Exercises week 6

Klaas Isaac Bijlsma s2394480
David Vroom s2309939

January 10, 2018

Exercise 42

Learn to extract lines using generic algorithms into a container holding ***`string`*** *objects, although* ***`operator>>()`*** *extracts strings*

main.cc

```
1  #include <iostream>
2  #include <iterator>
3  #include <string>
4  #include <vector>
5  #include <algorithm>
6
7  using namespace std;
8
9  class Derived: public string
10 {};
11
12 istream &operator>>(istream &istr, Derived &str)
13 {
14     return getline(istr, str);
15 }
16
17 int main()
18 {
19     vector<string> vs;
20
```

```
21      copy(istream_iterator<Derived>(cin), istream_iterator<Derived>(),
22           back_inserter(vs));
23  }
```

Exercise 43

Learn to use promotion with generic algorithms and predefined function objects when manipulating basic data types.

main.cc

```
1  #include <iostream>
2  #include <algorithm>
3  #include <functional>
4
5  using namespace std;
6
7  int main(int argc, char **argv)
8  {
9      sort(argv + 1, argv + argc, greater<string>());
10     copy(argv + 1, argv + argc, ostream_iterator<string>(cout));
11     cout << '\n';
12
13     sort(argv + 1, argv + argc, less<string>());
14     copy(argv + 1, argv + argc, ostream_iterator<string>(cout));
15     cout << '\n';
16 }
```

Exercise 44

Learn to recognize a situation where lambda functions may be used

main.cc

```
1  #include <iostream>
2  #include <map>
3  #include <algorithm>
4  #include <vector>
5  #include <fstream>
6
7  using namespace std;
8
9
10 class Vstring: public vector<string>
11 {
12     vector<string> d_vs;
13
14     public:
15         typedef map<char, size_t> Charmap;
16
17         explicit Vstring(istream &in);
18
19         size_t count(Charmap &cmap, bool (*accept)(char, Charmap &));
20
21         void display(Charmap &cmap);
22
23     private:
24         //static size_t countChar(string const &str, Charmap &cmap, bool (*acce
25 };
26
27 Vstring::Vstring(istream &in)
28 {
29     copy(istream_iterator<string>(in), istream_iterator<string>(),
30          back_inserter(d_vs));
31 }
32
33 inline size_t Vstring::count(Charmap &cmap, bool (*accept)(char, Charmap &))
34 {
```

```
35      size_t count = 0;
36      for_each(
37          d_vs.begin()->begin(), d_vs.end()->end(),
38          [&, accept](char &ch)
39          {
40              if (accept(ch, cmap))
41                  ++count;
42          }
43      );
44      return count;
45  }
46
47  void Vstring::display(Charmap &cmap)
48  {
49      for_each(
50          cmap.begin(), cmap.end(),
51          [](auto const &value)
52          {
53              cout << value.first << ": " << value.second << '\n';
54          }
55      );
56  }
57  void print_modulo(const vector<int>& v, ostream& os, int m) // output v[i] to o
58  {
59      for_each(begin(v),end(v),
60               [&os,m](int x) { if (x%m==0) os << x << '\n'; } );
61  }
62
63  //size_t Vstring::countChar(string const &str, Charmap &cmap, bool (*accept)(ch
64  //{
65  //}
66
67
68  bool vowels(char c, Vstring::Charmap &cmap)
69  {
70      if (string("aeiuoAEIUO").find(c) != string::npos)
71      {
72          ++cmap[c];
73          return true;
74      }
75      return false;
```

```
76 }
77
78 int main()
79 {
80     ifstream is("text.txt");
81     Vstring vstring(is);
82     Vstring::Charmap vmap;
83
84     cout << "Vowels: " << vstring.count(vmap, vowels) << '\n';
85     vstring.display(vmap);
86 }
```

Exercise 45

Exercise 46

Exercise 47

Exercise 48