STL

Тема 3. Контейнеры.

Часть 1. Последовательные контейнеры

Последовательные контейнеры

- Vector
- Deque
- List

Vector

Достоинства:

- Быстрый произвольный доступ
- Быстрая вставка\удалении в конец
- Итератор с произвольным доступом

```
template < class Type, class Allocator = allocator<Type> >
class vector
```

Vector. Typedefs

Typedef	Описание
const_iterator	Константный итератор
const_pointer	*const Type*.* В общем случае определен аллокатором
const_reference	*const Type&.* В общем случае определен аллокатором
const_reverse_iterator	Константный обратный итератор
difference_type	Знаковый целочисленный тип, который может определеть разность между двумя итераторами
iterator	итератор
pointer	*Type*.* В общем случае определен аллокатором
reference	*Type&.* В общем случае определен аллокатором
reverse_iterator	Обратный итератор
size_type	Тип представляющий количество элеменотов
value_type	*Type.*

Vector. Constructor

```
vector( );
```

```
explicit vector( size_type _Count );
```

```
vector( size_type _Count, const Type& _Val );
```

```
vector( const vector<Type, Allocator>& _Right );
```

```
template<class InputIterator>
vector( InputIterator _First, InputIterator _Last );
```

Vector. Constructor

```
using namespace std;

vector <int> v0;                    // Create an empty vector v0

vector <int> v1( 3 );               // Create a vector v1 with 3 elements of default value 0

vector <int> v2( 5, 2);             // Create a vector v2 with 5 elements of value 2

vector <int> v3( v2 );              // Create a copy, vector v3, of vector v2

 vector <int> v4( v3.begin( ) + 1, v3.begin( ) + 3 );  // Create a vector v4 by copying the
                                                       //range v3[_First, _Last)
```

Vector:: push_back\pop_back

void push_back(const Type& _Val);

Вставка элемента в конец вектора за O(1)

void pop_back( );

Удаление из конца вектора за O(1)

```
int main( )
{
        using namespace std;
        vector <int> v1;

        v1.push_back( 1 );
        v1.push_back( 2 );
        v1.pop_back( );
}
```

Vector::insert

```
iterator insert( iterator _Where, const Type& _Val );

void insert( iterator _Where, size_type _Count, const Type& _Val );

template<class InputIterator>
void insert( iterator _Where, InputIterator _First, InputIterator _Last );
```

```
int main( )
{
        vector <int> v1; vector <int>::iterator Iter;
        v1.push_back( 10 ); v1.push_back( 20 ); v1.push_back( 30 );

        v1.insert( v1.begin( ) + 1, 40 );
        v1.insert( v1.begin( ) + 2, 4, 50 );
        v1.insert( v1.begin( )+1, v1.begin( )+2, v1.begin( )+4 );
}
```

reverse / capacity

size_type capacity( ) const;

void reserve(size_type _Count **);**

```
int main( )
{
        vector <int> v1; //vector
        <int>::iterator Iter;

         v1.push_back( 1 );
         cout << "Current capacity of v1 = " << v1.capacity( ) << endl;

         v1.reserve( 20 );
         cout << "Current capacity of v1 = " << v1.capacity( ) << endl;
}
```

Erase

iterator erase(iterator _Where **);**
iterator erase(iterator _First**, iterator** _Last **);**

Удаление диапазона элементов из произвольного места в векторе

```
int main( )
{
        vector <int> v1;
        v1.push_back( 10 );
        v1.push_back( 20 );
        v1.push_back( 30 );
        v1.push_back( 40 );
        v1.push_back( 50 );

        v1.erase( v1.begin( ) );
        v1.erase( v1.begin( ) + 1, v1.begin( ) + 3 );
}
```

Аксессоры

- **iterator begin();**
- **iterator end( );**
- **iterator rbegin();**
- **iterator rend( );**
- **size_type size( ) const;**
- **size_type max_size( ) const;**
- **size_type capacity( ) const;**
- **bool empty( ) const;**
- **reference front( );**
- **reference back( );**
- **reference at(size_type** _Pos **);**
- **reference operator[](size_type** _Pos **);**

Assign

```
void assign( size_type _Count, const Type& _Val );

template<class InputIterator>
void assign( InputIterator _First, InputIterator _Last );
```

Очищает вектор и добавляет в него указанные элементы

```
int main( )
{
        vector<int> v1, v2, v3;
        v1.push_back(10);
        v1.push_back(20);
        v1.push_back(30);
        v1.push_back(40);
        v1.push_back(50);

        v2.assign(v1.begin(), v1.end());
        v3.assign(7, 4) ;
}
```

Swap

```
void swap ( vector<Type, Allocator>& _Right );

friend void swap (
   vector<Type, Allocator >& _Left,
   vector<Type, Allocator >& _Right
                        );
```

Обменивает значение двух векторов за O(1)

Практическое задание

Есть список студентов и их оценок по основным предметам. Необходимо

1. Выводить список в алфавитном порядке
2. Выводить средний бал по каждому
3. Искать всех студентов по введенному шаблону
4. Сортировать по оценке конкретного предмета и алфавитном порядке
5. Выводить всех у кого средний бал больше 4