

#include <iostream>

using namespace std;

#include <string>

class MyAdd

{

public:

int operator()(int v1, int v2)

{

return v1 + v2;

}

};

class MyPrint

{

public:

MyPrint()

{

this->count = 0;

}

void operator()(string test)

{

cout << test << endl;

this->count++;

}

int count;

};

void test01()

{

MyAdd myadd;

cout << myadd(10, 10) << endl;

}

void test02()

{

MyPrint myprint;

myprint("hello world");

myprint("hello world");

myprint("hello world");

cout << "myprint调用的次数为" << myprint .count << endl;

}

int main()

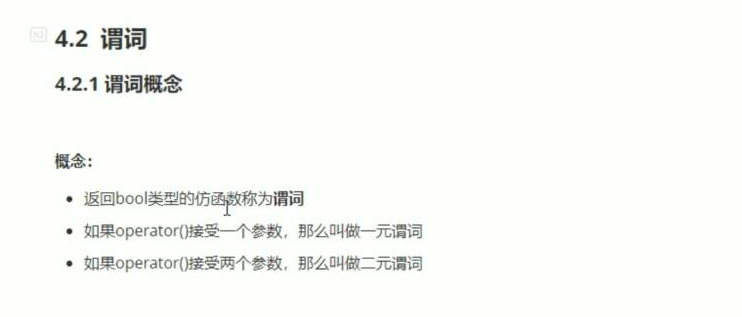
{

test02();

system("pause");

return 0;

}



#include <iostream>

using namespace std;

#include <string>

#include <vector>

#include<algorithm>

class GreaterFive

{

public:

bool operator()(int val)

{

return val > 5;

}

};

void test01()

{

vector<int> v;

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

{

v.push\_back(i);

}

vector<int>::iterator it = find\_if(v.begin(), v.end(), GreaterFive());

if (it == v.end())

{

cout << "未找到" << endl;

}

else

{

cout << "找到了 ，大于5的数字。" << \*it << endl;

}

}

int main()

{

test01();

system("pause");

return 0;

}

二元谓词

#include <iostream>

using namespace std;

#include <string>

#include <vector>

#include<algorithm>

class MyCompare

{

public:

bool operator()(int val1, int val2)//bool 为谓词，两个参数二元谓词。

{

return val1 > val2;

}

};

void test01()

{

vector<int>v;

v.push\_back(10);

v.push\_back(40);

v.push\_back(30);

v.push\_back(20);

v.push\_back(50);

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

for (vector<int>::iterator it =v.begin(); it!=v.end();it++)

{

cout << \*it << endl;

}

cout << endl;

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

cout << "-----------------------" << endl;

for (vector<int>::iterator it = v.begin(); it != v.end(); it++)

{

cout << \*it << endl;

}

}

int main()

{

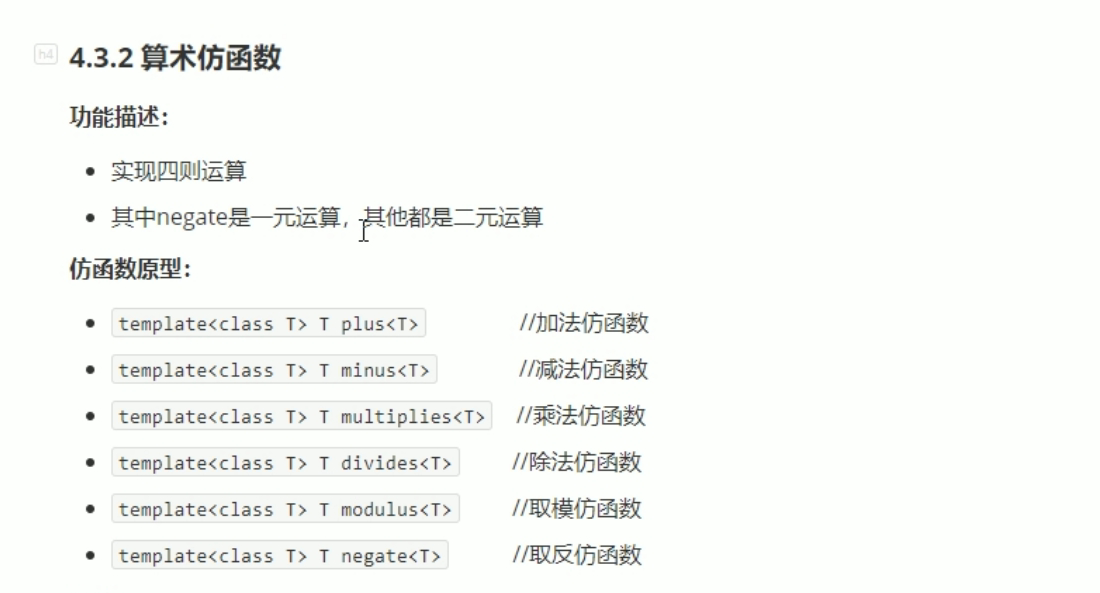
test01();

system("pause");

return 0;

}





#include <iostream>

using namespace std;

#include <string>

#include<functional>

void test01()

{

negate<int> n;

cout << n(50) << endl;

}

void test02()

{

plus<int>p;

cout << p(10, 20) << endl;

}

int main()

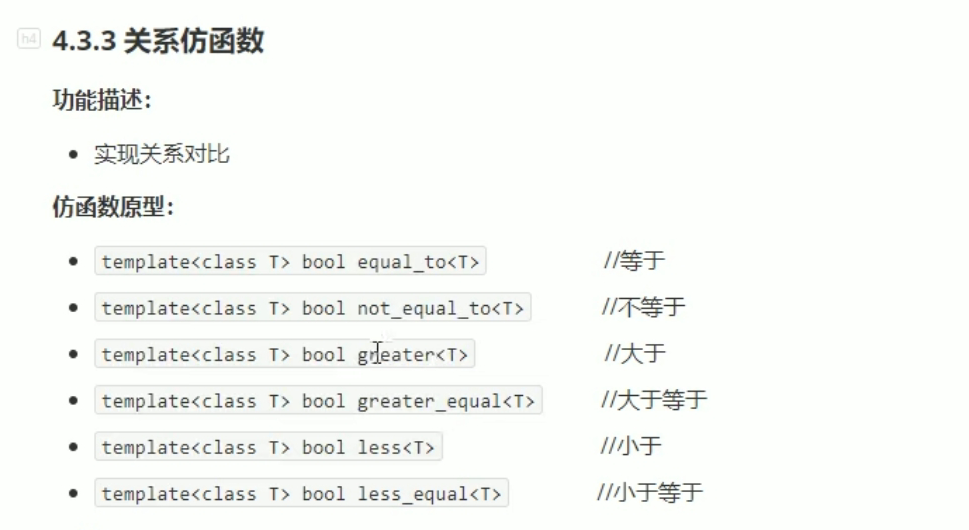
{

test02();

system("pause");

return 0;

}



#include <iostream>

using namespace std;

#include <string>

#include<functional>

#include<vector>

#include<algorithm>

class MyCompare

{

public:

bool operator()(int val1, int val2)//bool 为谓词，两个参数二元谓词。

{

return val1 > val2;

}

};

void test01()

{

vector<int> v;

v.push\_back(10);

v.push\_back(20);

v.push\_back(60);

v.push\_back(40);

v.push\_back(30);

for (vector<int>::iterator it = v.begin(); it != v.end(); it++)

{

cout << \*it << " " ;

}

cout << endl;

//sort(v.begin(), v.end(), MyCompare());

sort(v.begin(), v.end(), greater<int>());//关系仿函数，大于

cout << "-----------------------" << endl;

for (vector<int>::iterator it = v.begin(); it != v.end(); it++)

{

cout << \*it << " " ;

}

}

int main()

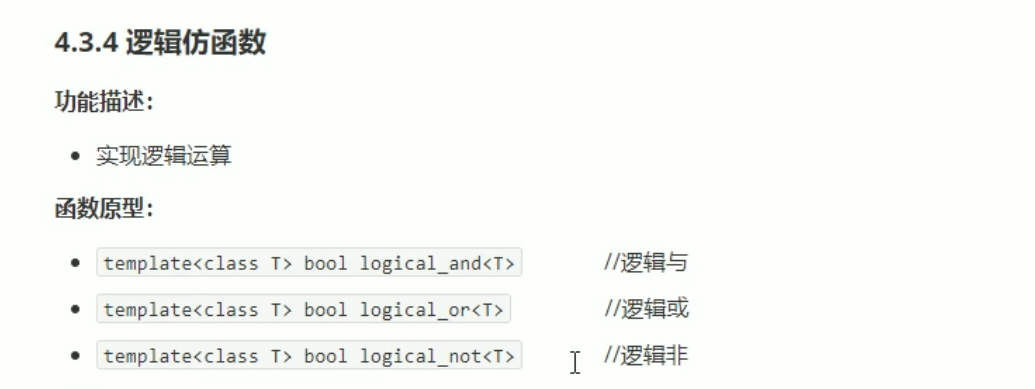
{

test01();

system("pause");

return 0;

}



#include <iostream>

using namespace std;

#include <string>

#include<functional>

#include<vector>

#include<algorithm>

void test01()

{

vector<bool> v;

v.push\_back(true);

v.push\_back(false);

v.push\_back(true);

v.push\_back(false);

for (vector<bool>::iterator it = v.begin(); it != v.end(); it++)

{

cout << \*it << " ";

}

cout << endl;

vector<bool> v2;

v2.resize(v.size());//重新开辟空间

transform(v.begin(), v.end(), v2.begin(), logical\_not<bool>());//搬运算法。

for (vector<bool>::iterator it = v2.begin(); it != v2.end(); it++)

{

cout << \*it << " ";

}

cout << endl;

}

int main()

{

test01();

system("pause");

return 0;

}