

Introduction

During this lab you will:

1. Practice using vim editor. We will use it every lab from now on.
2. Write a program to calculate standard deviation.
3. Compile and run a program that consists of several files.
4. Create a zipped archive called “tarball” and submit it to canvas.

Part I – Learning how to use vim editor

Your TAs will conduct a short demo on how to use vim editor. They will then assist you with it during the class. Here is a helpful tutorial: <https://www.openvim.com/>. And another useful link from Linux: <https://www.linux.com/tutorials/vim-101-beginners-guide-vim/>.

Part II. Write a program to calculate standard deviation.

Creating necessary files

Your program will consist of three files.

1. Header file

header file named *stddev.h* will contain your function prototype and the preprocessor directives. These directives ensure the file does not get included twice by the preprocessor.

```
#ifndef STDDEV_H
#define STDDEV_H
    void stats (float *data, int n);
#endif
```

2. Implementation file

stddev.cpp file will contain function definition (implementation). This file will include your header file to let the compiler know where your function prototype can be found. The function will calculate the standard deviation and print the result. This is a void function and will not be returning any values to main.

Pseudocode for calculating standard deviation

- a. Find the mean (average) by dividing the sum of all elements of the array by the number (N) of elements in the array.

- b. Then for each of the N number in the array: subtract the mean from the number and square the result.
- c. Calculate the mean of the squared differences.
- d. Calculate the square root of the result from the previous computation.

Here is a link that explains how this works: <https://www.mathsisfun.com/data/standard-deviation-formulas.html> and the formula from the same website:

This is the formula for Standard Deviation:

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2}$$

3. Main driver

the driver *main.cpp* will contain the main function. You will include your header file here, but not the “*stddev.cpp*” file. Here you will do the following:


- ask user to input the size of the array.
- allocate space on the heap by using the C++ style operator *new*. (Do not forget to free up that memory at the end when you are done with it by using C++ operator *delete*.)
- initialize the elements of the array to 4, 8, 12, 16, etc.
- pass this array to the function *stats* that will calculate and print the standard deviation.

Compiling multiple files into an executable file

To compile your program do the following:

```
g++ -c stddev.cpp -Wall -o stddev.o
g++ -c main.cpp -Wall -o main.o
g++ main.o stddev.o -Wall -o out
```

➡ Please note that compiling with the flag *-c* tells the compiler to compile, but not link the files. This will create a binary object code. Object files have an extension *.o*. Flag *-o* allows you to rename the default filenames created by compiler. *-Wall* enables warnings for many common errors and should always be used.

 Also, please note that the header file is not compiled separately. It will be included into the `.cpp` file by the preprocessor and compiled when you compile the `.cpp` file.

What/How to submit

That's it, you are done! After thoroughly testing your program create a zipped tarball and submit to canvas to be graded.

NOTE: If you submit the archive that cannot be open, or is corrupt, you will not be given the opportunity to redo the work and resubmit. You will get a zero on the lab assignment. You have one shot to get it right. Please ask your TAs for help, if you need it.