5. Iterators

In the directory `Iterators`, you need to write the implementation.cpp and header.hpp files for the following functions. Please add a main function to your implementation.cpp file and use a debugger if you encounter runtime bugs.

Find Smallest Index Range

You need to write function named `FindSmallestIndexRange` that takes vector (of ints) and two integers (denoting a range of indexes). This function should return an iterator pointer at the smallest element in the vector within the range denoted (inclusive). If multiple elements tie for the smallest, return the first one. Please note that the last argument may denote a position beyond the end of the vector, so make sure you don't read past the end of the vector. You can assume the range includes at least one element.

Example:

```
std::vector<int> data = {6, 1, 5, 4, 3, 2, -1};
auto iter_smallest = FindSmallestIndexRange(data, 2, 5);
// the range of the vector bounded by the integers is {5, 4, 3, 2}
// the smallest element in this range is the 2.
// So an iterator pointing at the 5th position in data should be returned.
```

Please note that the test cases are using auto to conceal the correct return type for this function. Please note that the function should be able to work on const vectors, as it doesn't need to change them.

Double Iterator Range

You need to write a templated void function (named `DoubleIteratorRange`) that takes two non-const forward iterators to a container of a numeric type (e.g. int or double). The function should double all the elements in the range denoted by the iterators. The end iterator denotes one-past-the-end of the range (please don't dereference it as it may point at one-past-the-end of the vector). You can assume the range includes at least one element.

Example:

```
std::vector<float> data = {5.6, 2.3, 12, 19};
DoubleIteratorRange(data.begin(), data.end());
std::vector<float> expected = {11.2, 4.6, 24, 38};
CHECK(data == expected);
```