Lab A Reflections

Student: Joaquin Saldana

This document is an account of how I constructed the lab assignment and the logic and reasoning behind my design choices.

I originally created 5 files.

1. Average.h
2. Summation.h
3. Average.cpp
4. Summation.cpp
5. LabAMain.cpp (main function)

I originally drafted my .cpp source files (average.cpp and summation.cpp) to accept static arrays as parameters for the functions. The function definitions for average and summation contained standard for loops. Each class contained one member variable of type double which was returned when the functions were called.

Ex:

```
double average::arrayAverage(double realNumberArray[] , int arraySize)
{
    for(int i = 0; i < arraySize; i++)
    {
        totalAverage += realNumberArray[i];
    }

    totalAverage = totalAverage / arraySize;

    return totalAverage;
}
```

Main contained a hard coded array w/ various values (see below) and performed the following output:

```
int main()
{
    double myArray[] = {2.5, 4.89, 5, 0.23, 6, 8, 9.2348};

    average arrayOne;

    summation arrayTwo;

    cout << "This is the average of array myArray: " <<
arrayOne.arrayAverage(myArray, 7) << endl;
```

```
    cout << "This is the sum of array myArray: " <<
arrayTwo.arraySummation(myArray, 7) << endl;

    return 0;
}
```

OUTPUT :

```
This is the average of array myArray: 5.12211
This is the sum of array myArray: 35.8548
Program ended with exit code: 0
```

I then modified the program to accept input from the user. I also changed my code to validate user input to ensure the user is entering the data type requested. In addition, I modified the array address parameter to a constant for both the summation and average function in their respective header and .cpp files.

I also used a pointer to a dynamic array in my main file since the input from the user is technically unknown.

Below is an example of the output produced after my changes:

I tested by validation code to ensure it validated when it prompted the user to enter the number of elements he/she wished to add to the array/list and the validate the numbers entered are indeed numbers and not letters.

```
Welcome, this program provides the sum and average of a list of
numbers.  How many numbers would you like to add to the list?
gh
You must enter a number.  How many numbers do you wish to add to the
list?
t
You must enter a number.  How many numbers do you wish to add to the
list?
3
Please provide a number you wish to add to the list:
1.5
Please provide a number you wish to add to the list:
as
You did not enter a number.  Please enter a whole or decimal number:
8.9
Please provide a number you wish to add to the list:
78945.5782
The sum of the 3 you entered is: 78956
This average of the 3 you entered is: 26318.7
```

Program ended with exit code: 0

I noticed if I did not include the .clear() and .ignore() functions of the cin object I would be left with an infinite loop so I ensured they were included in the while loops where data was validated.

I tested on last time to ensure the output values were correct:

Welcome, this program provides the sum and average of a list of numbers. How many numbers would you like to add to the list?
astgasdfdasf
You must enter a number. How many numbers do you wish to add to the list?
4
Please provide a number you wish to add to the list:
qwerquoipq
You did not enter a number. Please enter a whole or decimal number:
8.9
Please provide a number you wish to add to the list:
7.5555
Please provide a number you wish to add to the list:
qweripuqew
You did not enter a number. Please enter a whole or decimal number:
2
Please provide a number you wish to add to the list:
9.999999999
The sum of the 4 you entered is: 28.4556
This average of the 4 you entered is: 7.11389
Program ended with exit code: 0