Project #8 (50 points- 10% bonus for exact output match): Arrays
Submit Your Solution Using Canvas by 10PM on Sunday, 4/9/23
(late submissions will be accepted from 4/9/23 from 10PM to 11:58pm)

Obtaining Project 8 Input Order and Output:

Your program's output should look like that produced by the provided sample solution. You can run the sample solution **Project_08_solution** by typing the following at a command prompt in a terminal window**:**

```
/home/work/cpe211/Executables/Project_08/Project_08_solution
```

Provided sample input files are zipped in the file P08_in.zip
Read the Compare Solution document to see information on the comparison script
Go through the project slides.

Input is to follow the same order as that illustrated by the provided solution executable. Output is to be similar to that shown by the provided solution.

Project 8 Restrictions

Any C++ technique covered in Chapters 1 through 11 is allowed.
You are not allowed to use any global variables. If necessary, global constants may be used.
A minimum of six functions are required as described.
Read the Programs using functions document.

Project 8 Description

This project involves working with a one-dimensional array. The maximum size for the one dimensional array is 50 elements. The program will read floating point numbers from an input file and store them in a one dimensional array. The following statistical values will be determined for the numbers read from the input file: The Mean (Average), The median (middle value), the standard deviation and the variance. Formulas for these values are shown on page 4.

The program is to use various functions to perform the necessary actions, and all functions are called from main. No user defined function can call another user defined function. Furthermore, all output (to the terminal) of the statistical values is to be performed in main – so functions must return the values they calculate.

➔ The SIX required functions that must be written are: 1) Sort numbers from low to high (first element of the array is the lowest value) 2) Find the average of the numbers 3) Find the median value 4) Find the variance 5) open an input file and 6) Read in the floating point numbers. Other functions can be written if needed. ⇐

When opening the input file, all code is to be placed in the function that opens the input file. The program is to continually prompt for an input file and attempt to open it until a file is successfully opened. Look at the output of the sample solution for examples of invalid input files.

Information in the input file consists of 1 integer and several floating point numbers (1 per line). The first number in the input file is an integer indicating the number of floating point values to read. If the value is a number, it will be between 1 and 50 inclusive. **Therefore the one-dimensional array that is being used can be declared with a dimension size of 50. No need to read the number and use it to set the dimension size of the array**

The program is to read (**use extraction**) the integer value representing the number of floating point values. This read acts as a priming read on the file and is used to determine if a valid number was read, and this read is also used to determine if the input file is empty.
The value read is used as the limit for the loop control variable when reading(**use extraction**) in the floating point values into the array. Reading of the values is to be performed in a function. After a read of each number check the status of the input file stream to see if a number was correctly read or if the file stream is in the fail state.

After reading in the numbers, sort the numbers from low to high. Use a function to perform this task. This function is to have two parameters – the one dimensional array holding the numbers and an integer representing how many numbers are in the array.

After sorting the numbers, determine the average of the numbers. Use a function to determine this value by passing in the array of floating point numbers and the integer representing how many numbers are in the array. The average value is to be returned back to the caller of the function.

The median value is easy to determine once the numbers are sorted from lowest to highest. Use a function to calculate the median value which is returned to the caller of the function.

> **<u>If the number of values in the array is odd</u>**, the median value is the middle value in the list of numbers. To determine the index position of the middle value, divide the number of values in the array by 2 (**use integer division**).
>
> **<u>If the number of the values in the array is even</u>,** the median value is the average of the two middle values in the list of numbers. The index values for these two middle values are the **number of values in the array divided by 2 (use integer division)** and
> **(the number of values in the array divided by 2) - 1**

Next calculate the variance of the numbers using the formula provided on page 4. Remember, array indexing goes from 0 to dimension size -1. The function is to return the variance calculated to the caller of the function (which is main).

In main, the standard deviation is calculated as the square root of the variance.

Once all values are calculated, output the results to the terminal as shown in the sample solution.

<u>Project 8 Requirements</u>

- Program must contain at least the 6 functions mentioned in the description.
- **All statistical output statements are to be performed in main. 🡐 🡐**
- **User defined functions can be called from main only. All function def. go below main**

- **Identifying phrases for the statistical values are left justified in a field width of 25 – note setw and setfill must be used with these phrases**
- A function to open the input file is required. The function should **continue to prompt the user for a valid input file** name until a valid name has been entered or ctrl-c is typed to exit the program. All code required to open an input file is to be in this function; therefore, main should contain the input file stream variable declaration and the function call only.
- To reset an input stream that failed to open use **inputFileStreamVariable.clear();**

Project 8 Directions:

- Open a terminal window and move (cd) into the Project_08 directory created in the CPE211_SPR23 directory (This is the directory structure created in project 1.) The command for this is: **cd CPE211_SPR23/Project_08 (typing cd ~/CPE211/Project_08 works as well)**. You will need to modify the names as necessary to match your capitalization style for the two directories.
- Download all needed files from Canvas into this directory. Using a Text Editor (i.e. nedit), edit a file for project 8 (i.e. **nedit Project_08.cpp**). Write your complete program so that it **produces the desired output as shown by the provided solution executable.**
- Test the operation of your program using the sample data files provided and compare results from your program to those of the sample solution
- Several header files are required: **string, iomanip, fstream and iostream**.

Project 8 Helpful Information

- Statistical identifying phrases are output in a field width of 25 left justified
- Use setfill('.') as an output manipulator to obtain the periods between the identifying phrases and the value output.
- Re-Use the function for opening the input file from previous projects.
- Most of the functions written will require at least a one-dimensional array parameter and an integer parameter representing how many numbers are in the array
- Use a defined constant for the maximum number of values possible for the array (50 for this program). Use this defined constant as necessary.
- **All message headings and bottom lines are 47 characters long.**
- **Use LOOPS for performing all of the necessary actions with the calculations.**
- **Input file can be empty. First value in the input file representing the number of floating point values to read may be an integer number or some other character.**
- **Program should test for invalid data possibilities when reading the floating point values**
- **Program needs to be able to handle invalid file names**

Project 8 Assumptions

- Each number is on a separate line and all values are read using extraction.
- **Look at input files. Blank lines may exist between numbers in the input files**

Project 8 Formulas

The cmath header file: This header file contains definitions for various math functions including the pow(x,y) function. The pow(x,y) function calculates the value of x^y. If pow(x,y) is used when calculating the variance, y must be the integer value of 2.

For more information look at the Project 6 description.

Formulas Required

$$\textit{Sum} \quad X_{sum} = \sum_{i=0}^{n-1} X_i$$

$$\textit{Average:} \quad X_{avg} = \frac{X_{sum}}{n}$$

$$\text{variance:} \quad \text{var} = \left(\frac{1}{n}\right) \sum_{i=0}^{n-1} (X_i - X_{avg})^2$$

$$\textit{Standard Deviation} \quad \sigma = \sqrt{\text{var}}$$

Note: the summations go from 0 to n-1 where n is the number of floating point values. This terminology is used because the numbers are stored in the array starting with index value 0 to index value n-1.

**The variance calculated in this program is a population variance – this means that we have the entire population of values of interest and not just a sample of a population. Look at this page for an excellent description of calculating variance: https://www.wikihow.com/Calculate-Variance .