



## CptS 121 - Program Design and Development



### Programming Assignment 3: Statistical Analysis of Student Records

**Assigned:** Wednesday, September 16, 2020

**Due:** Wednesday, September 23, 2020 by midnight

#### I. Learner Objectives:

At the conclusion of this programming assignment, participants should be able to:

- Open and close files
- Read, write to, and update files
- Manipulate file handles
- Apply standard library functions: `fopen ()`, `fclose ()`, `fscanf ()`, and `fprintf ()`
- Compose decision statements ("if" conditional statements)
- Create and utilize compound conditions

#### II. Prerequisites:

Before starting this programming assignment, participants should be able to:

- Analyze a basic set of requirements and apply top-down design principles for a problem
- Customize and define C functions
- Apply the 3 file format: 1 header file and 2 source files
- Document and comment a modular C program according to class standards
- Implement guard code in a header file
- Summarize topics from Hanly & Koffman Chapter 4 including:
  - What is a selection or conditional statement?
  - What is a compound condition?
  - What is a Boolean expression?
  - What is a flowchart?

#### III. Overview & Requirements:

Write a program that processes numbers, corresponding to student records read in from a file, and writes the required results to an output file (see `main ( )`). Your program should define the following functions:



(5 pts) `double read_double (FILE *infile)` — Reads one double precision number from the input file.  
Note: You may assume that the file only contains real numbers.



(5 pts) `int read_integer (FILE *infile)` - Reads one integer number from the input file.



(5 pts) `double calculate_sum (double number1, double number2, double number3, double number4, double number5)` - Finds the sum of *number1*, *number2*, *number3*, *number4*, and *number5* and returns the result.



(5 pts) `double calculate_mean (double sum, int number)` - Determines the mean through the calculation  $sum / number$  and returns the result. You need to check to make sure that *number* is not 0.

it is 0 the function returns -1.0 (we will assume that we are calculating the mean of positive numbers), otherwise it returns the mean.



(5 pts) double calculate\_deviation (double number, double mean) - Determines the deviation of *number* from the *mean* and returns the result. The deviation may be calculated as *number - mean*.



(10 pts) double calculate\_variance (double deviation1, double deviation2, double deviation3, double deviation4, double deviation5, int number) - Determines the variance through the calculation:  $((deviation1)^2 + (deviation2)^2 + (deviation3)^2 + (deviation4)^2 + (deviation5)^2) / number$  and return the result. Hint: you may call your *calculate\_mean* ( ) function to determine the result.



(5 pts) double calculate\_standard\_deviation (double variance) - Calculates the standard deviation as  $\sqrt{variance}$  and returns the result. Recall that you may use the *sqrt* ( ) function that is found in *math.h*.



(10 pts) double find\_max (double number1, double number2, double number3, double number4, double number5) – Determines the maximum number out of the five input parameters passed into the function, returning the max.



(10 pts) double find\_min (double number1, double number2, double number3, double number4, double number5) – Determines the minimum number out of the five input parameters passed into the function, returning the min.



(5 pts) void print\_double (FILE \*outfile, double number) – Prints a double precision number (to the *hundredths* place) to an output file.



(20 pts) A main ( ) function that does the following (this is what the program does!!!):  
 Opens an input file "input.dat" for reading;  
 Opens an output file "output.dat" for writing;

Reads five records from the input file (input.dat); You will need to use a combination of *read\_double* ( ) and *read\_integer* ( ) function calls here!

Calculates the sum of the GPAs;  
 Calculates the sum of the class standings;  
 Calculates the sum of the ages;

Calculates the mean of the GPAs, writing the result to the output file (output.dat);  
 Calculates the mean of the class standings, writing the result to the output file (output.dat);  
 Calculates the mean of the ages, writing the result to the output file (output.dat);

Calculates the deviation of each GPA from the mean (Hint: need to call *calculate\_deviation* ( ) 5 times)  
 Calculates the variance of the GPAs  
 Calculates the standard deviation of the GPAs, writing the result to the output file (output.dat);

Determines the min of the GPAs, writing the result to the output file (output.dat);  
 Determines the max of the GPAs, writing the result to the output file (output.dat);

Closes the input and output files (i.e. input.dat and output.dat)

For this assignment you will be required to read five records from the "input.dat" file. Each record will have the following form:

Student ID# (an 8 digit integer number)

GPA (a floating-point value to the hundredths place)

Class Standing (1 - 4, where 1 is a freshmen, 2 is a sophomore, 3 is a junior, and 4 is a senior --> all integers)

Age (a floating-point value)

Example data for 1 student record in the file could be as follows:

12345678

3.78

3

20.5

#### IV. Expected Results:

The following sample session demonstrates how your program should work.

Assuming input.dat stores the following records:

12345678

3.78

3

20.5

87654321

2.65

2

19.25

08651234

3.10

1

18.0

11112222

3.95

4

22.5

22223234

2.45

3

19.3333

Your program should write the following to output.dat: NOTE: you only need to output the numbers, the text for demonstration purposes only.

3.19 -- GPA Mean

2.60 -- Class Standing Mean

19.92 -- Age Mean

0.60 -- GPA Standard Deviation

2.45 -- GPA Min

3.95 -- GPA Max

#### V. Submitting Assignments:

1. Using Blackboard Learn <https://learn.wsu.edu/webapps/login/> submit your assignment. Under the "Content" link navigate to the "Programming Assignment Submissions" folder and upload your solution to

the appropriate “Assignment” space. You must upload your solution, through an attachment, as <your last name>\_pa3.zip by the due date and time.

2. Your .zip file should contain your one header file (a .h file), two C source files (which must be .c files), and project workspace. Delete the debug folders before you zip the project folder.
3. Your project must build properly. The most points an assignment can receive if it does not build properly is 65 out of 100.

## VI. Grading Guidelines:

This assignment is worth 100 points. Your assignment will be evaluated based on a successful compilation and adherence to the program requirements. We will grade according to the following criteria:



85 pts for adherence to function definitions described above. Please see the individual points, for each function, above.



15 pts for adherence to proper programming style established for the class and comments

