【C++】Day67

Class	C++
Date	@March 2, 2022
Material	
Series Number	
Summary	Lambdas are Function Objects

【Ch14】Overloaded Operations and Conversions

14.8.1 Lambdas Are Function Objects

When we write a lambda, the compiler translates that expression into an unnamed object of an unnamed class. The class generated from a lambda contain an overloaded function-call operator.

For example, the lambda that we passed as the last argument to `stable_sort`:

```
// sort words by size, but maintain alphabetical order for words of the same size
stable_srt(words.begin(), words.end(),
  [](const string &a, const string &b) { return a.size() < b.size(); });
```

Acts like an unnamed object of a class that would look something like

```
class ShorterString {
public:
  bool operator() (const string &s1, const string &s2) const {
    return s1.size() < s2.size();
  }
};
```

By default, lambdas may not change their captured variables. As a result, be default, the function-call operator in a class generated from a lambda is a `const` member function.

Classes Representing Lambdas with Captures

Variables that are captured by value are copied into the lambda expression. As a result, classes generated from lambdas that capture variables by value have data members corresponding to each such variable.

These classes also have a constructor to initialize these data members from the value of the captured variables.

For example, the lambda that we used to find the first string whose length was greater than or equal to a given bound:

```
// get an iterator to the first element whose size is >= sz
auto wc = find_if(words.begin(), words.end(), [sz](const string &a) { return a.size() >= sz; });
```

would generate a class that looks something like

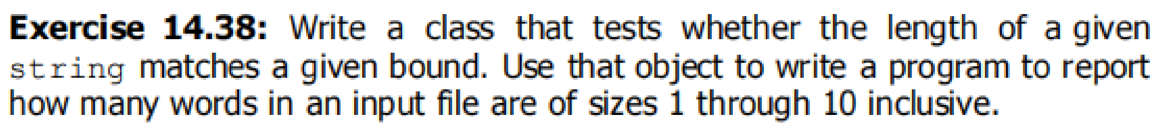
```
class SizeComp {
  SizeComp(size_t n) : sz(n) {} // parameter for each captured variable

  // call operator with the same return type, parameters, and body as the lambda
  bool operator()(const string &s) const {
    return s.size() >= sz;
  }

private:
  size_t sz; // a data member for each varaible captured by value
};
```

To use this class, we must pass an argument

```
// get an iterator to the first element whose size() is >= sz
auto wc = find_if(words.begin(), words.end(), SizeComp(sz));
```

Classes generated from a lambda expression have a deleted default constructor, deleted assignment operators, and a default constructor.

Exercise



Exercise 14.38: Write a class that tests whether the length of a given `string` matches a given bound. Use that object to write a program to report how many words in an input file are of sizes 1 through 10 inclusive.

See 14_38.cpp for code