ООП
Лекция 2

14 сентября, 2012

Алексей Лебедев

Итераторы — это обобщенные указатели

```
for (int *i = m; i != m+n; ++i)
    cout << *i << ", ";

for (list<int>::iter i = l.begin(); i != l.end(); ++i)
    count << *i << ", ";
```

Итераторы — это интерфейс

```
template<class I, class F>
void for_each(I from, I to, F f)
{
    for (; from != to; ++from)
        f(*from);
}
```

Вопрос: что скрывается за I?

Что скрывается за итератором?

Контейнер?

Что скрывается за итератором?

Как на счет файла?

Что скрывается за итератором?

Правильный ответ: алгоритм.

(То есть все, что угодно.)

Пример: слияние

```
template<class I1, class I2>
class merge_iterator {
   I1 from1, to1;
   I2 from2, to2;
public:
   merge_iterator(...) { ... }
   typename I1::value_type operator*() const;
   merge_iterator<I1, I2> &operator++();
};
```

Диапазоны: вдвое меньше сущьностей

```
template<class R1, class R2>
class merge_iterator {
   R1 r1;
   R2 r2;
public:
   merge_iterator(R1 r1, R2 r2) { ... }
   typename R1::value_type front() const;
   void pop_front();
    void empty();
};
```

Итераторы или диапазоны?

```
for (vector<int>::iterator i = v.begin();
      i != v.end(); ++i)
   *i *= k;
```

или

```
for (vector<int>::range r = v.all();
     !r.empty(); r.pop_front())
```

Итераторы или диапазоны?

```
template<class T>
class vector {
...
public:
   iterator begin();
   iterator end();
   iterator rbegin();
   itorator rend();
```

или

```
   range all();
   range rall();
};
```