Язык C++

Мещерин Илья

Лекция 13

8.7) Пример нестандартного аллокатора
StackAllocator<int>
Сначала выделяет большой кусок памяти. Особенность в том, что при deallocate ничего не делает (экономит время работы).

Контейнеры

vector, list, deque - Sequence containers (реализация идеи последовательного хранения элементов)
map, set (+multi, unordered) - Associative containers (реализация идеи построения отображения)

9.1) std::vector

```
1  template<class T, class Alloc = std::allocator<T>>
2  class vect{
3      T* arr;
4      size_t sz, cap;
5      Alloc alloc;
6      using traits = allocator_traits<Alloc>;
7  public:
8      vect(){
9          cap = 1;
10         sz = 0;
11         arr = (T*)(new char[sizeof(T)]);
12     }
13 };
```

a) *push_back(), pop_back(), emplace_back()* (создает объект непосредственно в конце вектора из аргументов)

```
1  void push_back(const T &x){
2      if (sz == cap){
3          T* newarr = traits::allocate(alloc, cap * 2);
4          for (int i = 0; i < cap; ++i){
5              traits::construct(alloc, newarr + i, arr[i]);
6          }
7          for (int i = 0; i < cap; ++i){
8              traits::destroy(alloc, arr + i);
9          }
10         traits::deallocate(alloc, arr, cap);
11         arr = newarr;
12         cap <<= 1;
13     }
14     //arr[sz] = x;
15     //new(arr + sz) T(x);
```

```
16      //alloc.construct(arr + sz, x);
17      traits::construct(alloc, arr + sz, x);
18      sz++;
19  }
20
21  template <class ... Args>
22  void emplace_back(const Args &... args) { //... }
```

б) *Operator[], at()* (если вышли за границы, то кинет исключение, при использовании *operator[]* будет ub)

```
1  T& operator[](size_t i) {
2      return arr[i];
3  }
4
5  const T& operator[](size_t i) const { return arr[i]; } //for const vector
```

в) Конструктор, деструктор, operator=
При копировании нужно определить аллокатор для нового контейнера. Функция ниже возвращает аллокатор для нового объекта.

```
traits::select_on_container_copy_construction(alloc);
```

г) *swap()*
Меняет местами указатели на массивы и меняет местами аллокаторы.

д) *size(), capacity(), front(), back(), resize(), shrink_to_fit()* (делает размер вектора равным числу элементов в нем)

Размер вектора константа

```
1  int main(){
2      vector<int> v(100);
3      sizeof(v);
4  }
```

Упадет из-за двойного удаления по одному месту.

```
1  int main(){
2      vector<int> v(100);
3      delete[] &(v[0]);
4  }
```