Ларин Егор

1.

struct line

{

string buf;

operator std::string() const { return buf; }

};

std::istream& operator>>(std::istream& in, line& line)

{

return std::getline(in, line.buf);

}

list<string> out;

std::ifstream in(fileName);

std::copy(std::istream\_iterator<line>(in),

std::istream\_iterator<line>(),

std::back\_inserter(out));

2.

list<string> lines;

std::ostream\_iterator<list<string>> out\_it(std::cout, "\n");

std::copy(lines.begin(), lines.end(), out\_it);

3.

std::vector<int> v1 = { 1,2,3,4 };

std::vector<int> v2;

std::vector<int>::iterator iter = v2.begin();

std::copy(v1.begin(), v2.end(), iter);

4.

Class Foo{

public:

int val;

};

auto cmp= [](Foo a, Foo b) {

return a.val > b.val;

}

vector<Foo> foos = ...;

std::sort(std::begin(foos), std::end(foos), cmp);

5.

class Match{

public:

char \_key;

Match(char key){

\_key = key;

}

bool operator()(string str){

return str[0] == \_key;

}

};

auto match = [](string str, char key){

if (str[0] == key){

return true

}

}

vector<string> out;

for (auto str : str){

if (match(str)){ // Match(str)

out.push\_back(str);

}

}

6.

auto match = [](string str, char key){

if (str[0] == key){

return true

}

}

list<string> strs;

Match(key);

strs.erase(remove\_if(strs.begin(), strs.end(), match), str.end());

7.

auto match = [](string str, char key){

if (str[0] == key){

return true;

}

}

int n = 0;

for (auto ptr = strs.begin(),

ptr != strs.end(), ptr++){

if (match(\*ptr)){

n++;

}

}

8.

list<string> lines;

std::ostream\_iterator<list<string>> out\_it(std::cout, "\n");

std::copy(++lines.end(), --lines.begin(), out\_it);

9.

set<string> s;

list<string> out;

std::ifstream in(fileName);

std::copy(std::istream\_iterator<string>(in),

std::istream\_iterator<string>(),

std::back\_inserter(out));

for (auto str : out){

s.inser(str);

}

vector<string> unique\_words;

for(auto str : set){

unique\_words.push\_back(str);

}

sort(unique\_words.begin(), unique\_words.end());

for (auto str : strs){

cout << str << endl;

}

10.

map<string, int> m;

list<string> out;

std::ifstream in(fileName);

std::copy(std::istream\_iterator<string>(in),

std::istream\_iterator<string>(),

std::back\_inserter(out));

for (auto str : out){

if (m.find(str) == m.end()){

m[str] = 1;

} else {

++m[str];

}

}

for (const auto &[key, value] : m){

cout << key << “ “ << value << endl;

}

11.

class Foo{

public:

int bar;

int egg;

int spam;

}

vector<Foo> out = { . . .};

auto cmp = [](Foo a, Foo b){

if (a.bar == b.bar){

if (a.egg == b.egg){

if (a.spam == b.spam){

return true;

} else {

return a.spam > b.spam;

}

} else {

return a.egg > b.egg;

}

} else {

return a.bar > b.bar;

}

}

sort(out.begin(), out.end(), cmp);