std::ref and std::cref

In the last lesson, we were introduced to reference wrappers. Now, we will learn how to create them.

With the helper functions std::ref and std::cref you can easily create
reference wrappers to variables. std::ref will create a non constant
reference wrapper, std::cref a constant one:

```
// referenceWrapperRefCref.cpp
#include <iostream>
#include <functional>
void invokeMe(const std::string& s){
  std::cout << s << ": const " << std::endl;</pre>
}
template <typename T>
 void doubleMe(T t){
 t*= 2;
}
int main(){
  std::string s{"string"};
 invokeMe(std::cref(s));  // string
 int i= 1;
  std::cout << i << std::endl; // 1
  doubleMe(i);
  std::cout << i << std::endl; // 1S</pre>
  doubleMe(std::ref(i));
  std::cout << i << std::endl; // 2
  return 0;
}
                                                                                         נט
```

The helper functions `std::ref` and `std::cref`

So it's possible to invoke the function invokeMe, which gets a constant
reference to an std::string, with a non-constant std::strings, which is

wrapped in a std::cref(s). If I wrap the variable i in the helper function

std::ref, the function template doubleMe will be invoked with a reference. So
the variable i will be doubled.

In the next lesson, let's talk about smart pointers in C++.