Assignment 4

Due Oct 19 by 7pm **Points** 120

Project 4.a

The mode is the value that appears most often in a set of data. Write a function named *findMode* that takes as parameters an **array** of int and the size of the array, and returns a **vector** containing the mode(s). If there is just a single most frequent value, the vector will only contain that one value, but if multiple values tie for maximum frequency, the vector will need to contain all such values. This includes the case where every number in the array appears only once. Each mode should appear only once in the vector. The values in the vector that is returned must be in ascending order. If you need to sort a vector, it's similar to sorting an array, but specifying the beginning and end of the vector look a little bit different. If your vector is named *result*, then it would look like this: "std::sort(result.begin(), result.end());".

The most straightforward approach is to:

1. Iterate (loop) through the array to find out what the highest frequency for any value is **without** worrying about storing any modes.
2. Iterate through the array again, this time comparing the counts for each value to the highest frequency that you already found, if the count for a value is the same as the highest frequency, push that value into your results vector.

The file must be named: **findMode.cpp**

Project 4.b

Write a void function named smallSort2 that takes as parameters the addresses of three int variables and sorts the ints at those addresses into ascending order. For example if the main method has:

```
int a = 14;
int b = -90;
int c = 2;
smallSort(&a, &b, &c);
cout << a << ", " << b << ", " << c << endl;
```

Then the output should be:

```
-90, 2, 14
```

The file must be named **smallSort2.cpp**.

Project 4.c

Write a class called Student that has two data members - a string variable called name and a double variable called score. It should have a constructor that takes two values and uses them to initialize the data members. It should have get methods for both data members (getName and getScore), but doesn't need any set methods.

Write a separate function called stdDev that takes two parameters - an array of pointers to Students and the size of the array - and returns the standard deviation for the student scores (the population standard deviation that uses a denominator of N, **not** the sample standard deviation, which uses a different denominator).

(Note: I could have had you use an array of Students for this instead of an array of pointers to Students, but I want you to practice pointer syntax.)

The files must be named **Student.hpp**, **Student.cpp** and **stdDev.cpp**.