Advanced Programming Concepts with C++

CSI 2372

Tutorial # 1

Solutions for selected exercises from chapter 3 & 5

Ahmedou Jreivine

Exercise 3.4

- Write a program to read two strings and report whether the strings are equal. If not, report which of the two is larger. Now, change the program to report whether the strings have the same length, and if not, report which is longer.

Solution 3.4 1/2

```
#include <iostream>
#include <string>
void check_equal() {
    std::string s1, s2;
    if (std::cin >> s1 >> s2) {
        if (s1 == s2)
        std::cout << "The two strings are equal." << std::endl;
        else if (s1 > s2)
            std::cout << "The first string is larger." << std::endl;
        else
            std::cout << "The second string is larger." << std::endl;
    }
}
```

Solution 3.4   2/2

```
void check_length() {
    std::string s1, s2;
    if (std::cin >> s1 >> s2) {
    if (s1.size() == s2.size())
        std::cout << "The two strings have same length." << std::endl;
    else if (s1.size() > s2.size())
        std::cout << "The first string is longer." << std::endl;
    else
        std::cout << "The second string is longer." << std::endl;
    }
}
int main() {
    check_equal();
    check_length();
    return 0;
}
```

Exercise 3.6:

Use a range for to change all the characters in a string to X.

Solution 3.6:

```
#include <iostream>
#include <string>
int main() {
    std::string s;
    if (std::cin >> s) {
        for (auto &c : s) // `c` is `char &`
        c = 'X';
    }
    std::cout << s << std::endl;
    return 0;
}
```

Exercise 3.12:

- Which, if any, of the following vector definitions are in error? For those that are legal, explain what the definition does. For those that are not legal, explain why they are illegal.
  - **(a)** vector<vector<int>> ivec;
  - **(b)** vector<string> svec = ivec;
  - **(c)** vector<string> svec(10, "null");

Solution 3.12

```
#include <vector>
#include <string>
int main() {
        std::vector<std::vector<int>> ivec;
        // OK, a vector of vector of int, similar to 2-d int array
        //std::vector<std::string> svec = ivec;
        // Error: the type of `svec` and `ivec` doesn't match
        std::vector<std::string> svec2(10, "null");
        // OK, a vector of ten strings whose value are all "null"
        return 0;
}
```

Exercise 3.13:

- How many elements are there in each of the following vectors? What are the values of the elements?

  (a) vector<int> v1;

  (b) vector<int> v2(10);

  (c) vector<int> v3(10, 42);

  (d) vector<int> v4{10};

  (e) vector<int> v5{10, 42};

  (f) vector<string> v6{10};

  (g) vector<string> v7{10, "hi"};

Solution 3.13

```
#include <vector>
#include <string>
using std::vector;
using std::string;
int main() {
        vector<int> v1; // 0 element
        vector<int> v2(10); // 10 elements, values are all 0
        vector<int> v3(10, 42); // 10 elements, values are all 42
        vector<int> v4{10}; // 1 element, value is 10
        vector<int> v5{10, 42}; // 2 elements, values are 10 and 42
        vector<string> v6{10}; // 10 elements, values are all empty string
        vector<string> v7{10, "hi"}; // 10 elements, values are all "hi"
        return 0;
}
```

Exercise 3.16:

Write a program to print the size and contents of the vectors from exercise 3.13. Check whether your answers to that exercise were correct.

Solution 3.16 1/2

```
#include <iostream>
#include <vector>
#include <string>
using std::vector;
using std::string;
void print_vector_int(const vector<int> &v) {
    for (const auto & elem : v)
        std::cout << elem << " ";
    std::cout << std::endl;
}
void print_vector_string(const vector<string> &v) {
    for (const auto & elem : v)
        std::cout << elem << " ";
    std::cout << std::endl;
}
```

Solution 3.16 2/2

```
int main() {
    vector<int> v1; // 0 element
    vector<int> v2(10); // 10 elements, values are all 0
    vector<int> v3(10, 42); // 10 elements, values are all 42
    vector<int> v4{10}; // 1 element, value is 10
    vector<int> v5{10, 42}; // 2 elements, values are 10 and 42
    vector<string> v6{10}; // 10 elements, values are all empty string
    vector<string> v7{10, "hi"}; // 10 elements, values are all "hi"
    print_vector_int(v1);
    print_vector_int(v2);
    print_vector_int(v3);
    print_vector_int(v4);
    print_vector_int(v5);
    print_vector_string(v6);
    print_vector_string(v7);
    return 0;
}
```

Exercise 5.4:

- Explain each of the following examples, and correct any problems you detect.

(a)

```
while (string::iterator iter != s.end()) { /* . . . */ }
```

(b)

```
while (bool status = find(word)) { /* . . . */ }
if (!status) { /* . . . */ }
```

Solution 5.4

- **(a)** The loop variable iter was not initialized before used. It should be initialized first.

```
string::iterator iter = s.begin();
while (iter != s.end()) { /* ... */ }
```

- **(b)** The loop variable status is used outside the scope of the while statement, thus it should be defined outside that scope.

```
bool status;
while (status = find(word)) { /* ... */ }
if (!status) { /* ... */ }
```

Exercise 5.7:

Correct the errors in each of the following code fragments:

(a)

```
if (ival1 != ival2)
        ival1 = ival2
else ival1 = ival2 = 0;
```

(b)

```
if (ival < minval)
        minval = ival;
        occurs = 1;
```

Click here to view code image

(c)

```
if (int ival = get_value())
        cout << "ival = " << ival << endl;
if (!ival)
        cout << "ival = 0\n";
```

(d)

```
if (ival = 0)
        ival = get_value();
```

Solution 5.7

```
(a)     if (ival1 != ival2)
                ival1 = ival2; // Missing semicolon
        else ival1 = ival2 = 0;
(b)     if (ival < minval) { // Need a block for more than one statements
                minval = ival;
                occurs = 1;
        }
(c)     int ival; // Used outside the first if scope, thus defined outside
        if (ival = get_value())
                cout << "ival = " << ival << endl;
        if (!ival)
                cout << "ival = 0\n";
(d)     if (ival == 0) // Should be equality operator instead of assignment
                ival = get_value();
```

Exercise 5.14:

Write a program to read strings from standard input looking for duplicated words.

The program should find places in the input where one word is followed immediately by itself.

Keep track of the largest number of times a single repetition occurs and which word is repeated.

Print the maximum number of duplicates, or else print a message saying that no word was repeated.

For example, if the input is:

how now now now brown cow cow, the output should indicate that the word now occurred three times.

Solution 5.14

```
#include <iostream>
#include <string>
using std::cout; using std::cin; using std::endl; using std::string; using std::pair;
int main()
{
    pair<string, int> max_duplicated;
    int count = 0;
    for (string str, prestr; cin >> str; prestr = str){
        if (str == prestr) ++count;
        else count = 0;
        if (count > max_duplicated.second) max_duplicated = { prestr, count };
    }
    if (max_duplicated.first.empty())
          cout << "There's no duplicated string." << endl;
    else cout << "the word " << max_duplicated.first << " occurred "
    << max_duplicated.second + 1 << " times. " << endl;
    return 0;
}
```

Exercise 5.15:

Explain each of the following loops. Correct any problems you detect.

Click here to view code image

```
(a) for (int ix = 0; ix != sz; ++ix) { /* . . . */ }
if (ix != sz)
// . . .
```

Click here to view code image

```
(b) int ix;
for (ix != sz; ++ix) { /* . . . */ }
(c) for (int ix = 0; ix != sz; ++ix, ++ sz) { /* . . . */ }
```

Solution 5.15

```
(a) int ix;
    for (ix = 0; ix != sz; ++ix) { /* ... */ }
    //for (int ix = 0; ix != sz; ++ix) { /* ... */ }
    if (ix != sz)
    // ...
    // The loop variable is used outside `for` scope, thus should be defined outside.
(b) int ix;
    //for (ix != sz; ++ix) { /* ... */ }
    // When the initialization is unnecessary, a null statement should be used.
    for (; ix != sz; ++ix) { /* ... */ }
(c) for (int ix = 0; ix != sz; ++ix, ++ sz) { /* ... */ }
    // The loop will never end.
    for (int ix = 0; ix != sz; ++ix) { /* ... */ }
```

Exercise 5.16:

- The *while-loop* is particularly good at executing while some condition holds; for example, when we need to read values until end-of-file.
- The *for-loop* is generally thought of as a step loop: An index steps through a range of values in a collection. Write an idiomatic use of each loop and then rewrite each using the other loop construct. If you could use only one loop, which would you choose? Why?

Solution 5.16

```
#include <iostream>
#include <vector>
int main() {
        int i;
        while (std::cin >> i) { /* ... */ }
        for (int j; std::cin >> j;) { /* ... */ }
        std::vector<int> iv(10, 1);
        for (auto it = iv.begin(); it != iv.end(); ++it) { /* ... */ }
        auto it2 = iv.begin();
        while (it2 != iv.end()) {
            ++it2;          /* ... */
        }
        // `for`-loop is better: it can do what a `while`-loop can, but not vise versa.
        return 0;
}
```

Exercise 5.17:

Given two vectors of *ints*, write a program to determine whether one vector is a prefix of the other.

For vectors of unequal length, compare the number of elements of the smaller vector.

For example, given the vectors containing 0, 1, 1, and 2 and 0, 1, 1, 2, 3, 5, 8, respectively your program should return true.

Solution 5.17

```
#include <iostream>
#include <vector>
#include <string>
#include <sstream>
using namespace std;
void readVector(vector<int> &v) {
    string str;
    cout << "Enter vector elements:" << endl;
    getline(cin, str);
    stringstream ss(str);
    int i;
    while (ss >> i)
        v.push_back(i);
}
bool isPrefix(const vector<int> &v1, const vector<int> &v2) {
    auto it1 = v1.cbegin(), it2 = v2.cbegin();
    for (; it1 != v1.cend() && it2 != v2.cend(); ++it1, ++it2)
        if (*it1 != *it2)
            break;
    return it1 == v1.cend() || it2 == v2.cend();
}
int main() {
    vector<int> v1, v2;
    readVector(v1); readVector(v2);
    cout << (isPrefix(v1, v2) ? "true" : "false") << endl;
    return 0;
}
```

Thanks

Accreditation:

- This presentation is prepared/extracted from the following resources:
  - C++ Primer, Fifth Edition.
    Stanley B. Lippman Josée Lajoie Barbara E. Moo
  - https://github.com/jaege/Cpp-Primer-5th-Exercises
  - https://github.com/Mooophy/Cpp-Primer