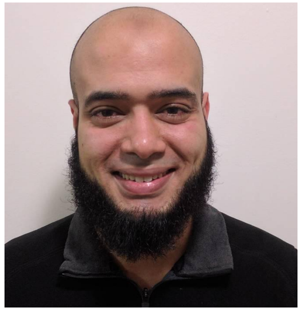
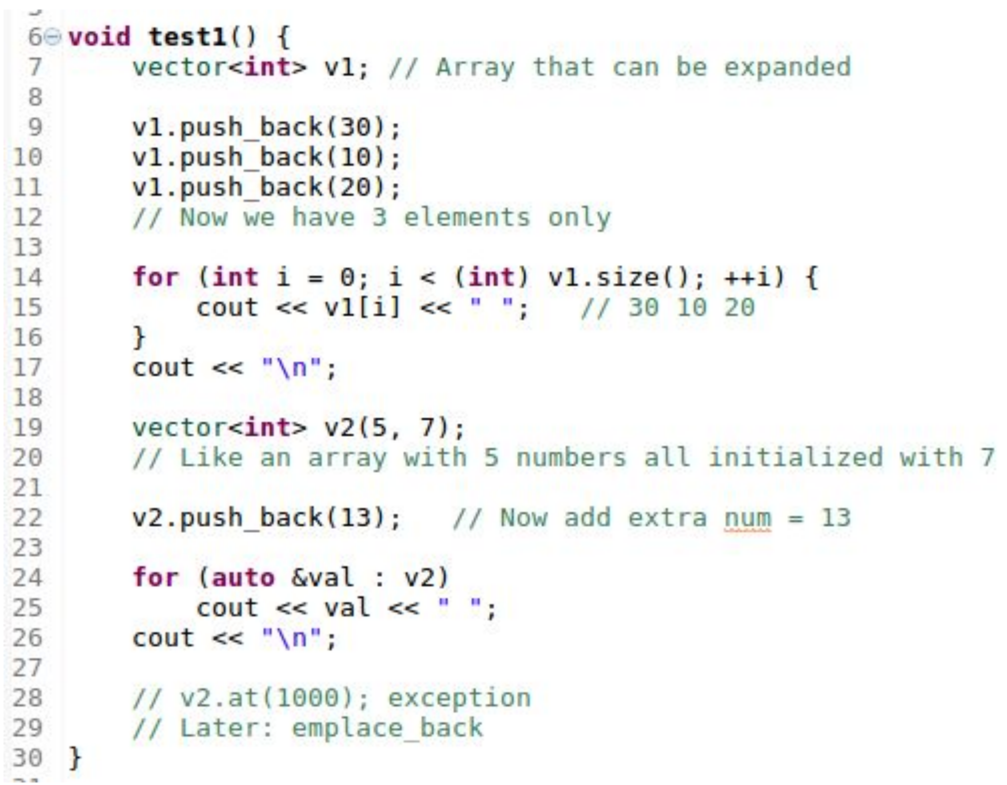
C++ Programming
STL Vector 1

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

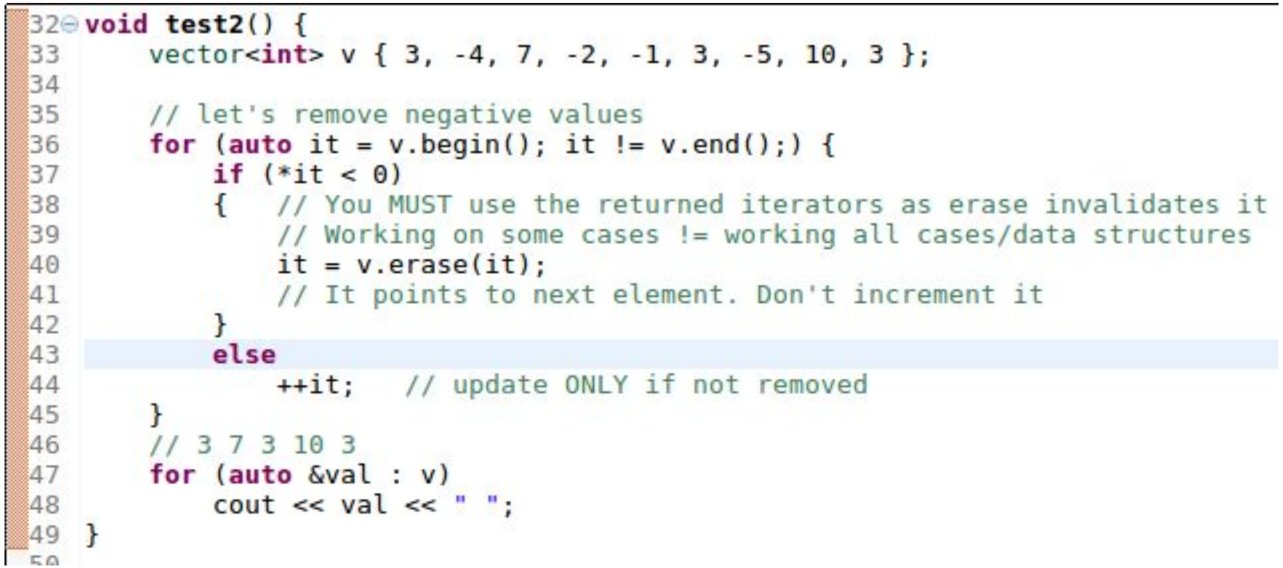


Vector



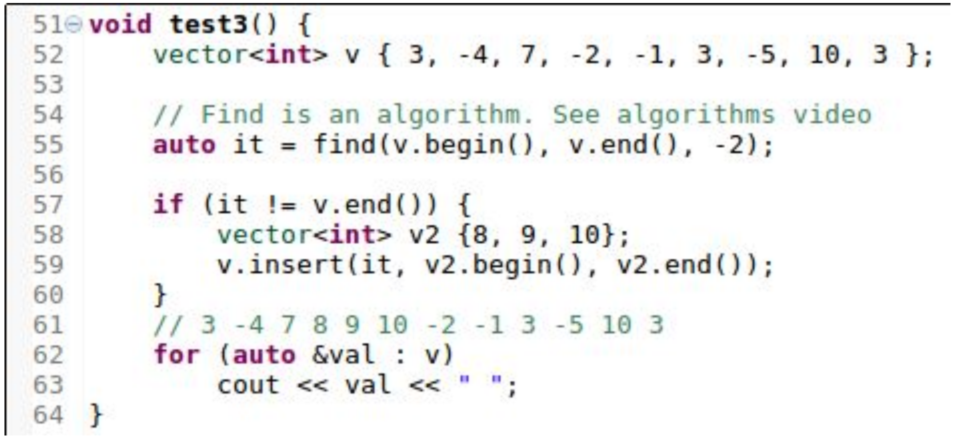
```
void test1() {
    vector<int> v1; // Array that can be expanded

    v1.push_back(30);
    v1.push_back(10);
    v1.push_back(20);
    // Now we have 3 elements only

    for (int i = 0; i < (int) v1.size(); ++i) {
        cout << v1[i] << " ";   // 30 10 20
    }
    cout << "\n";

    vector<int> v2(5, 7);
    // Like an array with 5 numbers all initialized with 7

    v2.push_back(13);   // Now add extra num = 13

    for (auto &val : v2)
        cout << val << " ";
    cout << "\n";

    // v2.at(1000); exception
    // Later: emplace_back
}
```

Vector



```
void test2() {
    vector<int> v { 3, -4, 7, -2, -1, 3, -5, 10, 3 };

    // let's remove negative values
    for (auto it = v.begin(); it != v.end();) {
        if (*it < 0)
        {   // You MUST use the returned iterators as erase invalidates it
            // Working on some cases != working all cases/data structures
            it = v.erase(it);
            // It points to next element. Don't increment it
        }
        else
            ++it;   // update ONLY if not removed
    }
    // 3 7 3 10 3
    for (auto &val : v)
        cout << val << " ";
}
```

Vector



```
void test3() {
    vector<int> v { 3, -4, 7, -2, -1, 3, -5, 10, 3 };

    // Find is an algorithm. See algorithms video
    auto it = find(v.begin(), v.end(), -2);

    if (it != v.end()) {
        vector<int> v2 {8, 9, 10};
        v.insert(it, v2.begin(), v2.end());
    }
    // 3 -4 7 8 9 10 -2 -1 3 -5 10 3
    for (auto &val : v)
        cout << val << " ";
}
```

How vector works?

- Inside the vector there is an array of some size. Let’s call it int capacity
  - E.g. Initially capacity = 200;
- Let’s say you push_back 10 elements
  - Now size = 10. Capacity is 200
- Let’s say you pushed another 190 elements
  - Now size = capacity = 200
- Let’s add another 20 elements
  - Vector creates a new array with some bigger capacity, e.g. capacity = 400
  - Copy old 100 elements. Add new 20 elements. Now: size = 220. Capacity = 400
- Performance Tips
  - Pushing a lot is expensive. Know size? vector<int> v(1000000);
  - Know initial possible growth and seems big value? vector<int> v;   v.reserve(50000);

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."