will only be rejected at the end of the input line.
Scores of quizzes 1 to 4: (Type '-' if you typed OK twice earlier)
88
87
76
Scores of Prelims and Finals:
87
85
Scores of Seatworks 1 to 10:

```

Problem

- V. Second to the last that the user will input the notebook score and it is over 100

```

C:\Antonina Antonina\Desktop\EXP\machine\Task
Enter the student's last and first name: (Type in 'OK' twice to terminate)
Antonina, Ivan
=====
WARNING: Please follow the cap values per criteria. This will only be rejected at the end of the input line.
Scores of quizzes 1 to 4: (Type '-' if you typed OK twice earlier)
0
0
0
0
0
Scores of Prelims and Finals:
0
0
Scores of Seatworks 1 to 10:
0
0
0
0
0
0
0
0
0
0
Notebook Grade:

```

- VI. Lastly the user will input the board work scores

```

C:\Antonina Antonina\Desktop\EXP\machine\Task
Enter the student's last and first name: (Type in 'OK' twice to terminate)
Antonina, Ivan
=====
WARNING: Please follow the cap values per criteria. This will only be rejected at the end of the input line.
Scores of quizzes 1 to 4: (Type '-' if you typed OK twice earlier)
0
0
0
0
0
Scores of Prelims and Finals:
0
0
Scores of Seatworks 1 to 10:
0
0
0
0
0
0
0
0
0
0
Notebook Grade:
0
Boardwork Grade:
=====
Enter the student's last and first name: (Type in 'OK' twice to terminate)

```

- VII. After this you need to repeat it 49 times to have a total of 50 student grades

Problem

VIII. The final output of the program will be consist of the ff:

- Top 10 students
- Students who failed
- Students who passed

```

C:\Users\apron\Desktop\Untitled2.exe
=====
The following students passed this subject:
A_ A - 2.5
A_ B - 2.5
A_ C - 2.5
A_ D - 2.5
A_ E - 2.5
A_ F - 2.5
A_ G - 2.5
A_ H - 2.5
A_ I - 2.5
A_ J - 2.5
A_ K - 1
A_ L - 2.5
A_ M - 2.5
A_ N - 2.5
A_ O - 2.5
=====
The following students are asked to seek counselling:
A_ P - 5
A_ Q - 5
A_ R - 5
A_ S - 5
A_ T - 5
A_ U - 5
A_ V - 5
=====
The following students show the most promise in the subject:
A_ W - 101
A_ X - 101
A_ Y - 101
A_ Z - 101
A_ A - 101
A_ B - 101
A_ C - 101
A_ D - 101
A_ E - 101
A_ F - 101
=====
Process exited after 179.4 seconds with return value 0
Press any key to continue . . .

```

B. MACHINE PROBLEM 2

INTRODUCTION

This is a program that would compute the following values based on the given sides of the triangle. Where the user will input length of the sides height and base length of the triangle and then the program will compute for all interior angles, classify whether scalene, isosceles, or equilateral, determine the area and perimeter., classify whether acute triangle, right triangle, or obtuse triangle. And compute for length of apothem and circumcenter.

PROGRAM SIMULATIONS (TESTING)

1. Opening the program you will see that the user needs to enter the x and y coordinate to find the 1st side of the triangle



2. Right after the user also needs to input the 2nd and 3rd x and y coordinates to have a total of 3 sides



Problem

- After the user had input the coordinates the program will classify the triangle whether it is scalene, isosceles, or equilateral.

```

C:\Users\Student\Documents>g++ Triangle.cpp
interior Angle: 100
Enter coordinates of the triangle.
Enter (x1, y1): 45
45
Enter (x2, y2): 76
43
Enter (x3, y3): 67
54
The triangle is scalene.
Enter base and height:

```

- Then the user will input the data for height and base of the triangle after inputting the length of the base and height the program will give you the area and the perimeter of the triangle.

```

interior Angle: 100
Enter coordinates of the triangle.
Enter (x1, y1): 45
45
Enter (x2, y2): 76
43
Enter (x3, y3): 67
54
The triangle is scalene.
Enter base and height: 54
54
Area: 1755 squared units.
Perimeter: 76.8226 units.
It is classified as acute triangle.
Enter opposite and adjacent side:

```

- Lastly the program will ask you to input the length of the opposite and adjacent side of the triangle after that the program will give you the length of apothem and circumcenter.

```

C:\Users\Student\Documents>g++ Triangle.cpp
interior Angle: 100
Enter coordinates of the triangle.
Enter (x1, y1): 45
45
Enter (x2, y2): 76
43
Enter (x3, y3): 67
54
The triangle is scalene.
Enter base and height: 54
54
Area: 1755 squared units.
Perimeter: 76.8226 units.
It is classified as acute triangle.
Enter opposite and adjacent side: 45
76
Apotham: 8.18765
Circumcircle: 46.4646
.....
Process exited after 100 seconds with return value 0
Press any key to continue . . .

```

C. APPENDICES

3.1. C++ PROGRAM CODES FOR MP1

The codes used to create a C++ program for MP1 are as follows:

```
//Machine
Problem 1

// Created by: SANTONIA Ivan & GOKIOCO Joshua
//Begin Date: TUESDAY-21-05-2019, file created at 2156 hours (9:56 PM).
//Completion: SATURDAY-25-05-2019, final prototype completed at
#include <iostream>
#include <string>
using namespace std;
int main()
{
    //Input for Quizzes (qx), Exams (ex), Seatworks (sx), Notebook (nb),
    and Boardwork (bw).
    int q1, q2, q3, q4, e1, e2, s1, s2, s3, s4, s5, s6, s7, s8, s9, s10, nb, bw;

    //Name Initializers.
    string lname;
    string fname;

    //Name Storage Arrays.
    string studentLName[50];
    string studentFName[50];

    //Placeholder (counter for initialPhase, x for betaPhase) counters.
    int x, z;
    int counter = 0;

    //Leaderboard (totalled highests, top students' first and last names)
    Storage Arrays.
    int htote[10] = {0};
    string sLNtop[10];
    string sFNtop[10];

    //Function Results (quiz-, exam-, and seatwork-averages; notebook's
    average; total overall), Percentage (totalled) Storage Array.
    float qave, eave, save, nbsa, tot;
```

Problem

```

float tote[50] = {0};

//Grade Point Average Discriminator (line = borderline), Grade Point
Average (fweighted = final weighted) Storage Array.
float line = 3.00;
float fweighted[50] = {0};
//initialPhase - input of raw data and names.
while (fname!= "OK"){
    cout <<
    "===== " <<
endl;

    cout << "Enter the student's last and first name: (Type in 'OK'
twice to terminate)" << endl;
    cin >> lname >> fname;
    //Name Input.
    studentLName[counter] = lname;
    studentFName[counter] = fname;

    cout <<
    "===== " <<
endl;

    //Data Input.
    cout << "WARNING: Please follow the cap values per criteria.
This will only be rejected at the end of the input line." << endl;
    cout << "Scores of quizzes 1 to 4: (Type '-' if you typed OK
twice earlier)" << endl;
    cin >> q1 >> q2 >> q3 >> q4;
    //^^^Quizzes.

    cout << "Scores of Prelims and Finals: " << endl;
    cin >> e1 >> e2;
    //^^^Exams.

    cout << "Scores of Seatworks 1 to 10: " << endl;
    cin >> s1 >> s2 >> s3 >> s4 >> s5 >> s6 >> s7 >> s8 >> s9 >>
s10;
    //^^^Seatworks.

    cout << "Notebook Grade: " << endl;
    cin >> nb;
    //^^^Notebook data.

```

Problem

```

        cout << "Boardwork Grade: " << endl;
    cin >> bw;
    //^Boardwork data.

    //alphaPhase - data computation and checking of states.
    {
        qave = ((q1 + q2 + q3 + q4)/4) * 0.60;
        eave = ((e1 + e2)/2) * 0.40;
        save = ((s1 + s2 + s3 + s4 + s5 + s6 + s7 + s8 + s9 + s10)/10) * 0.02;
        nbsa = (nb/100);
        tot = qave + eave + save + nbsa;
        tote[counter] = tot;
    }

    //This transmutes percentages to Grade Point Averages.
    //NOTE: The maximum is supposedly 100, but the raw
percentage accounts to 105.
        {if (tote[counter] >= 95.57){
            fweighted[counter] = 1.00;
            counter++;}
            else if (tote[counter] >= 91.12 && tote[counter] <= 95.57){
                fweighted[counter] = 1.25;
                counter++;}
                else if (tote[counter] >= 86.68 && tote[counter] <= 91.11){
                    fweighted[counter] = 1.50;
                    counter++;}
                    else if (tote[counter] >= 82.23 && tote[counter] <= 86.67){
                        fweighted[counter] = 1.75;
                        counter++;}
                        else if (tote[counter] >= 77.79 && tote[counter] <= 82.22){
                            fweighted[counter] = 2.00;
                            counter++;}
                            else if (tote[counter] >= 73.34 && tote[counter] <= 77.78){
                                fweighted[counter] = 2.25;
                                counter++;}
                                else if (tote[counter] >= 68.90 && tote[counter] <= 73.33){
                                    fweighted[counter] = 2.50;
                                    counter++;}
                                    else if (tote[counter] >= 64.45 && tote[counter] <= 68.89){
                                        fweighted[counter] = 2.75;
                                        counter++;}
                                        else if (tote[counter] >= 60.00 && tote[counter] <= 64.44){

```

Problem

```

    fweighted[counter] = 3.00;
    counter++;}
//Else If instead of Else for clarity. This is arbitrary and can be
replaced by an Else statement.
    else if (tote[counter] <= 59.99){
        fweighted[counter] = 5.00;
        counter++;}
    }

//betaPhase - analyses and updates leaderboards.
for (x = 0; x < counter; x++){
    //All succeeding If statements update the leaderboards by a
certain degree, depending on what place is being replaced.
    if(tote[x] > htote[0]){
        htote [9] = htote [8];
        sLNtop [9] = sLNtop [8];
        sFNtop [9] = sFNtop [8];
        htote [8] = htote [7];
        sLNtop [8] = sLNtop [7];
        sFNtop [8] = sFNtop [7];
        htote [7] = htote [6];
        sLNtop [7] = sLNtop [6];
        sFNtop [7] = sFNtop [6];
        htote [6] = htote [5];
        sLNtop [6] = sLNtop [5];
        sFNtop [6] = sFNtop [5];
        htote [5] = htote [4];
        sLNtop [5] = sLNtop [4];
        sFNtop [5] = sFNtop [4];
        htote [4] = htote [3];
        sLNtop [4] = sLNtop [3];
        sFNtop [4] = sFNtop [3];
        htote [3] = htote [2];
        sLNtop [3] = sLNtop [2];
        sFNtop [3] = sFNtop [2];
        htote [2] = htote [1];
        sLNtop [2] = sLNtop [1];
        sFNtop [2] = sFNtop [1];
        htote[1] = htote [0];
        sLNtop[1] = sLNtop [0];
        sFNtop[1] = sFNtop [0];
        //This updates the 1st position or the highest.
        htote[0] = tote[x];

```

Problem

```

sLNtop[0] = studentLName[x];
sFNtop[0] = studentFName[x];
}

if(tote[x] > htote[1] && tote[x] < htote[0]){
    htote [9] = htote [8];
    sLNtop [9] = sLNtop [8];
    sFNtop [9] = sFNtop [8];
    htote [8] = htote [7];
    sLNtop [8] = sLNtop [7];
    sFNtop [8] = sFNtop [7];
    htote [7] = htote [6];
    sLNtop [7] = sLNtop [6];
    sFNtop [7] = sFNtop [6];
    htote [6] = htote [5];
    sLNtop [6] = sLNtop [5];
    sFNtop [6] = sFNtop [5];
    htote [5] = htote [4];
    sLNtop [5] = sLNtop [4];
    sFNtop [5] = sFNtop [4];
    htote [4] = htote [3];
    sLNtop [4] = sLNtop [3];
    sFNtop [4] = sFNtop [3];
    htote [3] = htote [2];
    sLNtop [3] = sLNtop [2];
    sFNtop [3] = sFNtop [2];
    htote [2] = htote [1];
    sLNtop [2] = sLNtop [1];
    sFNtop [2] = sFNtop [1];
    //This updates the 2nd highest.
    htote[1] = tote[x];
    sLNtop[1] = studentLName[x];
    sFNtop[1] = studentFName[x];
}

if(tote[x] > htote[2] && tote[x] < htote[1]){
    htote [9] = htote [8];
    sLNtop [9] = sLNtop [8];
    sFNtop [9] = sFNtop [8];
    htote [8] = htote [7];
    sLNtop [8] = sLNtop [7];
    sFNtop [8] = sFNtop [7];
    htote [7] = htote [6];

```

Problem

```

sLNTop [7] = sLNTop [6];
sFNtop [7] = sFNtop [6];
htote [6] = htote [5];
sLNTop [6] = sLNTop [5];
sFNtop [6] = sFNtop [5];
htote [5] = htote [4];
sLNTop [5] = sLNTop [4];
sFNtop [5] = sFNtop [4];
htote [4] = htote [3];
sLNTop [4] = sLNTop [3];
sFNtop [4] = sFNtop [3];
htote [3] = htote [2];
sLNTop [3] = sLNTop [2];
sFNtop [3] = sFNtop [2];
        //This updates the 3rd highest.
        htote[2] = tote[x];
sLNTop[2] = studentLName[x];
sFNtop[2] = studentFName[x];
    }
    if(tote[x] > htote[3] && tote[x] < htote[2]){
        htote [9] = htote [8];
sLNTop [9] = sLNTop [8];
sFNtop [9] = sFNtop [8];
htote [8] = htote [7];
sLNTop [8] = sLNTop [7];
sFNtop [8] = sFNtop [7];
htote [7] = htote [6];
sLNTop [7] = sLNTop [6];
sFNtop [7] = sFNtop [6];
htote [6] = htote [5];
sLNTop [6] = sLNTop [5];
sFNtop [6] = sFNtop [5];
htote [5] = htote [4];
sLNTop [5] = sLNTop [4];
sFNtop [5] = sFNtop [4];
htote [4] = htote [3];
sLNTop [4] = sLNTop [3];
sFNtop [4] = sFNtop [3];
        //This updates the 4th highest.
        htote[3] = tote[x];
sLNTop[3] = studentLName[x];
sFNtop[3] = studentFName[x];
    }

```

Problem

```

if(tote[x] > htote[4] && tote[x] < htote[3]){
    htote [9] = htote [8];
    sLNtop [9] = sLNtop [8];
    sFNtop [9] = sFNtop [8];
    htote [8] = htote [7];
    sLNtop [8] = sLNtop [7];
    sFNtop [8] = sFNtop [7];
    htote [7] = htote [6];
    sLNtop [7] = sLNtop [6];
    sFNtop [7] = sFNtop [6];
    htote [6] = htote [5];
    sLNtop [6] = sLNtop [5];
    sFNtop [6] = sFNtop [5];
    htote [5] = htote [4];
    sLNtop [5] = sLNtop [4];
    sFNtop [5] = sFNtop [4];
    //This updates the 5th highest.
    htote[4] = tote[x];
    sLNtop[4] = studentLName[x];
    sFNtop[4] = studentFName[x];
}

    if(tote[x] > htote[5] && tote[x] < htote[4]){
        htote [9] = htote [8];
        sLNtop [9] = sLNtop [8];
        sFNtop [9] = sFNtop [8];
        htote [8] = htote [7];
        sLNtop [8] = sLNtop [7];
        sFNtop [8] = sFNtop [7];
        htote [7] = htote [6];
        sLNtop [7] = sLNtop [6];
        sFNtop [7] = sFNtop [6];
        htote [6] = htote [5];
        sLNtop [6] = sLNtop [5];
        sFNtop [6] = sFNtop [5];
        //This updates the 6th highest.
        htote[5] = tote[x];
        sLNtop[5] = studentLName[x];
        sFNtop[5] = studentFName[x];
    }

    if(tote[x] > htote[6] && tote[x] < htote[5]){

```


Problem

```

    htote [9] = htote [8];
    sLNtop [9] = sLNtop [8];
    sFNtop [9] = sFNtop [8];
    htote [8] = htote [7];
    sLNtop [8] = sLNtop [7];
    sFNtop [8] = sFNtop [7];
    htote [7] = htote [6];
    sLNtop [7] = sLNtop [6];
    sFNtop [7] = sFNtop [6];
        //This updates the 7th highest.
        htote[6] = tote[x];
    sLNtop[6] = studentLName[x];
    sFNtop[6] = studentFName[x];
    }

    if(tote[x] > htote[7] && tote[x] < htote[6]){
        htote [9] = htote [8];
        sLNtop [9] = sLNtop [8];
        sFNtop [9] = sFNtop [8];
        htote [8] = htote [7];
        sLNtop [8] = sLNtop [7];
        sFNtop [8] = sFNtop [7];
            //This updates the 8th highest.
            htote[7] = tote[x];
        sLNtop[7] = studentLName[x];
        sFNtop[7] = studentFName[x];
    }

    if(tote[x] > htote[8] && tote[x] < htote[7]){
        htote [9] = htote [8];
        sLNtop [9] = sLNtop [8];
        sFNtop [9] = sFNtop [8];
            //This updates the 9th highest.
            htote[8] = tote[x];
        sLNtop[8] = studentLName[x];
        sFNtop[8] = studentFName[x];
    }

    if(tote[x] > htote[9] && tote[x] < htote[8]){
        //This updates the 10th highest.
        htote[9] = tote[x];
        sLNtop[9] = studentLName[x];
        sFNtop[9] = studentFName[x];
    }

```

Problem

```

    }
    }
}

cout <<
"===== " <<
endl;

//This displays the names of students scoring approximately 3.0 to 1.0.
cout << "The following students passed this subject:" << "\n";
for (x = 0; x <= counter; x++)
{
    if(fweighted[x] <= line && fweighted[x] > 0)
    {
        cout << studentLName[x] << ", " << studentFName[x] << " - " <<
fweighted[x] << "\n";
    }
}
cout <<
"===== " <<
endl;
//This displays the names of students scoring all outside of 1.0 to 3.0, with
0 excluded.
cout << "The following students are asked to seek counselling: " << "\n";
for (x = 0; x < counter; x++)
{
    if(fweighted[x] > line)
    {
        cout << studentLName[x] << ", " << studentFName[x] <<
" - " << fweighted[x] << "\n";
    }
}
cout <<
"===== " <<
endl;
//This displays the names of students scoring the highest.
cout << "The following students show the most promise in the subject: "
<< "\n";
for (z = 0; z < 10; z++)
{
    cout << sLNtop[z] << ", " << sFNtop[z] << " - " << htote[z] << "\n";
}
}

```

3.2 C++ PROGRAM CODES FOR MP2

The codes used to create a C++ program for MP2 are as follows:

```
// Created by: BUSCADO, Son
#include <iostream>
#include <math.h>

using namespace std;

int main(){
    int x1, y1, x2, y2, x3, y3, n = 3, intAngle;
    float a, b, c, area, perimeter, height, base, opp, adj, tangent, apothem, circumcircle;
    double angle1, angle2, angle3;

    intAngle = ((n-2)*180); //formula to find the interior angle
    cout << "Interior Angle: " << intAngle << endl;

    cout << "Enter coordinates of the triangle.";
    cout << "\nEnter (x1, y1): ";
    cin >> x1 >> y1; // The user will input the first point (p1).
    cout << "\nEnter (x2, y2): ";
    cin >> x2 >> y2; // The user will input the second point (p2).
    cout << "\nEnter (x3, y3): ";
    cin >> x3 >> y3; // The user will input the third point (p3).

    a = sqrt(pow((x1-x2),2.0)+pow((y1-y2),2.0)); //formula to find the length or distance
    between p1 and p2
    b = sqrt(pow((x2-x3),2.0)+pow((y2-y3),2.0)); //formula to find the length or distance
    between p2 and p3
    c = sqrt(pow((x3-x1),2.0)+pow((y3-y1),2.0)); //formula to find the length or distance
    between p3 and p1
    // a, b, c will be used to find the perimeter and to classify the triangle.

    //Classifying the triangle:
    // all sides are equal
    if ((a==b)&&(b==c)){
        cout << "The triangle is equilateral." << endl;
```

Problem

```

}

// atleast 2 sides are equal
else if ((a==b)|| (b==c)){
    cout << "The triangle is isosceles." << endl;
}

// no sides are equal
else{
    cout << "The triangle is scalene." << endl;
}

//Determining the area and perimeter of triangle:
cout << "\nEnter base and height: ";
cin >> base >> height; /* The user will input the base and height of triangle.
These will be used to find the area of the triangle. */

area = 0.5*base*height; // formula to find the area of the triangle
perimeter = a + b + c; // formula to find the perimeter of the triangle

cout << "Area: " << area << " squared units." << endl;
cout << "Perimeter: " << perimeter << " units." << endl;

//Classifying the type of triangle based on the angle:
if (pow(a,2)+pow(b,2) > pow(c,2)){ //formula used to determine if it is an acute triangle
by its side lengths
    cout << "It is classified as acute triangle." << endl;
}
else if (pow(a,2)+pow(b,2) == pow(c,2)){ //formula used to determine if it is an right
triangle by its side lengths
    cout << "It is classified as right triangle." << endl;
}
else{
    cout << "It is classified as obtuse triangle." << endl;
}

cout << "\nEnter opposite and adjacent side: ";
cin >> opp >> adj; /* The user will input the opposite and adjacent side of triangle.
These will be necessary to find the tangent which will be used in finding the apothem.*/

```

```
tangent = opp/adj; //tangent will be used to find the apothem
tangent = pow(tangent,-1.0); //formula to convert the tangent to degrees
apothem = a/(2*tangent*(180/n)); //formula to find the apothem
circumcircle = (a*b*c)/sqrt((a+b+c)*(b+c-a)*(c+a-b)*(a+b-c)); //formula to find the
circumcircle

cout << "Apothem: " <<apothem <<endl;
cout << "Circumcircle: " <<circumcircle <<endl;

return 0;
}
```