10.1答：#include<iostream>

#include <algorithm>

#include <vector>

using namespace std;

int main()

{

int i;

vector<int> ivec;

while(cin>>i)

ivec.push\_back(i);

int i\_cnt=0;

i\_cnt=count(ivec.cbegin(),ivec.cend(),2);

cout<<i\_cnt<<endl;

}

10.2答：#include<iostream>

#include <string>

#include <algorithm>

#include <list>

using namespace std;

int main()

{

string s;

list<string> svec;

while(cin>>s)

svec.push\_back(s);

int i\_cnt=0;

string word="chen";

i\_cnt=count(svec.cbegin(),svec.cend(),word);

cout<<i\_cnt<<endl;

}

10.3答：#include<iostream>

#include <string>

#include <vector>

#include <algorithm>

#include <numeric>

using namespace std;

int main()

{

vector<int> ivec;

for(int i=0;i<10;++i)

ivec.push\_back(i);

int sum=accumulate(ivec.begin(),ivec.end(),0);

cout<<sum<<endl;

}

10.4答：没有错误，int与double可以进行转换。

10.5答：不会发生什么，照样比较元素，const char\*可以用==来比较。

10.6答：#include<iostream>

#include<vector>

#include<string>

#include<algorithm>

using namespace std;

int main()

{

vector<int> ival(10);

fill\_n(ival.begin(), 10, 1);

for (auto i : ival)

cout << i << ' ';

cout << endl;

vector<int> ivec;

fill\_n(back\_inserter(ivec), 10, 0);

for (auto i : ivec)

cout << i << ' ';

cout << endl;

}

10.7答：（a）vec.size()不确定；（b）正确；

10.8答：会调用push\_back()；

10.9答：#include<iostream>

#include<vector>

#include<string>

#include<algorithm>

using namespace std;

void elimDups(vector<string> &words)

{

for (auto s : words)

cout << s << " ";

cout << endl;

sort(words.begin(),words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

auto end\_unique = unique(words.begin(), words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

words.erase(end\_unique, words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

}

int main()

{

vector<string> words{ "the", "quick", "red", "fox", "junmps", "over", "the", "slow", "red", "trutle" };

elimDups(words);

}

10.10答：不会添加元素和删除容器中的元素。

11.11答：

#include<iostream>

#include<vector>

#include<string>

#include<algorithm>

using namespace std;

bool isShorter(const string &s1, const string &s2)

{

return s1.size() < s2.size();

}

void elimDups(vector<string> &words)

{

for (auto s : words)

cout << s << " ";

cout << endl;

sort(words.begin(), words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

auto end\_unique = unique(words.begin(), words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

words.erase(end\_unique, words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

}

int main()

{

vector<string> words{ "the", "quick", "red", "fox", "junmps", "over", "the", "slow", "red", "trutle" };

elimDups(words);

stable\_sort(words.begin(), words.end(), isShorter);

for (const auto &s : words)

cout << s << ' ';

cout << endl;

return 0;

}

10.12答：#include <iostream>

#include <string>

#include <vector>

#include <algorithm>

#include <numeric>

#include "my\_Sales\_data.h"

using namespace std;

bool compareIsbn(const Sales\_data &s1, const Sales\_data &s2)

{

return sd1.isbn().size() < sd2.isbn().size();

}

int main()

{

Sales\_data d1("aaaaaa"), d2("a"), d3("aaa"), d4("b"), d5("aaaabbbb");

std::vector<Sales\_data> v = { d1, d2, d3, d4, d5 };

sort(v.begin(), v.end(), compareIsbn);

for (const auto &element : v)

std::cout << element.isbn() << " ";

std::cout << "\n";

return 0;

}

10.13答：#include <iostream>

#include <string>

#include <vector>

#include <algorithm>

#include <numeric>

using namespace std;

bool my\_compare(const string &s)

{

return s.size() >= 5;

}

int main()

{

vector<string> svec{ "aaaa", "aa", "aaaaaaa", "aaaaaaaaa", "aaa", "aaaa" };

auto it = partition(svec.begin(), svec.end(), my\_compare);

for (auto s : svec)

cout << s << " ";

cout << endl;

for (auto iter = svec.begin(); iter != it; ++iter)

cout << \*iter << " ";

cout << endl;

return 0;

}

10.14答：auto sum = [](const int i1, const int i2)

{

return i1 + i2;

};

10.15答：void f()

{

int i = 0;

auto sum = [i](const int j)

{

return i + j;

};

}

10.16答：#include <iostream>

#include <string>

#include <vector>

#include <algorithm>

#include <numeric>

using namespace std;

void elimDups(vector<string> &words)

{

for (auto s : words)

cout << s << " ";

cout << endl;

sort(words.begin(), words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

auto end\_unique = unique(words.begin(), words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

words.erase(end\_unique, words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

}

bool isShorter(const string &s1, const string &s2)

{

return s1.size() < s2.size();

}

string make\_plural(size\_t ctr, const string &word, const string &ending)

{

return (ctr>1) ? word + ending : word;

}

void biggies(vector<string> &words, vector<string>::size\_type sz)

{

elimDups(words);

stable\_sort(words.begin(), words.end(),isShorter);

auto wc = find\_if(words.begin(), words.end(), [sz](const string &a){return a.size() >= sz; });

auto count = words.end() - wc;

cout << count << " " << make\_plural(count, "word", "s")

<< " of length " << sz << " or longer" << endl;

for\_each(wc, words.end(), [](const string &s){cout << s << " "; });

cout << endl;

}

int main()

{

vector<string> words{ "the", "quick", "red", "fox", "junmps", "over", "the", "slow", "red", "trutle" };

biggies(words,5);

}

10.17答：

#include <iostream>

#include <string>

#include <vector>

#include <algorithm>

#include <numeric>

#include "my\_Sales\_data.h"

using namespace std;

//bool compareIsbn(const Sales\_data &s1, const Sales\_data &s2)

//{

// return s1.isbn().size() < s2.isbn().size();

//}

int main()

{

Sales\_data d1("aaaaaa"), d2("a"), d3("aaa"), d4("b"), d5("aaaabbbb");

std::vector<Sales\_data> v = { d1, d2, d3, d4, d5 };

sort(v.begin(), v.end(), [](const Sales\_data &s1, const Sales\_data &s2){return s1.isbn().size() < s2.isbn().size(); });

for (const auto &element : v)

std::cout << element.isbn() << " ";

std::cout << "\n";

return 0;

}

10.18答：#include <iostream>

#include <string>

#include <vector>

#include <algorithm>

#include <numeric>

using namespace std;

void elimDups(vector<string> &words)

{

for (auto s : words)

cout << s << " ";

cout << endl;

sort(words.begin(), words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

auto end\_unique = unique(words.begin(), words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

words.erase(end\_unique, words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

}

bool isShorter(const string &s1, const string &s2)

{

return s1.size() < s2.size();

}

string make\_plural(size\_t ctr, const string &word, const string &ending)

{

return (ctr>1) ? word + ending : word;

}

void my\_biggies\_partition(vector<string> &words, vector<string>::size\_type sz)

{

elimDups(words);

//stable\_sort(words.begin(), words.end(), isShorter);

stable\_sort(words.begin(), words.end(), [](const string &s1, const string &s2){return s1.size() < s2.size();});

for (auto s : words)

{

cout << s << " ";

}

cout << endl;

//auto wc = find\_if(words.begin(), words.end(), [sz](const string &a){return a.size() >= sz; });

auto wc = partition(words.begin(),words.end(), [sz](const string &a){return a.size() >= sz; });

for (auto s : words)

{

cout << s << " ";

}

cout << endl;

//auto count = words.end() - wc;

//cout << count << " " << make\_plural(count, "word", "s")

// << " of length " << sz << " or longer" << endl;

for\_each(words.begin(), wc, [](const string &s){cout << s << " "; });

cout << endl;

}

int main()

{

vector<string> words{ "the", "quick", "red", "fox", "junmps", "over", "the", "slow", "red", "trutle" };

my\_biggies\_partition(words, 5);

}

10.19答：讲上一题中的partition改为stable\_partition即可

10.20答：

#include <iostream>

#include <string>

#include <vector>

#include <algorithm>

#include <numeric>

using namespace std;

int my\_count\_if(vector<string> &words)

{

auto index = count\_if(words.begin(), words.end(), [](const string &s){return s.size() > 6; });

return index;

}

int main()

{

vector<string> svec{ "chenxun", "chen", "chenxun2", "chen1", "chenxun2" };

int i = my\_count\_if(svec);

cout << i << endl;

return 0;

}

10.21答：

#include <iostream>

#include <string>

#include <vector>

#include <algorithm>

#include <numeric>

using namespace std;

int main()

{

int count = 2;

for (int i = 0; i != 5; ++i)

{

std::cout << ([&count](){

count ? --count : count;

return count == 0;

}())

<< std::endl;

}

return 0;

}

10.22答：#include <iostream>

#include <string>

#include <vector>

#include <algorithm>

#include <numeric>

#include <functional>

using namespace std;

using namespace placeholders;

inline bool biggerThan(const string &s, const vector<string>::size\_type sz)

{

return s.size() > sz;

}

unsigned biggerThan6\_function(const vector<string> &words)

{

return count\_if(words.begin(), words.end(), bind(biggerThan, \_1, 6));

}

int main()

{

vector<string> svec{ "chenxun", "chen", "chenxun2", "chen1", "chenxun2" };

int i = biggerThan6\_function(svec);

cout << i << endl;

return 0;

}

10.23答：接受一个参数或二个参数。

10.24答：#include <iostream>

#include <string>

#include <vector>

#include <algorithm>

#include <numeric>

#include <functional>

using namespace std;

using namespace placeholders;

bool check\_size(const string &s, string::size\_type sz)

{

return s.size() < sz;

}

vector<int>::iterator

find\_first\_bigger(vector<int> &v, const string &s)

{

auto it = std::find\_if(v.begin(), v.end(), std::bind(check\_size, s, \_1));

return it;

}

int main()

{

vector<int> ival{ 1, 2, 3, 4, 5, 6, 7, 8, 9 };

string s = "chenxun";

auto it = find\_first\_bigger(ival, s);

cout << \*it << endl;

return 0;

}

10.25答：#include <iostream>

#include <string>

#include <vector>

#include <algorithm>

#include <numeric>

#include <functional>

using namespace std;

using namespace placeholders;

void elimDups(vector<string> &words)

{

for (auto s : words)

cout << s << " ";

cout << endl;

sort(words.begin(), words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

auto end\_unique = unique(words.begin(), words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

words.erase(end\_unique, words.end());

for (auto s : words)

cout << s << " ";

cout << endl;

}

bool isShorter(const string &s1, const string &s2)

{

return s1.size() < s2.size();

}

bool check\_size(const string &s, string::size\_type sz)

{

return s.size() >= sz;

}

ostream &print(ostream &os, const string &s, char c)

{

return os << s << c;

}

string make\_plural(size\_t ctr, const string &word, const string &ending)

{

return (ctr>1) ? word + ending : word;

}

void my\_biggies\_partition(vector<string> &words, vector<string>::size\_type sz, ostream &os=cout, char c=' ')

{

elimDups(words);

//stable\_sort(words.begin(), words.end(), isShorter);

stable\_sort(words.begin(), words.end(), [](const string &s1, const string &s2){return s1.size() < s2.size(); });

for (auto s : words)

{

cout << s << " ";

}

cout << endl;

//auto wc = find\_if(words.begin(), words.end(), [sz](const string &a){return a.size() >= sz; });

//auto wc = partition(words.begin(), words.end(), [sz](const string &a){return a.size() >= sz; });

auto wc = partition(words.begin(), words.end(), bind(check\_size,\_1,sz) );

for (auto s : words)

{

cout << s << " ";

}

cout << endl;

//auto count = words.end() - wc;

//cout << count << " " << make\_plural(count, "word", "s")

// << " of length " << sz << " or longer" << endl;

//for\_each(words.begin(), wc, [](const string &s){cout << s << " "; });

for\_each(words.begin(), wc, bind(print, ref(os), \_1, ' '));

cout << endl;

}

int main()

{

vector<string> words{ "the", "quick", "red", "fox", "junmps", "over", "the", "slow", "red", "trutle" };

my\_biggies\_partition(words, 5, cout, ' ');

}

10.26答：back\_inserter使用push\_back在容器末端插入元素。front\_inserter在容器前端使用push\_front插入元素。inserter有两个参数，调用insert在指定起始位置插入元素。

10.27答：#include <iostream>

#include <string>

#include <vector>

#include <list>

#include <algorithm>

using namespace std;

int main()

{

vector<string> svec{ "chen", "xun", "chen", "chen", "xun" };

list<string> slist;

unique\_copy(svec.begin(), svec.end(), back\_inserter(slist));

for (auto s : slist)

cout << s << " ";

cout << endl;

return 0;

}

10.28答：#include <iostream>

#include <string>

#include <vector>

#include <list>

#include <algorithm>

using namespace std;

int main()

{

vector<int> ivec{ 1, 2, 3, 4, 5, 6, 7, 8, 9 };

vector<int> vec1;

list<int> vec2;

vector<int> vec3;

copy(ivec.begin(), ivec.end(), back\_inserter(vec1));

for (auto i : vec1)

cout << i << " ";

cout << endl;

copy(ivec.cbegin(), ivec.cend(), front\_inserter(vec2));//注意vector和string不支持push\_front操作

for (auto i : vec2)

cout << i << " ";

cout << endl;

copy(ivec.begin(), ivec.end(), inserter(vec3,vec3.begin()));

for (auto i : vec3)

cout << i << " ";

cout << endl;

return 0;

}

10.29答：#include <list>

#include <algorithm>

#include <fstream>

#include <iterator>

using namespace std;

int main()

{

ifstream in("chen.txt");

vector<string> svec1;

//vector<string> svec2;

istream\_iterator<string> cin\_iter1(in),eof;

while (cin\_iter1 != eof)

svec1.push\_back(\*cin\_iter1++);

for (auto s : svec1)

cout << s << " ";

cout << endl;

ifstream in2("chen.txt");

istream\_iterator<string> cin\_iter2(in2);

vector<string> svec2(cin\_iter2, eof);

ostream\_iterator<string> out\_iter(cout," ");

for (auto e : svec2)

\*out\_iter++ = e;

cout << endl;

return 0;

}

10.30答：#include <iostream>

#include <string>

#include <vector>

#include <list>

#include <algorithm>

#include <fstream>

#include <iterator>

using namespace std;

int main()

{

vector<int> ivec;

istream\_iterator<int> cin\_iter(cin), eof;

while (cin\_iter != eof)

ivec.push\_back(\*cin\_iter++);

for (auto i : ivec)

cout << i << " ";

cout << endl;

sort(ivec.begin(), ivec.end());

ostream\_iterator<int> cout\_iter(cout, " ");

for (auto e : ivec)

\*cout\_iter++ = e;

cout << endl;

copy(ivec.begin(), ivec.end(), cout\_iter);

cout << endl;

unique\_copy(ivec.begin(), ivec.end(), cout\_iter);

cout << endl;

return 0;

}

10.31答：见上一题

10.33答：#include <iostream>

#include <string>

#include <vector>

#include <list>

#include <algorithm>

#include <fstream>

#include <iterator>

using namespace std;

void function(const string &intput\_file, const string &out\_file\_even, const string &out\_file\_old)

{

ifstream fin(intput\_file);

istream\_iterator<int> fin\_iter(fin), eof;

ofstream out\_even(out\_file\_even), out\_old(out\_file\_old);//关联输出文件

ostream\_iterator<int> out\_even\_iter(out\_even, " "), out\_old\_iter(out\_old, "\n");//创建输出流迭代器

//while (fin\_iter != eof)

//{

// if ((\*fin\_iter) % 2 == 0)

// \*out\_even\_iter++ = \*fin\_iter++;

// else

// \*out\_old\_iter++ = \*fin\_iter++;

//}

for\_each(fin\_iter, eof, [&](const int i)

{

i % 2 ? \*out\_even\_iter++ = i : \*out\_old\_iter++ = i;

});

cout << endl;

}

int main()

{

function("chenxun.txt", "even.txt", "old.txt");

return 0;

}

10.34答：#include <iostream>

#include <string>

#include <vector>

#include <iterator>

#include <algorithm>

using namespace std;

int main()

{

vector<int> vec= { 1, 2, 3, 4, 5, 6, 7, 8, 9 };

vector<int>::reverse\_iterator iter = vec.rbegin();

//cout << \*iter << endl;

for (; iter != vec.rend(); ++iter)

cout << \*iter << " ";

cout << endl;

for\_each(vec.crbegin(), vec.crend(), [](const int &i){ cout << i << " "; });

cout << endl;

}

10.35答：#include <iostream>

#include <string>

#include <vector>

#include <iterator>

#include <algorithm>

using namespace std;

int main()

{

vector<int> v= {0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };

//for (auto iter = vec.end() - 1; iter <= vec.begin(); --iter)

// cout << \*iter << endl;

for (auto it = v.end() - 1; it != v.begin() ; --it)

std::cout << \*it << " ";

}

10.36答：#include <iostream>

#include <string>

#include <vector>

#include <iterator>

#include <algorithm>

using namespace std;

int main()

{

vector<int> v= {0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };

auto it=find(v.rbegin(), v.rend(),0);

cout << \*it << endl;

}

10.37答：#include <iostream>

#include <string>

#include <vector>

#include <iterator>

#include <algorithm>

#include <list>

using namespace std;

int main()

{

vector<int> v= {0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };

list<int> ilist;

copy(v.rbegin(), v.rend(), back\_inserter(ilist));

for (auto i : ilist)

cout << i << endl;

return 0;

}

10.42答：#include <iostream>

#include <string>

#include <vector>

#include <iterator>

#include <algorithm>

#include <list>

using namespace std;

int main()

{

list<string> slist = { "chen", "chen", "chen", "xun" };

slist.unique();

ostream\_iterator<string> out\_iter(cout, " ");

copy(slist.begin(), slist.end(), out\_iter);

}