LW: Pass By Constant Reference

Objectives

- To understand that:
  - argument passing is similar to initialization,
  - when a function is called, each formal argument is initialized by its corresponding actual argument, and
  - when a constant reference is provided as a formal parameter, the corresponding actual argument will be referred to by the formal parameter, but will not be mutable.

Labwork

1. There are questions throughout this document that you will need to respond to for credit.
2. You must compile and run this code on **`build.tamu.edu`**, executing the command `g++ LW_Pass-By-Const-Ref.cpp` to compile, and then `./a.out` to run the resultant executable.
3. Download the skeleton code for this lab from **`https://drive.google.com/open?id=0B_ouNNuWgNZCVVZuS0xBLXdZdTA`**
4. At this point, I haven't updated the function declaration or definition to pass the arguments to our vint_half_sum function by constant reference. We will do that in a moment.
5. First, compile and run the program.
6. Now update the function declaration and definition to pass the argument by constant reference; if you're unsure how to do this, update the function declaration to `int vint_half_sum(vector<int> const&);` and the first line of the function definition to read `int vint_half_sum(vector<int> const& v)`.
7. Compile the program and then answer the following questions?
   a. What error message is presented to you by the compiler?

      "Assignment of read-only location"

   b. What do you think that this error message means?

      The memory location cannot be changed, as it has been set to be read-only by the usage of "const".

c. How could you update the code such that running sum being calculated in vint_half_sum is still correct, but that the vector isn't modified?

You can use a temporary variable within the loop to assign the value of the halved v value, which doesn't attempt to modify v.

8. As I mentioned in the form of a comment in the provided code, it really doesn't make much sense to first assign `v.at(i) = v.at(i) / 2;` and then `sum += v.at(i);` when we could have simply `sum += v.at(i)/2;` provided that this function should simply calculate the sum of one-half the value of each element of the vector.
9. How does pass by constant reference differ from pass by reference?

Passby constant references allow a function to directly work with the addresses and values of an object, but do not allow for those values to be changed, while regular references do.

10. How does pass by constant reference differ from pass by value? When might you chose the former over the latter? The latter over the former?

Constant reference is able to get the values of the object as they are in the main object and main function without creating a copy. Using value only gives a copy of what values the object had at one set point. With constant references, you can work with the values as they change, without trying to change them. Const reference is dynamic where value is static.

11. Capture the output written to the terminal window by this program in the form of a screenshot; if you cannot include everything, that's okay. Drag and drop or paste your screenshot to the box below:

Submission

Save this completed labwork as a PDF [File -> Download As -> PDF Document (.pdf)] and submit to gradescope for grading.