# C++ za takmičare

BubbleCup 15 finale, Septembar 2022

Ivan Stošić

# Online/onsite takmičenja

- Različito okruženje
- Različiti ciljevi
- Pristup internetu
- Biblioteka kodova

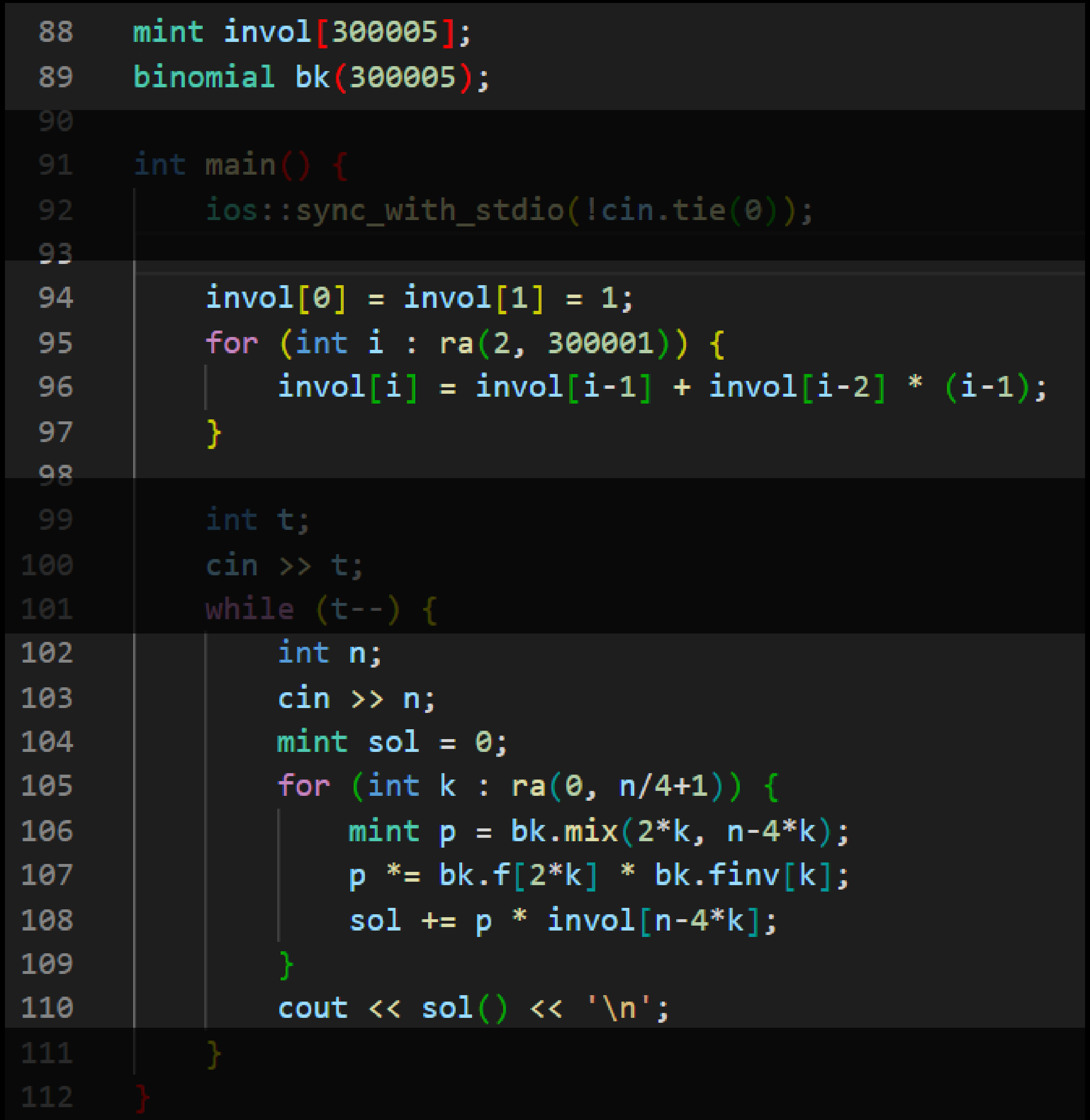
# Razvojno okruženje

- Prilagodite okruženje sebi
- Budite otvoreni ka promenama
- Pazite se na onsite takmičenjima
- Koristite iste kompajlere

# Biblioteka kodova

- Izuzetno važno za online takmičenja!
- Što više, to bolje, ali...
- Dizajn je važan!

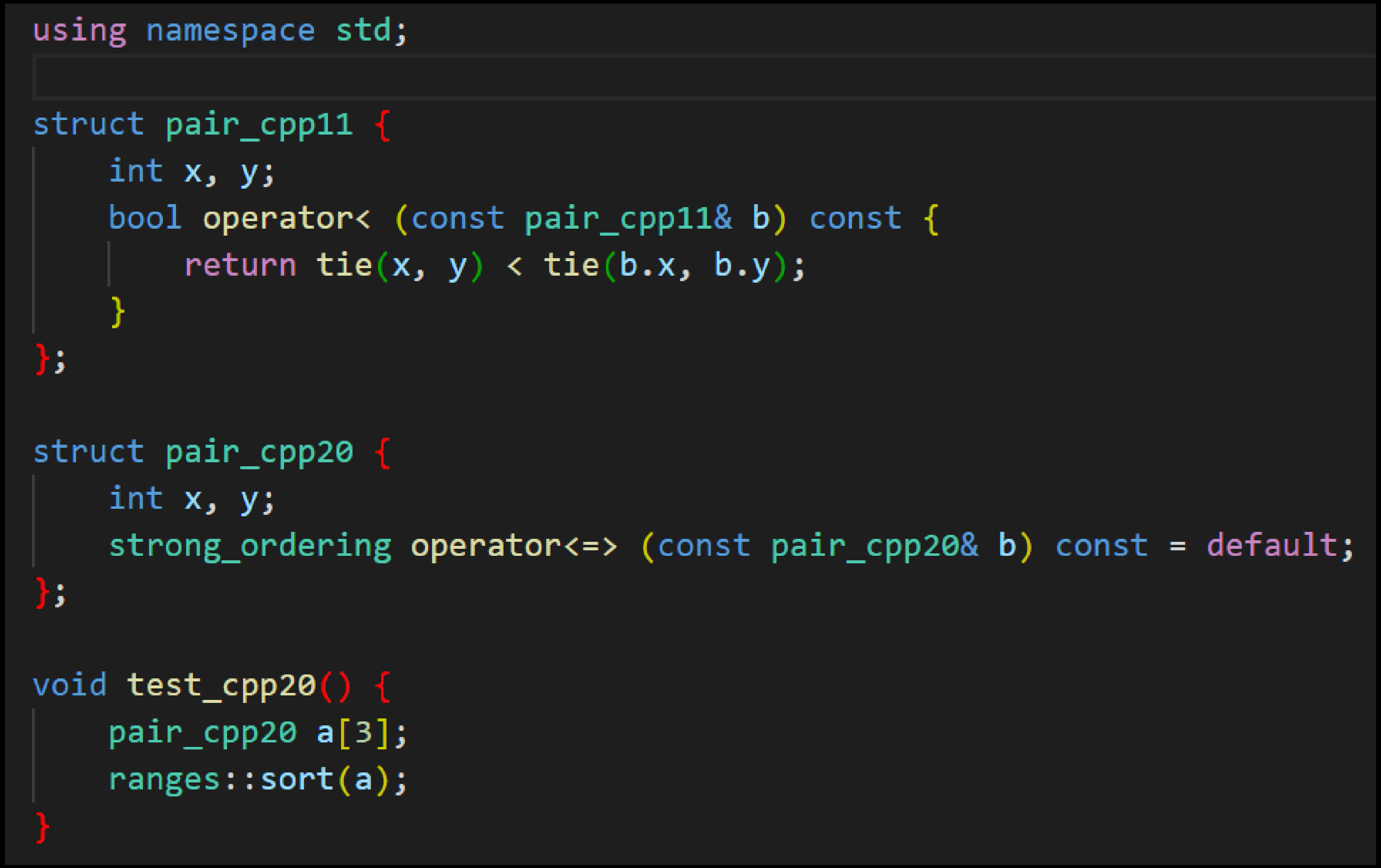
# Biblioteka kodova



```cpp
mint invol[300005];
binomial bk(300005);

int main() {
    ios::sync_with_stdio(!cin.tie(0));

    invol[0] = invol[1] = 1;
    for (int i : ra(2, 300001)) {
        invol[i] = invol[i-1] + invol[i-2] * (i-1);
    }

    int t;
    cin >> t;
    while (t--) {
        int n;
        cin >> n;
        mint sol = 0;
        for (int k : ra(0, n/4+1)) {
            mint p = bk.mix(2*k, n-4*k);
            p *= bk.f[2*k] * bk.finv[k];
            sol += p * invol[n-4*k];
        }
        cout << sol() << '\n';
    }
}
```

# Biblioteka kodova

- Paziti na različite verzije C++-a

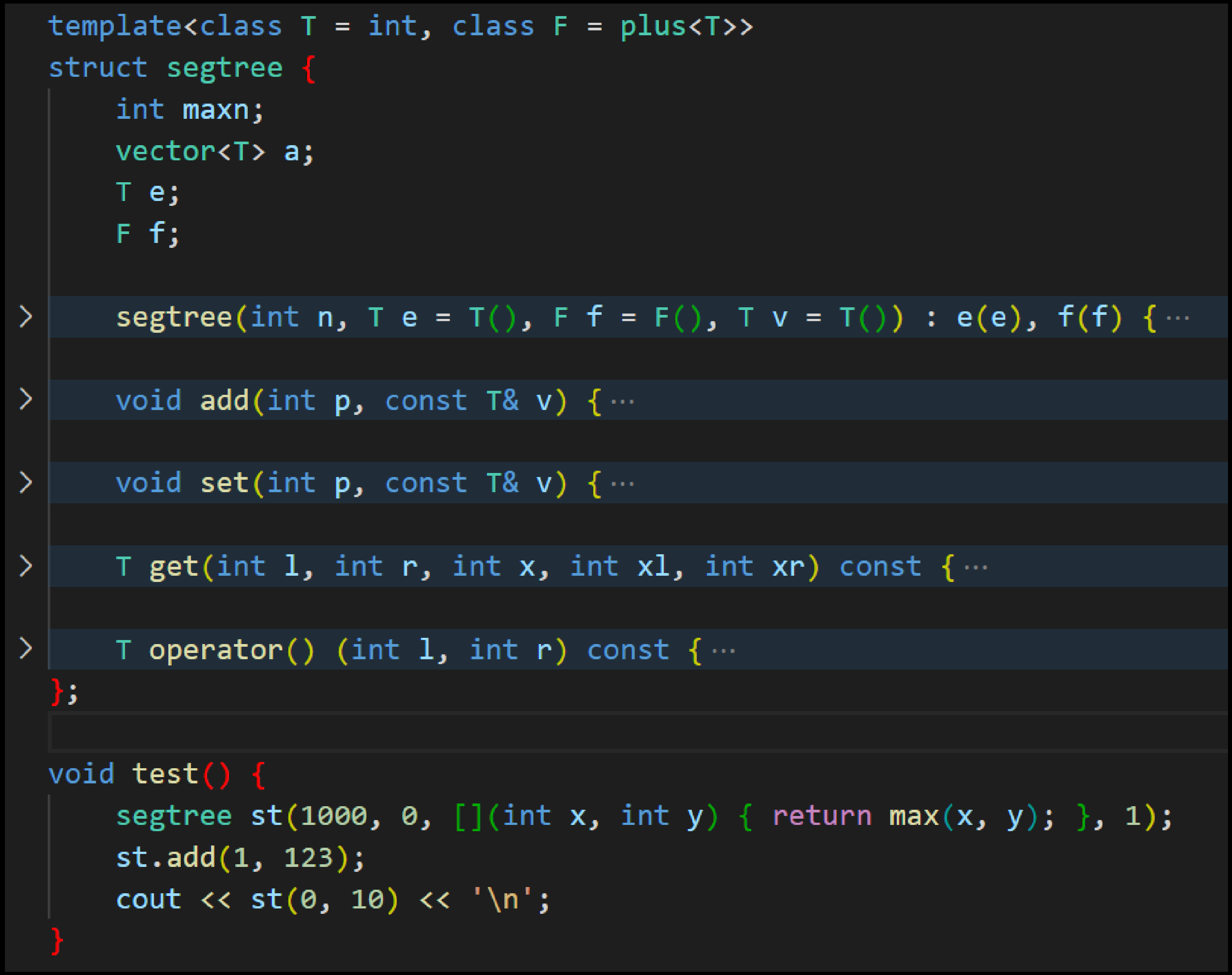


```cpp
using namespace std;

struct pair_cpp11 {
    int x, y;
    bool operator< (const pair_cpp11& b) const {
        return tie(x, y) < tie(b.x, b.y);
    }
};

struct pair_cpp20 {
    int x, y;
    strong_ordering operator<=> (const pair_cpp20& b) const = default;
};

void test_cpp20() {
    pair_cpp20 a[3];
    ranges::sort(a);
}
```

# Šablonske funkcije i klase

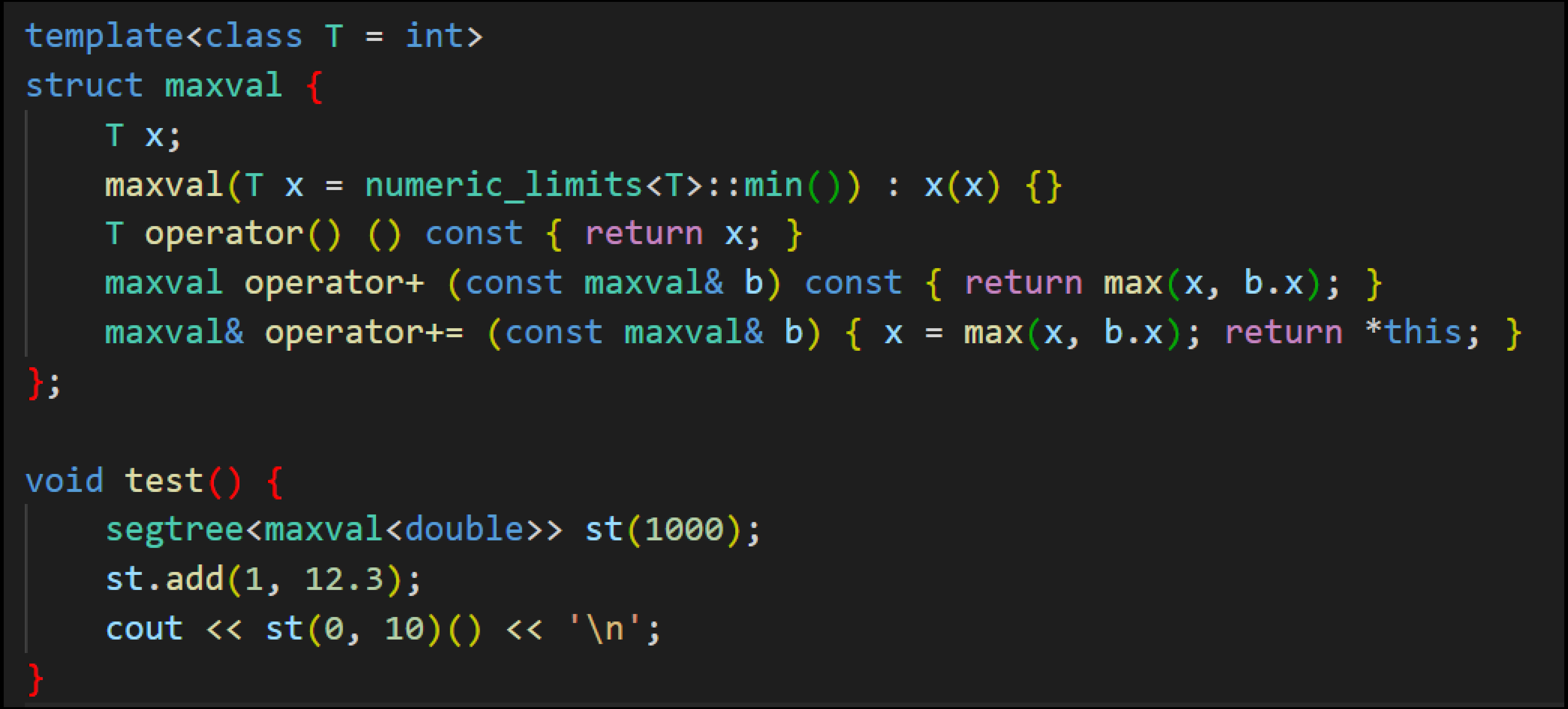
- Svi ih koristimo, a ko ih implementira?
- Korisno onsite, nezamenljivo online

# Primer – segmentno stablo



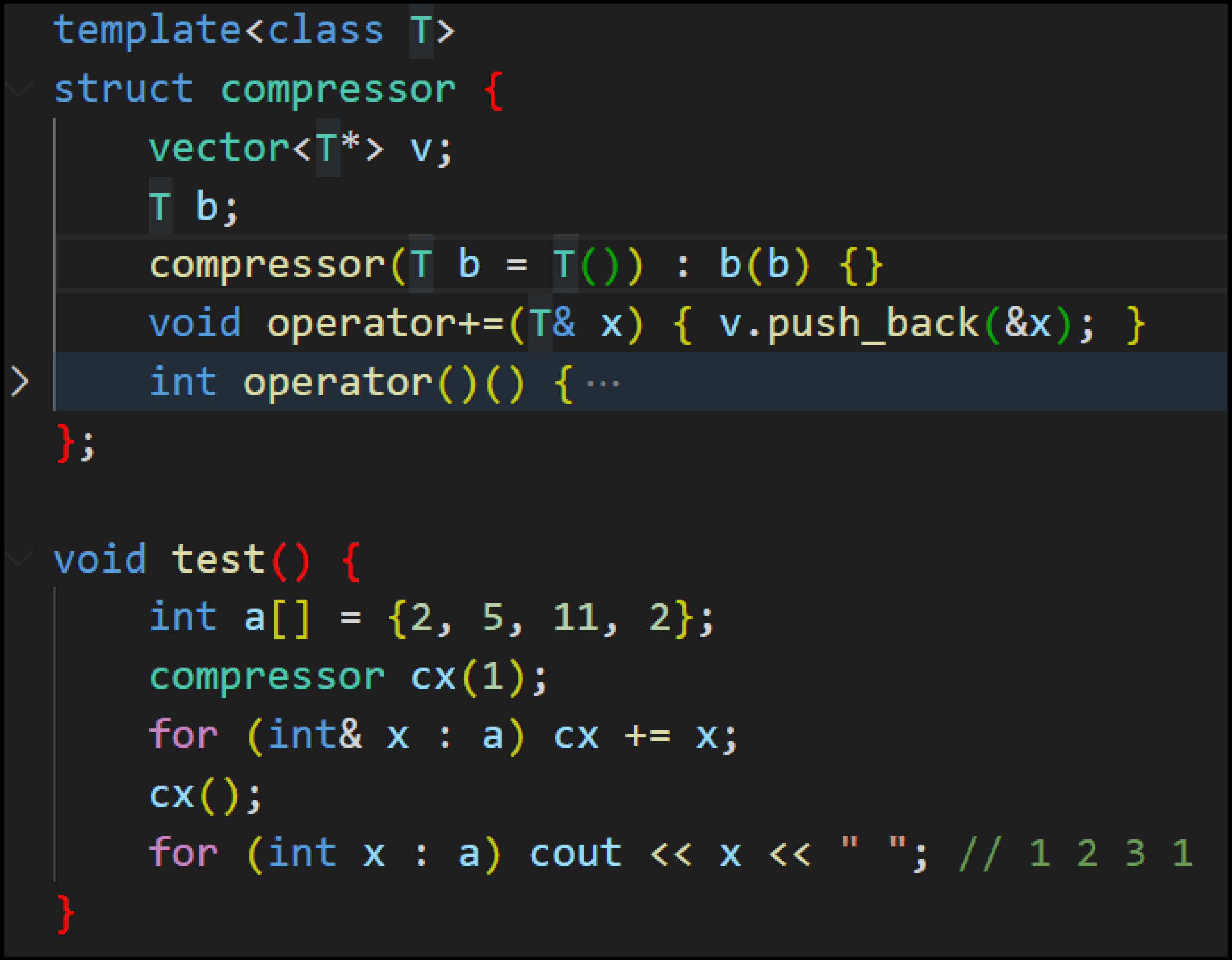
```
template<class T = int, class F = plus<T>>
struct segtree {
    int maxn;
    vector<T> a;
    T e;
    F f;

    segtree(int n, T e = T(), F f = F(), T v = T()) : e(e), f(f) {⋯

    void add(int p, const T& v) {⋯

    void set(int p, const T& v) {⋯

    T get(int l, int r, int x, int xl, int xr) const {⋯

    T operator() (int l, int r) const {⋯
};

void test() {
    segtree st(1000, 0, [](int x, int y) { return max(x, y); }, 1);
    st.add(1, 123);
    cout << st(0, 10) << '\n';
}
```

# Primer – segmentno stablo



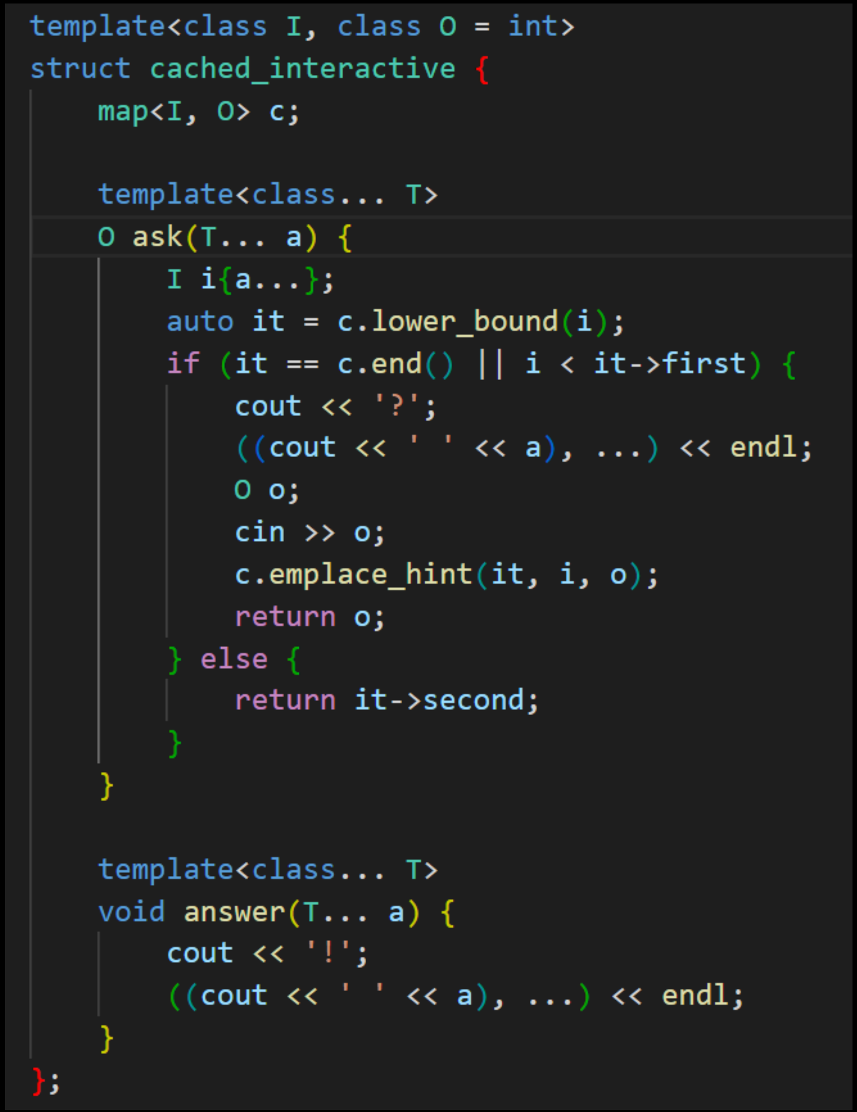
```cpp
template<class T = int>
struct maxval {
    T x;
    maxval(T x = numeric_limits<T>::min()) : x(x) {}
    T operator() () const { return x; }
    maxval operator+ (const maxval& b) const { return max(x, b.x); }
    maxval& operator+= (const maxval& b) { x = max(x, b.x); return *this; }
};

void test() {
    segtree<maxval<double>> st(1000);
    st.add(1, 12.3);
    cout << st(0, 10)() << '\n';
}
```

# Kompresija koordinata

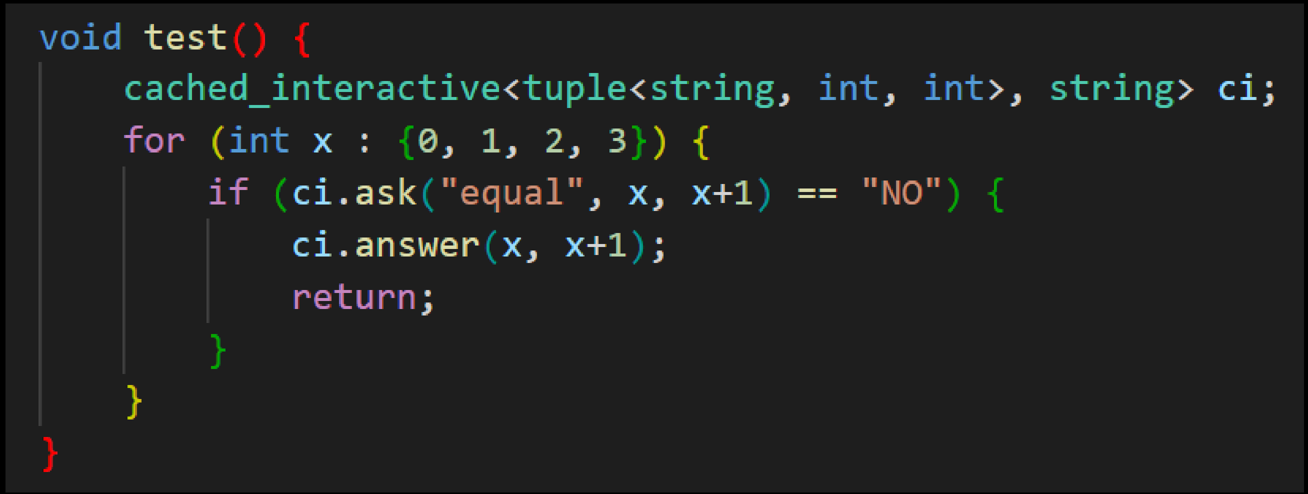


```
template<class T>
struct compressor {
    vector<T*> v;
    T b;
    compressor(T b = T()) : b(b) {}
    void operator+=(T& x) { v.push_back(&x); }
    int operator()() {···
};

void test() {
    int a[] = {2, 5, 11, 2};
    compressor cx(1);
    for (int& x : a) cx += x;
    cx();
    for (int x : a) cout << x << " "; // 1 2 3 1
}
```

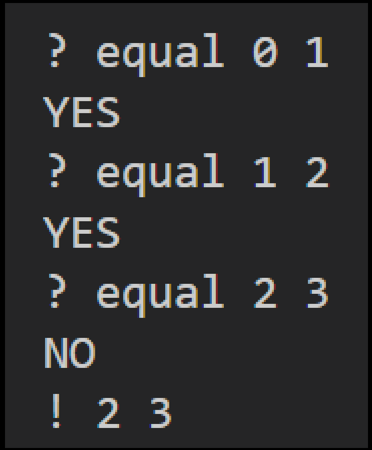
# Primer – interaktivni zadaci



```cpp
template<class I, class O = int>
struct cached_interactive {
    map<I, O> c;

    template<class... T>
    O ask(T... a) {
        I i{a...};
        auto it = c.lower_bound(i);
        if (it == c.end() || i < it->first) {
            cout << '?';
            ((cout << ' ' << a), ...) << endl;
            O o;
            cin >> o;
            c.emplace_hint(it, i, o);
            return o;
        } else {
            return it->second;
        }
    }

    template<class... T>
    void answer(T... a) {
        cout << '!';
        ((cout << ' ' << a), ...) << endl;
    }
};
```



```cpp
void test() {
    cached_interactive<tuple<string, int, int>, string> ci;
    for (int x : {0, 1, 2, 3}) {
        if (ci.ask("equal", x, x+1) == "NO") {
            ci.answer(x, x+1);
            return;
        }
    }
}
```



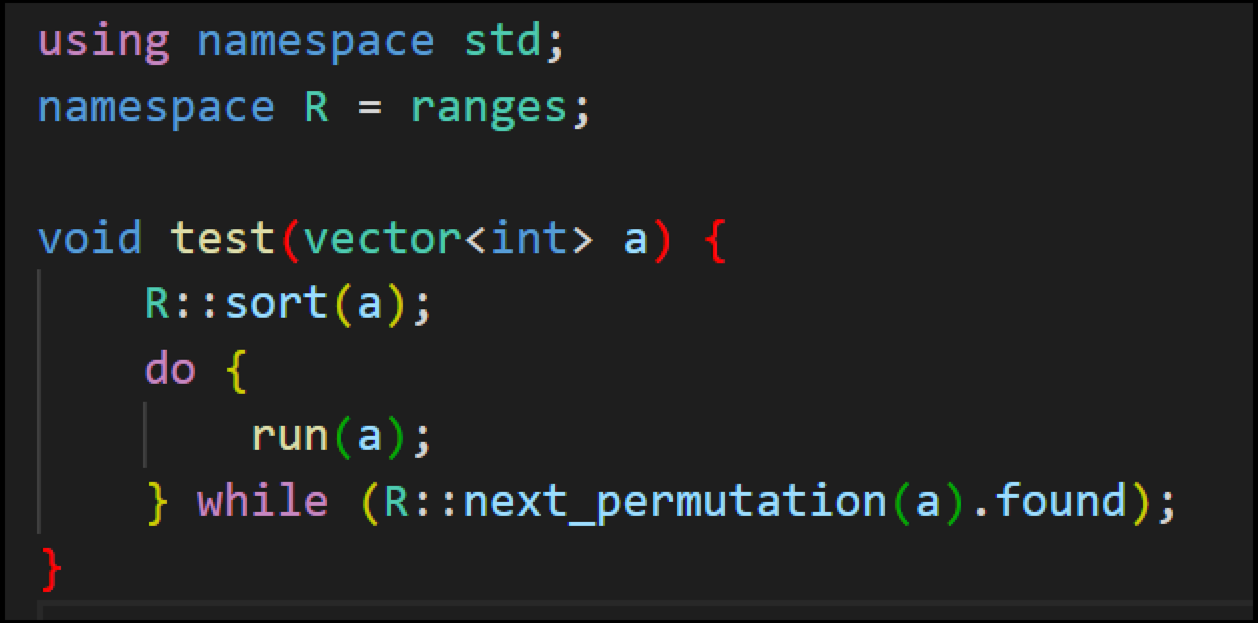
```
? equal 0 1
YES
? equal 1 2
YES
? equal 2 3
NO
! 2 3
```

# Novine u C++20

- std::ranges (algoritmi)
- std::ranges::views (transformacije)
- std::span
- operator<=>

# std::ranges

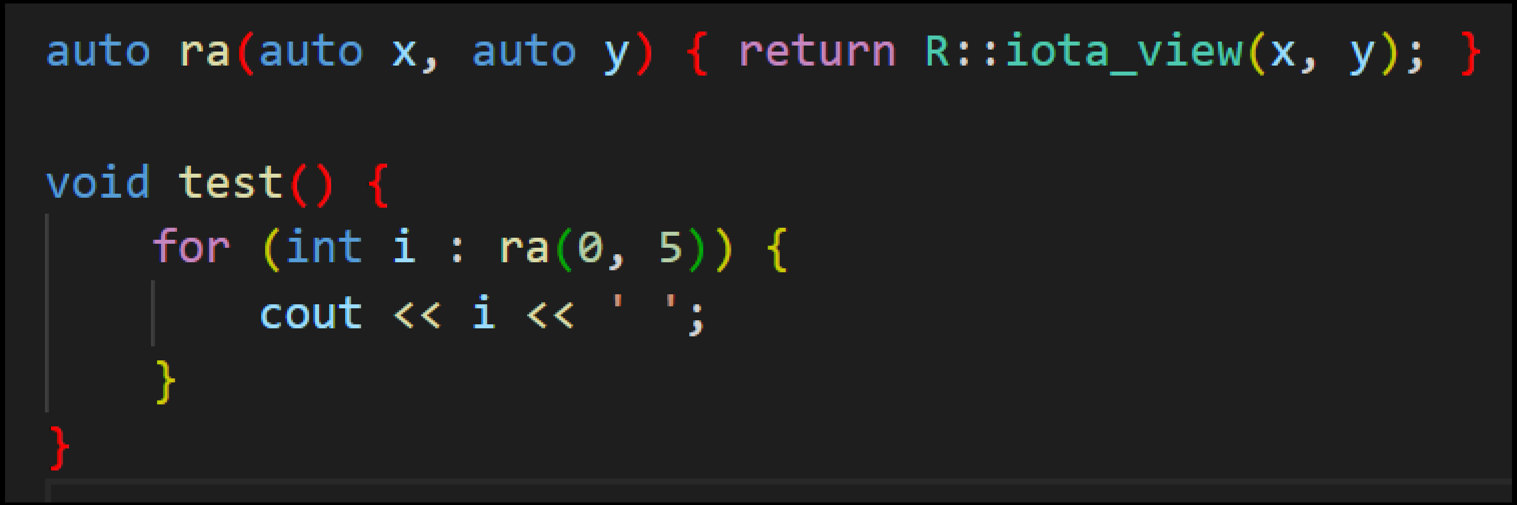
- Kraći pozivi funkcija
- Postoje razlike



```
using namespace std;
namespace R = ranges;

void test(vector<int> a) {
    R::sort(a);
    do {
        run(a);
    } while (R::next_permutation(a).found);
}
```

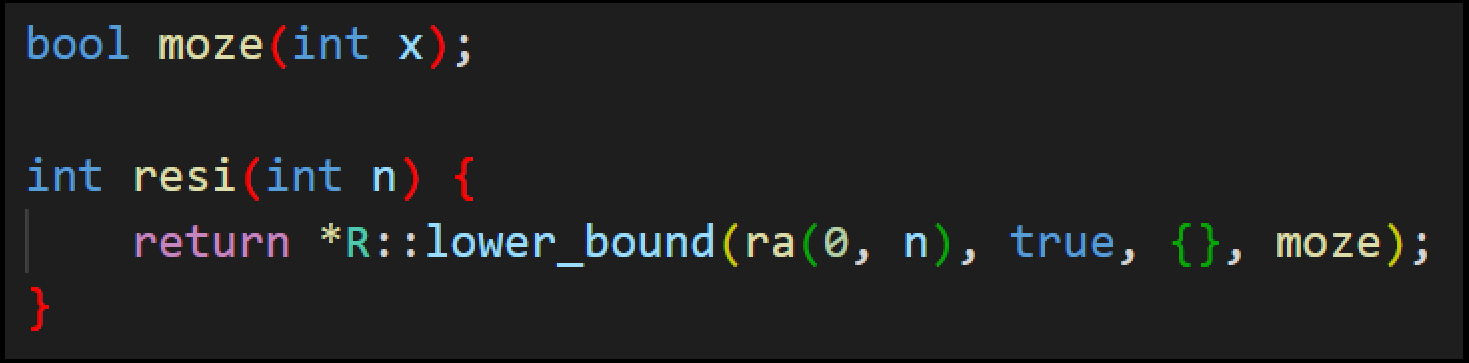
# std::ranges::views

- R::views::iota, odnosno R::iota_view



```
auto ra(auto x, auto y) { return R::iota_view(x, y); }

void test() {
    for (int i : ra(0, 5)) {
        cout << i << ' ';
    }
}
```

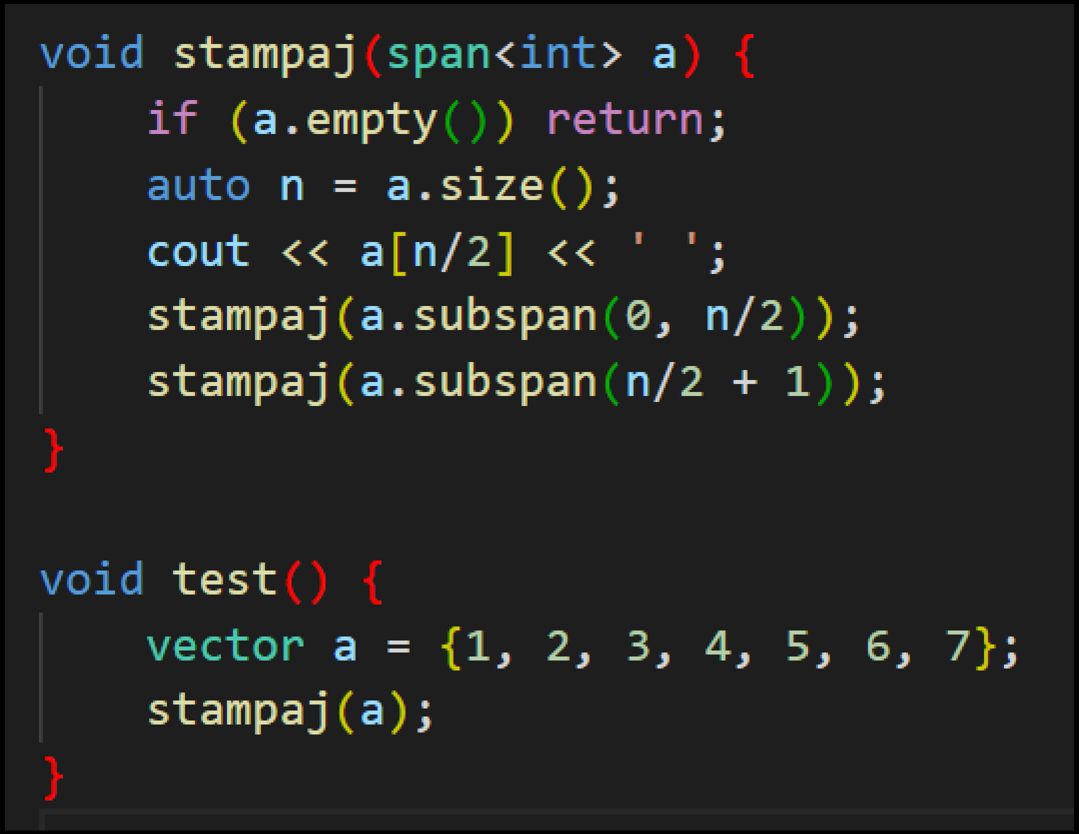
# Binarna pretraga po rešenju



```
bool moze(int x);

int resi(int n) {
    return *R::lower_bound(ra(0, n), true, {}, moze);
}
```

# std::span

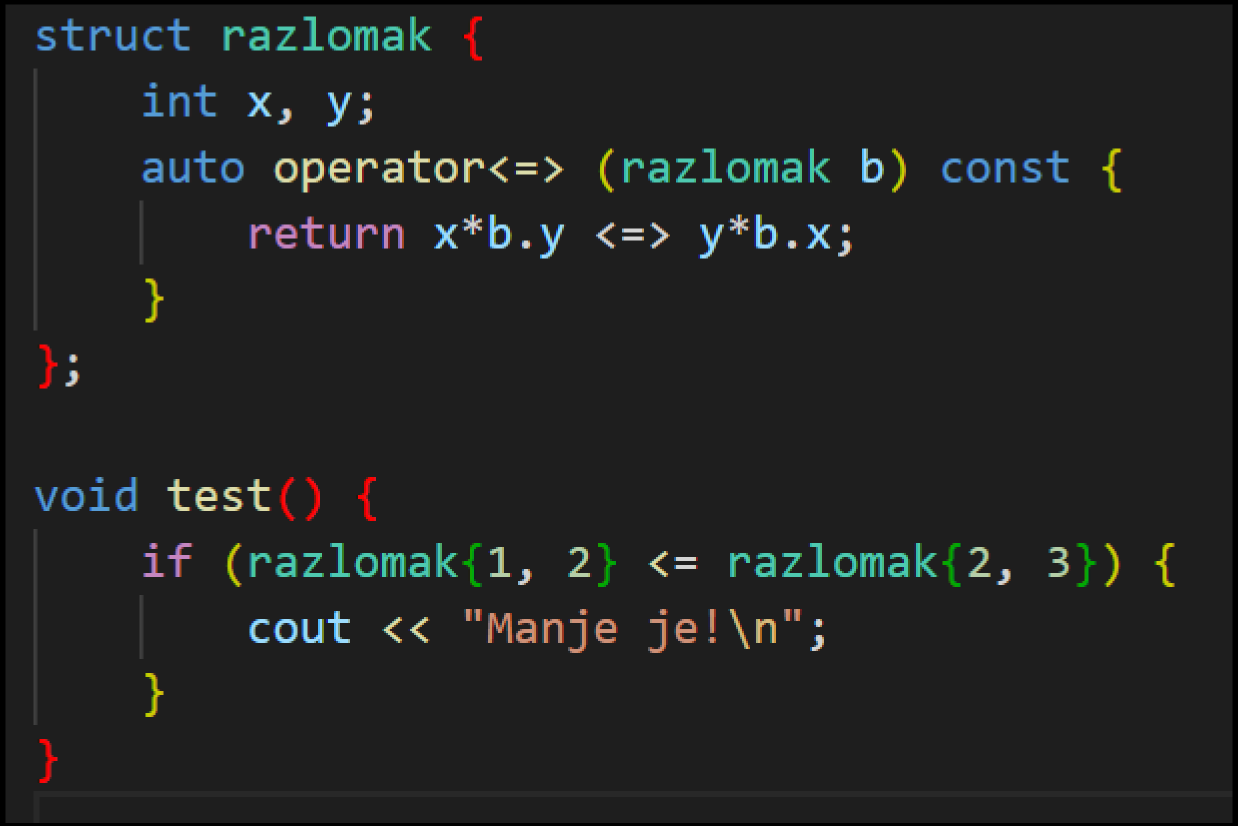
- Pogled na niz elemenata u memoriji



```
void stampaj(span<int> a) {
    if (a.empty()) return;
    auto n = a.size();
    cout << a[n/2] << ' ';
    stampaj(a.subspan(0, n/2));
    stampaj(a.subspan(n/2 + 1));
}

void test() {
    vector a = {1, 2, 3, 4, 5, 6, 7};
    stampaj(a);
}
```

# operator <=>

- Zamenjuje <, >, <=, >=



```cpp
struct razlomak {
    int x, y;
    auto operator<=> (razlomak b) const {
        return x*b.y <=> y*b.x;
    }
};

void test() {
    if (razlomak{1, 2} <= razlomak{2, 3}) {
        cout << "Manje je!\n";
    }
}
```

# Pitanja?

# Hvala na pažnji!

- github.com/ivan100sic/competelib-snippets
- github.com/ivan100sic/bc-15-lecture