

Олимпиада по информатике

Синица Сергей Геннадьевич

Председатель региональной предметно-методической комиссии
Доцент кафедры информационных технологий КубГУ



Язык программирования

C++ или Java

- + Стандартная библиотека структур данных и алгоритмов
- + Документация доступна онлайн
- + Удобные современные инструменты разработки и отладки
- + Возможность работы с компиляторами, используемыми при автоматизированной проверке Яндекс.Контеcт



Компилятор и IDE для C++

GCC и CodeBlocks или Eclipse

- + Кроссплатформенные
- + Быстро работают
- + Компилятор GCC как в Яндекс.Контеcт



CodeBlocks + GCC

Debian/Ubuntu Linux

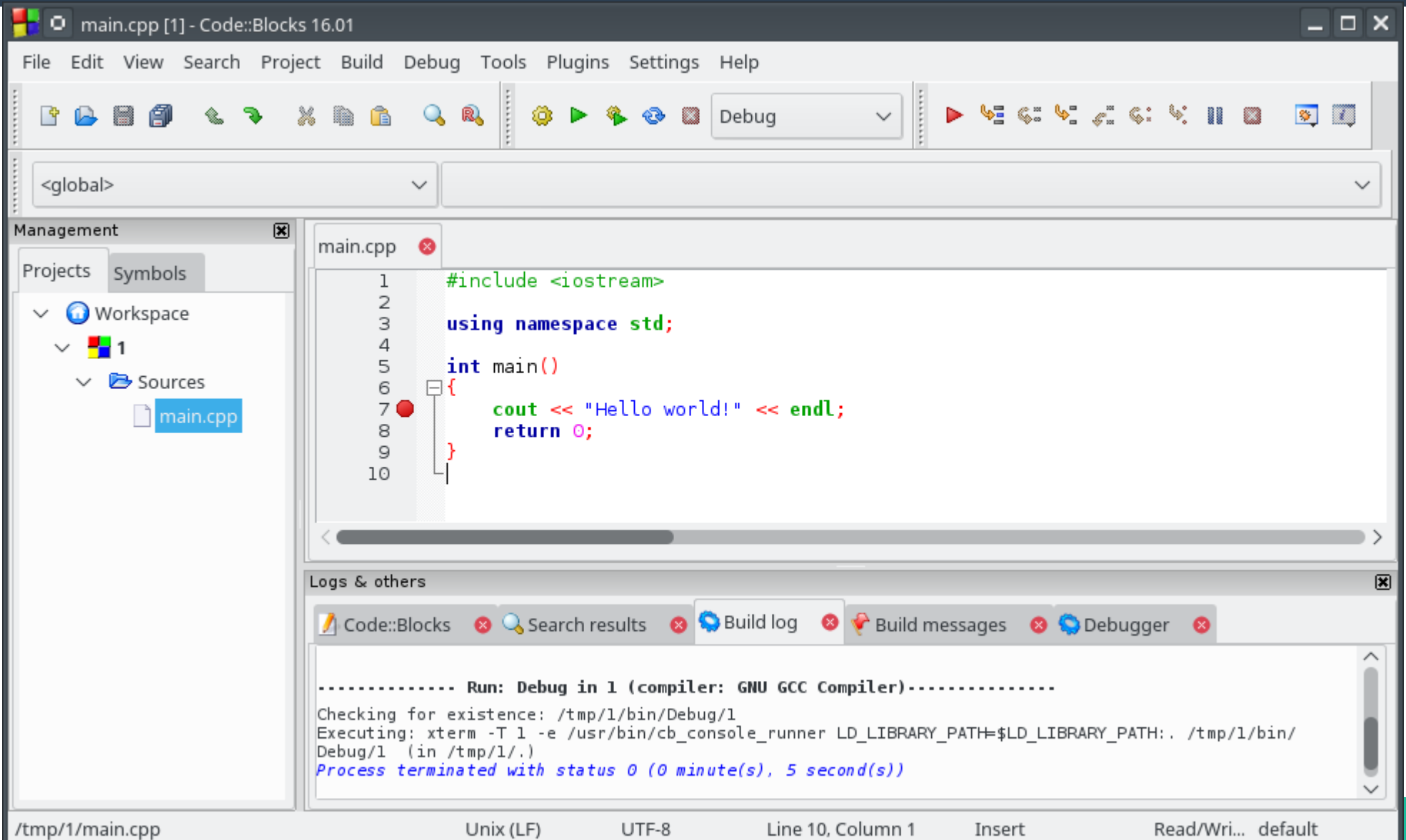
```
sudo apt-get install build-essential  
sudo apt-get install codeblocks
```

Windows

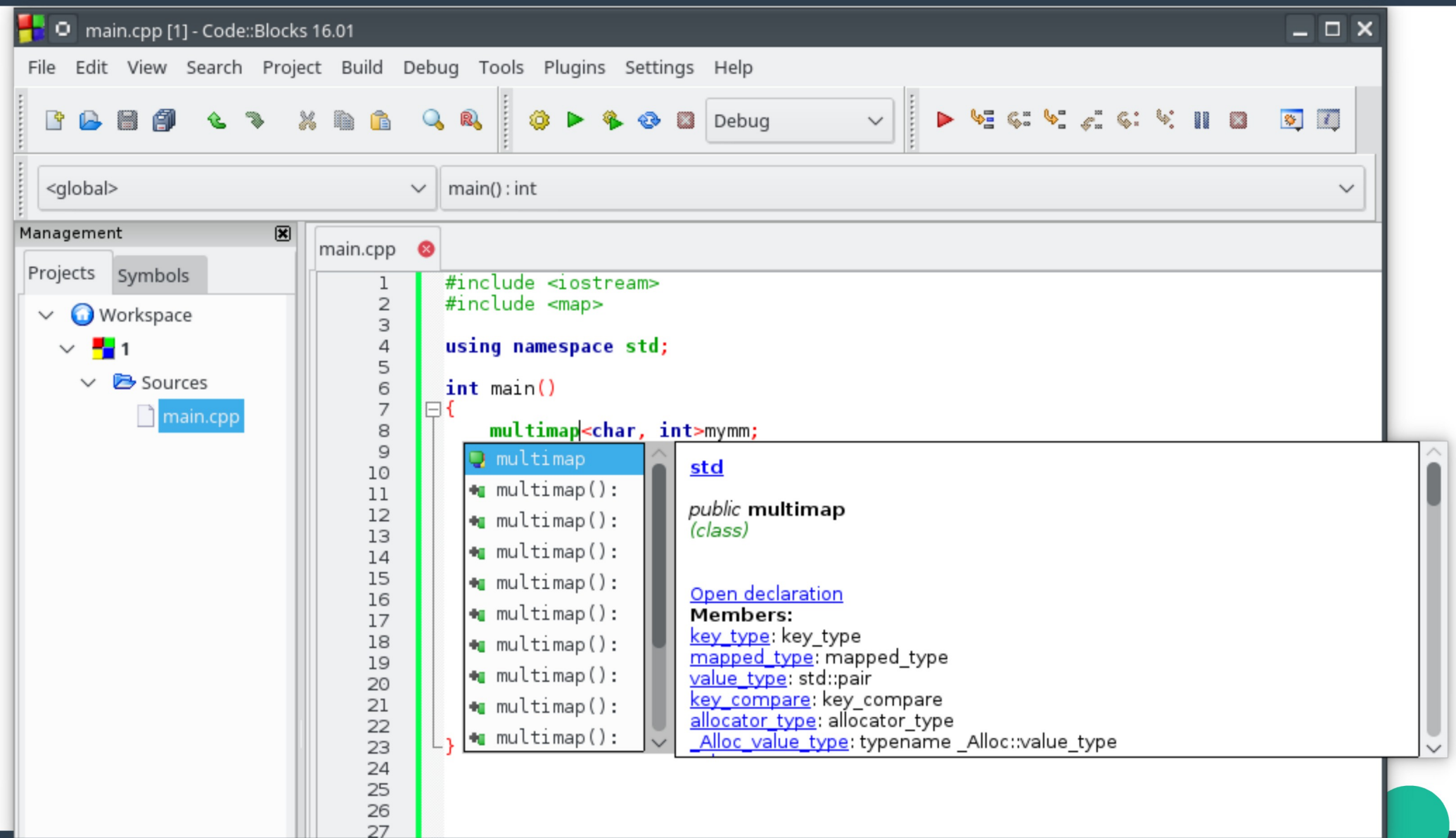
- 1) <http://codeblocks.org>
- 2) Downloads
- 3) Binary release
- 4) codeblocks-17.12mingw-nosetup.zip



CodeBlocks + GCC



CodeBlocks + GCC



C++ 11, третье издание



Hello World!

```
#include <bits/stdc++.h>

using namespace std;

int main()
{
    cout << "Hello world!" << endl;
    return 0;
}
```



IO

```
#include <bits/stdc++.h>

using namespace std;

int main() {
    std::ifstream infile("task1.in");
    int a, b;
    infile >> a;
    infile >> b;
    std::ofstream outfile("task1.out");
    outfile << (a+b);
    return 0;
}
```



STL pair

```
#include <bits/stdc++.h>

using namespace std;

int main() {
    pair <string, double> product1;
    pair <string, double> product2("Помидоры", 2.30);
    pair <string, double> product3(product2);

    product1 = make_pair(string("Лампочки"), 0.99);

    product2.first = "Обувь";
    product2.second = 39.90;

    cout << "Цена " << product1.first << " B $" << product1.second << '\n';
    cout << "Цена " << product2.first << " B $" << product2.second << '\n';
    cout << "Цена " << product3.first << " B $" << product3.second << '\n';

    return 0;
}
```



STL tuple

```
#include <bits/stdc++.h>

using namespace std;

int main () {
    tuple<int,char> foo (10, 'x');
    auto bar = make_tuple ("test", 3.1, 14, 'y');

    get<2>(bar) = 100;

    int myint; char mychar;

    tie (myint, mychar) = foo;
    tie (ignore, ignore, myint, mychar) = bar;

    mychar = get<3>(bar);

    get<0>(foo) = get<2>(bar);
    get<1>(foo) = mychar;

    cout << "foo contains: ";
    cout << get<0>(foo) << ' ';
    cout << get<1>(foo) << '\n';

    return 0;
}
```



STL vector

```
#include <bits/stdc++.h>
using namespace std;

int main ()
{
    vector<int> first;
    vector<int> second (4,100);
    vector<int> third (second.begin(),second.end());
    vector<int> fourth (third);

    cout << second[1];

    // the iterator constructor can also be used to construct from arrays:
    int myints[] = {16,2,77,29};
    vector<int> fifth (myints, myints + sizeof(myints) / sizeof(int) );

    cout << "The contents of fifth are:";
    for (vector<int>::iterator it = fifth.begin(); it != fifth.end(); ++it)
        cout << ' ' << *it;
    cout << '\n';

    return 0;
}
```



STL unordered_map

```
#include <bits/stdc++.h>

using namespace std;

int main ()
{
    unordered_map<string,double> mymap = {
        {"mom",5.4},
        {"dad",6.1},
        {"bro",5.9} };

    string input;
    cout << "who? ";
    getline (cin,input);

    unordered_map<string,double>::const_iterator got = mymap.find (input);

    if ( got == mymap.end() )
        cout << "not found";
    else
        cout << got->first << " is " << got->second;

    cout << std::endl;

    return 0;
}
```



STL multimap

```
#include <bits/stdc++.h>

using namespace std;

int main()
{
    multimap<char, int>mymm;

    mymm.insert(make_pair('x', 10));
    mymm.insert(make_pair('y', 20));
    mymm.insert(make_pair('z', 30));
    mymm.insert(make_pair('z', 40));

    multimap<char, int>::iterator it = mymm.find('x');
    mymm.erase(it);
    mymm.erase(mymm.find('z'));

    cout << "elements in mymm:" << '\n';
    cout << "y=>" << mymm.find('y') -> second << '\n';
    cout << "z=>" << mymm.find('z') -> second << '\n';
    return 0;
}
```



STL bitset

```
#include <bits/stdc++.h>

using namespace std;

int main () {
    bitset<5> foo (string("01011"));

    cout << "foo contains:\n";
    cout << std::boolalpha;
    for (size_t i=0; i<foo.size(); ++i)
        cout << foo.test(i) << '\n';

    return 0;
}
```



STL algorithms: sort, $N * \log(n)$

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    vector<int> intVec = {56, 32, -43, 23, 12, 93, 132, -154};
    vector<string> stringVec = {"Сергей", "Катя", "Алексей", "Лена"};

    // sort using the default operator <
    sort(intVec.begin(), intVec.end());

    for (vector<int>::size_type i = 0; i != intVec.size(); ++i)
        cout << intVec[i] << " ";
    cout << endl;

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

    for (auto s : stringVec)
        cout << s << " ";

    // sort using a standard library compare function object
    sort(stringVec.begin(), stringVec.end(), greater<string>());
    for (auto a : stringVec) {
        cout << a << " ";
    }
    cout << endl;
}
```



STL algorithms: sort, $N * \log(n)$

```
// sort using a custom function object
struct {
    bool operator()(int a, int b) const
    {
        return a < b;
    }
} customLess;
sort(intVec.begin(), intVec.end(), customLess);
for (auto a : intVec) {
    cout << a << " ";
}
cout << '\n';

// sort using a lambda expression
sort(intVec.begin(), intVec.end(), [](int a, int b) {
    return a > b;
});
for (auto a : intVec) {
    cout << a << " ";
}
cout << '\n';

return 0;
}
```



STL binary search

```
#include <iostream>
#include <algorithm>
#include <vector>

using namespace std;

int main()
{
    vector<int> haystack {1, 3, 4, 5, 9};
    vector<int> needles {1, 2, 3};

    for (auto needle : needles) {
        cout << "Searching for " << needle << '\n';
        if (binary_search(haystack.begin(), haystack.end(), needle)) {
            cout << "Found " << needle << '\n';
        } else {
            cout << "no dice!\n";
        }
    }
}
```



Алгоритмы

<http://ru.cppreference.com/w/cpp/algorithm>



Задачи

<http://codeforces.com/problemset>

