C++

Разбор домашней работы

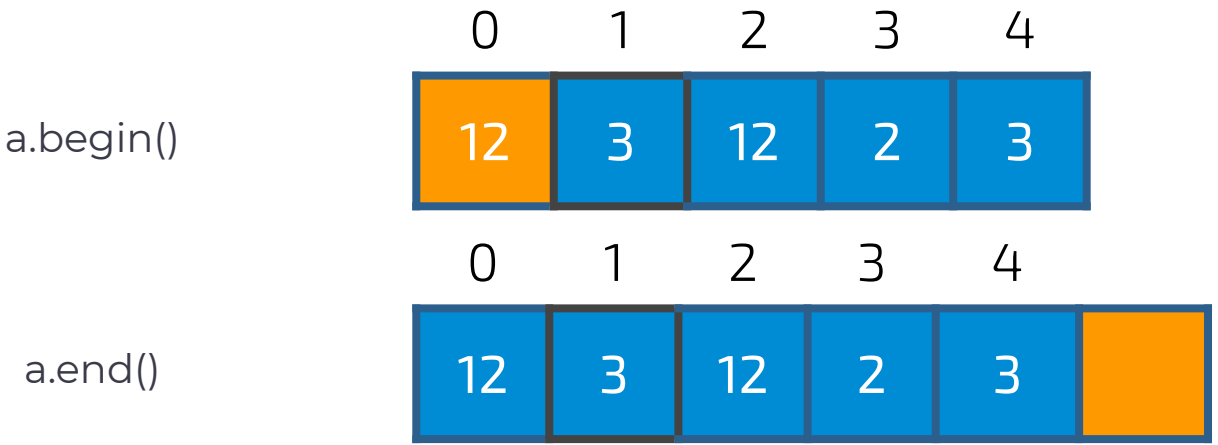
Алгоритмы

Теория

15 минут

Обозначения

Н - начальный итератор или указатель. Пример: a.begin() + 1
К - конечный итератор или указатель. Пример: a.end() - 2



for_each

for_each - алгоритм, проходящий по элементам контейнера от Н до К, применяя к элементам функцию.

```
for_each(н, к, функция)
```

Пример

```
1.  void show_val(int value) {
2.       cout << value << " ";
3.  }
4.  . . .
5.  . . .
6.  for_each(a.begin(), a.end(), show_val);
```

Что произойдет, если как параметр функции указать ссылку?

all_of

all_of - алгоритм, проверяющий, что все элементы удовлетворяют условию функции. Как результат возвращает *true* или *false*. Функция-параметр должна возвращать значение типа bool.

```
all_of(н, к, функция)
```

Пример

```
1.  bool check_positive(int value) {
2.      return (value > 0);
3.  }
4.  . . .
5.  if (all_of(a.begin(), a.end(), check_positive)) {
6.      cout << “YES”;
7.  } else {
8.      cout << “NO”;
9.  }
```

any_of, none_of

`any_of(н, к, функция)` - алгоритм, проверяющий, что хотя бы один элемент удовлетворяет условию функции. Как результат возвращает *true* или *false*. Функция-параметр должна возвращать значение типа bool.

`none_of(н, к, функция)` - алгоритм, проверяющий, что ни один элемент не удовлетворяет условию функции. Как результат возвращает *true* или *false*. Функция-параметр должна возвращать значение типа bool.

count_if

`count_if(н, к, функция)` - алгоритм, считающий, сколько элементов удовлетворяют условию функции. Функция-параметр должна возвращать значение типа bool.

Пример

```
1.  bool check_positive(int value) {
2.      return (value > 0);
3.  }
4.  . . .
5.  . . .
6.  cout << count_if(a.begin(), a,end(), check_positive);
```

find_if

`find_if(н, к, функция)` - алгоритм поиска элемента, удовлетворяющего условию функции. Функция-параметр должна возвращать значение типа bool.

Как результат возвращает указатель или итератор на первый найденный элемент.

Если искомых элементов нет, возвращает как результат K (итератор конца).

Как получить из итератора индекс элемента?

Практика

20 минут

Теория

15 минут

remove

`remove(н, к, значение)` - алгоритм, удаляющий из контейнера все элементы, равные указанному значению.

Может применяться как к вектору, так и к обычному массиву.

Можно ли изменить размер обычного массива?

remove

`remove` - не изменяет размер исходного контейнера, но сдвигает все оставленные элементы влево, оставляя в конце последовательности удаленные элементы.

Как результат эта функция возвращает итератор на конец новой последовательности.

Как удалить часть вектора, если мы знаем итераторы?

Удаление

```
1.  vector<int> v = {2, 2, 4, 4, 3};
2.  //Получаем итератор на конец оставшихся элементов
3.  auto it = remove(v.begin(), v.end(), 4);
4.  //Начиная с этого итератора и до конца удаляем всё
5.  v.erase(it, v.end());
```

Предположите, что делает функция unique

unique

`unique` - функция, удаляющая последовательно повторяющиеся элементы, оставляя только одно из этих значений.

Как и функция remove, не меняет размер исходного контейнера, но возвращает итератор на конец оставшихся элементов

unique

```
1.  vector<int> v = {1, 2, 2, 3, 3, 2};
2.  //Получаем итератор на конец оставшихся элементов
3.  auto it = unique(v.begin(), v.end()); //1 2 3 2 x x
4.  //Начиная с этого итератора и до конца удаляем всё
5.  v.erase(it, v.end()); //1 2 3 2
```

Практика

20 минут

Итоги урока

1) Функции можно передавать как параметры других функций
2) Новые функции из библиотеки algorithm