2023 11 06

std::vector

```
class Myclass
{
    public:
        Myclass(int);
};

int main()
{
    using namespace std;

    vector<int> ivec(100);

    auto size = ivec.size();
    auto cap = ivec.capacity();

    bool isEmpty = ivec.empty();

    ivec.reverse(6'000'000); // sürekli realloction yapmaması için

    // vector ctor
    vector<int> ivec; // default ctor : boş vector
    vector<int> ivec1(20); // value init

    vector<Myclass> mvec(20); // illegal çünkü default ctor'u yok


    vector<int> vec1(12); // 12 tane 0 var
    vector<int> vec2{ 12 }; // 1 tane 12 tane var

    // range ctor
    list<int> ilist{ 2, 5, 6, 1, 9, 1};
    vector<int> ivec2{ilist.begin(), ilist.end()};
    vector<double> dvec(ilist.begin(), ilist.end()); // legal

    const char const p[] = { "furkan", "emre", "gano"};
    vector<string> svec(begin(p), end(p)); // legal
}
```

```
// range assign
int main()
{
    using namespace std;
    vector<int> ivec;

    rfill(ivec, 100, Irand{ 0, 40});
    print(ivec);

    auto myset = set<int>{ ivec.begin(), ivec.end() };

    // range assign
    ivec.assign(myset.begin(), myset.end());
}
```

```cpp
int main()
{
    using namespace std;

    list mylist{ 2, 5, 1, 3, 2, 7, 1, 8};

    // auto -> vector<list<int>::iterator> myvec
    auto myvec= { mylist.begin(), mylist.end() };
}
```

```cpp
using namespace std;
void foo(vector<string>&& x)
{
    auto y = std::move(x) // taşıma var
}

int main()
{
    vector<string> svec(100'000);

    foo(std::move(svec)); // taşıma yok burada
}
```

Vectorlerin Elemanlarına Erişme

```cpp
// Vectorlerin Elemanlarına Erişme
using namespace std;
int main()
{
    vector<string> svec{ "burhan", "ferhan", "nurhan", "alphan" };
    // Eğer vector const olsaydı const overloading fonksiyonlar çağrılacaktı.
    svec[2] = "oğuzhan";
    svec.at(2);
    try
    {
        svec.at(23);
    }
    catch (const std::out_of_range&)
    {
        // exception gönderir
    }

    svec.front() = 'can'; // ilk veri
    svec.back() = 'emre'; // son veri

    for (size_t{}; i < svec.size(); ++i)
    {
        svec[i]; // böyle erişebiliriz
    }
}
```

std::find_if

```cpp
// range base loop
using namespace std;
int main()
{
    vector<string> svec{};

    rfill(svec, 100, [] {return rname() + ' ' + rfname();} );

    auto iter = find_if(svec.begin(), svec.end(),
    [](const string &s) {return s.contains('e'); );

    if (iter != svec.end())
    {
        while (iter != svec.end())
        {
            cout << *++iter << "\n";
        }
    }
}
```

```cpp
// range base loop (reverse iteration)
 using namespace std;
int main()
{
    vector<string> svec{};
    rfill(svec, 100, [] {return rname() + ' ' + rfname();} );

    for (auto riter = svec.crbegin(); riter != svec.crend(); ++riter)
            cout << *riter << '\n';
}
```

Vector'e Eleman Ekleme Fonksiyonları

```cpp
// Vector Ekleme Fonksiyonları
template<typename T>
class Vector
{
    public:
        void push_back(const T& r)
        {
            new[adress] T(r); // l value argumanda
        }
        void push_back(T&& r)
        {
            new[adress] T(std::move(r)); // r value argumanda
        }

        template<typename ...U>
        void emplace_back(U&& ...args)
        {
            new[adress]T(std::forward<U>(args)...);
        }
};
```

```cpp
int main()
{
    vector<string> svec{ "ali", "kemal", "naz", "derin" };

    svec.push_back("murat"); // sondan ekleme

    svec.insert(svec.begin(), "deniz"); // başa ekler

    svec.insert(next(svec.begin()), "deniz"); // 2'ye ekler

    // init list overload
    svec.insert(next(svec.begin()), { "deniz", "taylan" ,"emre" }); // 2'ye ekler

    // range overload
    list<string> female_list {"ganoş", "özge", "merve"};
    svec.insert(svec.begin(), female_list.begin(), female_list.end());

    // insert fonksiyonları insert edilmiş üyenin konumu döndürür
    vector<string> svec1 { "ali", "gul", "tan", "eda"};
    auto iter = svec.insert(svec.begin() + 1, "NURULLAH");
    cout << *iter << "\n"; // NURULLAH
}
```

Vector Atama Fonksiyonları

```cpp
// vector atama fonksiyonları
using namespace std;
int main()
{
    vector<int> ivec(100);

    cout << "ivec.size() = " << ivec.size() << "\n";
    ivec = { 2, 3, 5, 1, 7, 11}; // size = 6

    ivec.assign({2, 3, 1, 4, 1}); // init list overload
}
```