**PROBLEM STATEMENT**

This program will read in grades from a txt file and will calculate the average grade of each student. It will also sort the average grades in ascending order and display the lowest and highest average in the class. Lastly, it will show the class average of each assignment.

**Output Display**

A (90-100) \*\*\*\*

B (80-89) \*\*\*\*\*\*\*\*\*\*

C (70-79) \*\*\*\*\*\*

D (60-69) \*\*\*

F (0-59) \*\*

Lowest score in the class: 56.3  
Highest score in the class: 93.4  
Number of students: 25

Exam01 Average: 77.4  
Exam02 Average: 78.2  
Exam03 Average: 80.8  
Exam04 Average: 78.4  
Final Exam Average: 82.2

Labs Average: 78.7  
Quizzes Average: 79.2

Structure Chart

Main

Average  
Functions x7

Sort

Histogram

Flowchart for Main

Start

Grades Array

Structure

Averages Array

Call Histogram Function

End

Display Highest and lowest score and averages

Call Average Functions (seven functions)

Call Sort Function

Pseudo Code for Main

Start of Algorithm for Main

1. Struct Grades  
   - int studentId; double exam01, exam02, exam03, exam04, finalExam, lab01, lab02, lab03, lab04, lab05, quizAverage;
2. Grades array[ARRAY\_SIZE]  
   - while (count < ARRAY\_SIZE)  
   { inputFile >> array[count].studentID; inputFile >> array[count].exam01; inputFile >> array[count].exam02; inputFile >> array[count].exam03; inputFile >> array[count].exam04; inputFile >> array[count].finalExam; inputFile >> array[count].lab01; inputFile >> array[count].lab02; inputFile >> array[count].lab03; inputFile >> array[count].lab04; inputFile >> array[count].lab05; inputFile >> array[count].quizAverage; count++ }
3. Double averageArray[ARRAY\_SIZE]  
   - for ( count = 0; count < ARRAY\_SIZE; count++)  
   { averageArray[count] = (array[count].exam01 + array[count].exam02 + array[count].exam03 + array[count].exam04 + array[count].finalExam + array[count].lab01 + array[count].lab02 + array[count].lab03 + array[count].lab04 + array[count].lab05 + array[count].quizAverage) / 11; }
4. Histogram(averageArray, count, ARRAY\_SIZE);
5. Sort(averageArray, count, ARRAY\_SIZE);
6. Call the seven average functions
7. Display Scores  
   -cout << “Lowest score: ” << averageArray[0] << endl;  
   -cout << “Highest score: ” << averageArray[24] << endl;  
   -cout << “Number of students: “ << count << endl;  
   - Display assignment averages

End of Algorithm for Main

Flowchart for Histogram

Start

“A (90-100)”  
for(count = 0; count < ARRAY\_SIZE; count++)

If (averageArray[count] >= 90 && averageArray[count] <= 100)

Cout << “\*”

End

If (averageArray[count] >= 0 && averageArray[count] < 60)

If (averageArray[count] >= 60 && averageArray[count] < 70)

If (averageArray[count] >= 70 && averageArray[count] < 80)

If (averageArray[count] >= 80 && averageArray[count] < 90)

Cout << “\*”

“F (0-59)”  
for(count = 0; count < ARRAY\_SIZE; count++)

Cout << “\*”

“D (60-69)”  
for(count = 0; count < ARRAY\_SIZE; count++)

“B (80-89)”  
for(count = 0; count < ARRAY\_SIZE; count++)

Cout << “\*”

“C (70-79)”  
for(count = 0; count < ARRAY\_SIZE; count++)

Cout << “\*”

Pseudo Code for Histogram

Start of Algorithm for Histogram

1. A (90-100)  
   - for (count = 0; count < ARRAY\_SIZE; count++)  
   { if (averageArray[count] >= 90 && averageArray <= 100)  
    cout << “\*”; }
2. B (80-89)  
   - for (count = 0; count < ARRAY\_SIZE; count++)  
   { if (averageArray[count] >= 80 && averageArray < 90)  
    cout << ”\*”; }
3. C (70-79)  
   - for (count = 0; count < ARRAY\_SIZE; count++)  
   { if (averageArray[count] >= 70 && averageArray < 80)  
    cout << “\*”; }
4. D (60-69)  
   - for (count = 0; count < ARRAY\_SIZE; count++)  
   { if(averageArray[count] >= 60 && averageArray < 70)  
    cout << “\*”; }
5. F (0-59)  
   - for (count = 0; count < ARRAY\_SIZE; count++)  
   { if(averageArray[count] >= 0 && averageArray < 60)  
    cout << “\*”; }

End of Algorithm for Histogram

Flowchart for Sort

Start

Do

End

While (swap)

Swap = true

averageArray[count+1] = temp

averageArray[count] = averageArray[count+1]

temp = averageArray[count]

If(averageArray[count] > averageArray[count+1]

Swap = false

For ( count = 0; count < (ARRAY\_SIZE – 1); count++

Pseudo Code for Sort

Start of Algorithm for Sort

1. Do  
   {swap = false  
    for (count = 0; count < (ARRAY\_SIZE – 1); count++)  
    { if (averageArray[count] > averageArray[count + 1])  
    { temp = averageArray[count]  
    averageArray[count] > averageArray[count + 1]  
    averageArray[count+ 1] = temp  
    swap = true }  
    }  
   }
2. While (swap)

End of Algorithm for Sort

Flowchart for Average Functions

Start

sumAssignment = 0.0

sumAssignment += array[count].assignment

For ( count = 0; count < ARRAY\_SIZE; count++

Return (sumAssignment / 25.0)

End

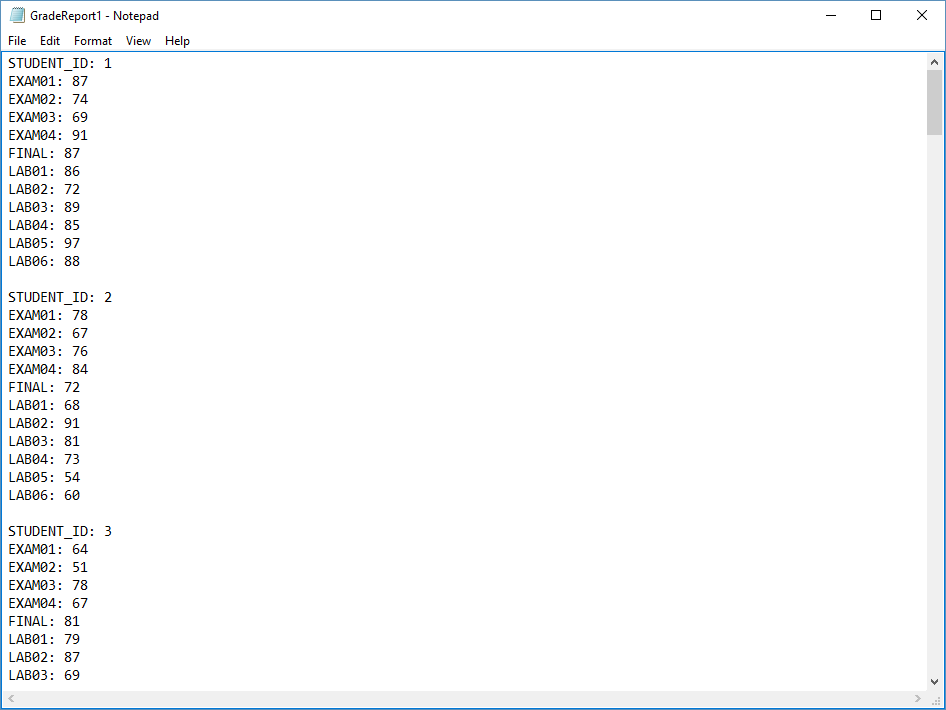
Pseudo Code for Average Functions

Start of Algorithm for Average Functions

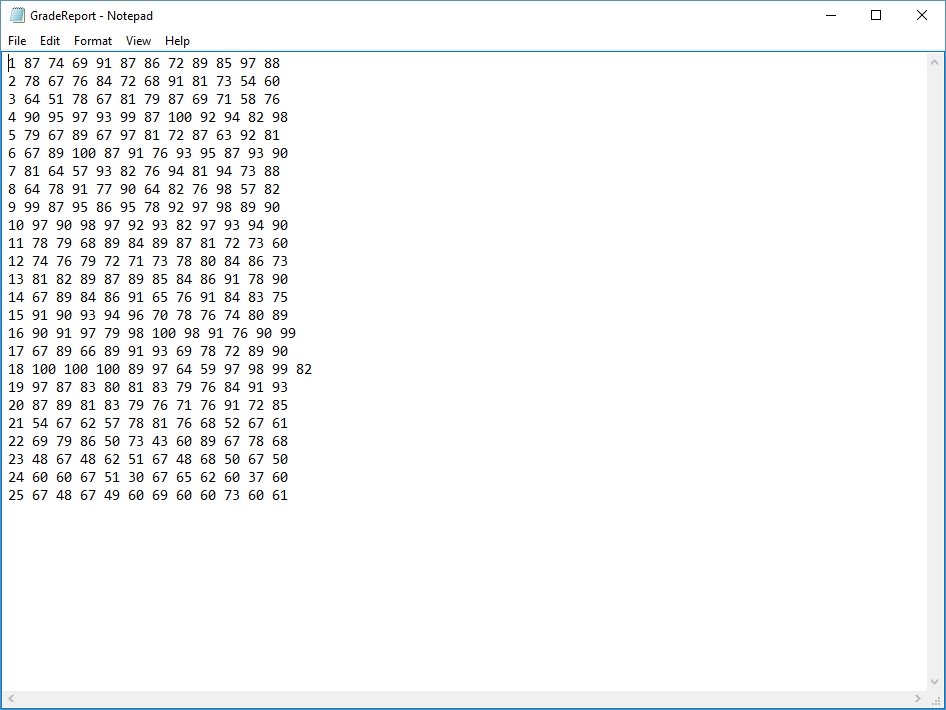
1. Double sumAssignment = 0.0
2. For (count = 0; count < ARRAY\_SIZE; count++)  
   -sumAssignment =+ array[count].assignment
3. Return (sumAssignment / 25.0) // For lab average divide by 125.0

End of Algorithm for Average Functions

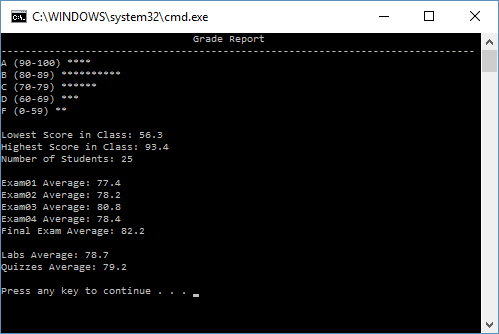
**INPUT**



**INPUT**



**OUTPUT**



User Instructions

This program is designed to read in grades from a txt and calculate a numerous amount of things. Its functions are: calculating the average of each student, sorting the averages of the students in ascending order, display a histogram of the amount of students that made a certain letter grade, displaying the highest and lowest scores in the class, and it will display the amount of students in the class. To use the program, simply create a txt file and enter the grades for each student. Finally, place the txt file in the project folder and run the program.

Comments

I must admit, I was expecting to have a lot more problems with this lab then what I actually had. I thought that reading the data from the txt file in to an array in my program was going to be more of a pain, but I managed to figure it out way quicker than what I expected. I actually cancelled some of my plans for the day because I expected to have so much trouble with that when actually I really didn’t need to cancel my plans at all! It was really cool implementing everything we have learned this year in to one final lab. This lab was probably my favorite because there was the least amount of help and the most amount of the different elements of programming we have learned. I definitely felt the most proud after finishing this lab than any other. Scrolling through the almost 300 lines of code and knowing that all of that was your work and came out of your brain makes you feel really good and accomplished. I’m looking forward to using these newly learned skills in the future.