

# NVBL, AS

## Boost. Compute

### Семинар 3

#### Контейнеры (часть 2)

`std::set (multiset)`

`std::set < T`, простертое сравнение, опонатор?

сторона СЛАБОЕ

УПОРЯДОЧЕННЫЕ

Свойства:

1) Аntисимметричность  $x < y \in T \Rightarrow y > x \in R$

2) Транзитивность

3) Упротивенность  $x < x \in R$

4) Транзитивность субординатности

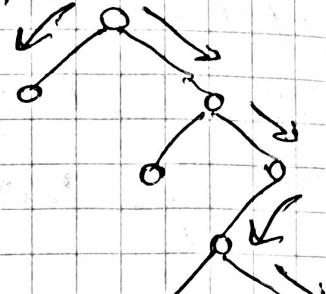
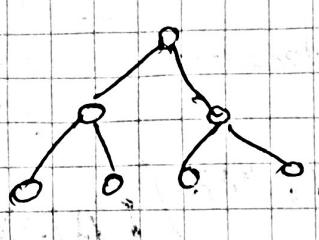
$x \geq y ; y \geq z ; \Rightarrow x \geq z$ .

$x \geq y \wedge z$

$!(x < y) \wedge !(y < x)$

`std::set` - RB-tree - сбалансированное

бинарное дерево поиска.



сбалансированное несбалансированное

Если red / black

root 1 leafs - black

Быть может красного - чёрные.

Критерий сбалансированности

Во всех праях путях от root до leafs

одинаковое количество чёрных узлов.

Если это не так, то несбалансированное дерево.

std::set (multiset)

→ можно хранить данные в отсортированном порядке.

+ симметричное представление.

lower\_bound lee менюне  
 upper\_bound 1. [2, 2, 3]  
 equal\_range дөлвие  $\rightarrow$  std::pair(2, 3)  
 binary-search  $\rightarrow$  TRUE / FALSE

Возбр. оператор, который не менюне  
 (ониже) сеесеелено.

#include <set>

6 main () :

std::set<int> set;

set.insert(1); // ее каго

set.insert(2); // указатель

на элемент

берабар

(как в бирнеле)

No elements

set.insert(std::begin(set), 1); // hint

зато

std::pair<iterator>

некоторые

Возбрауяралык оператор на int.

$\rightarrow$  указатель.

Задачи: insert, erase, swap & бал.

11

std::map (multimap)  
( $\hookrightarrow$  std::set < std::pair<key, Value> >)

#include <map>

int main (int argc, char\*\* argv)

{  
 std::map < std::string, int > students;

students["Jens"] = 10;

students["Jens"] = 11;

students["Dinara"] = 10;

students.insert (std::make\_pair(" - ", 12))

students.erase ("Jens");

↑ year remove

auto first = students.find ("Pinara")

auto second = students.find (" - ")

Изменяя КЕ МОРГТ ячейку побольше,  
исчеза МОРГТ

std::swap (a->second, b->first);

```
for (auto e : students)
{
    std::cout << e.first << std::endl;
    PACHEYATKA
```

### Динамические

```
std::vector<int> v;
std::multimap<int> m;
std::multiset<int> s(v.begin(),  
v.end());
```

std::map (ордерение присвоюется  
отсортированный порядок ← присвоение)

Сортируем по ключу  
map[5] → значение и 5 (ключ)

Несортируемые коллекции

### ХЭШ-ТАБЛИЦЫ

```
std::unordered_set<int> (multi)
- map(multi)
```

Object  
на баг

как хм-структура

→

hash  
-bag

[0; INT\_MAX]

%h

Остаток от деления

(Равномерное конвертация  
значений в единиц  
изображение)

остаток от

n-1

деление

01
02
03
04

Хм-Код деление  
на h

h

ХМ-ТАБЛИЦА

(Контузье)

Преобразование в хм-структуру.

1) детерминированность

2) скорость выполнение  $\propto$  одна

(не зависит от n, где n - количество  
элементов в хм-таблице)

Зависит от! сортировки (от его длины)

Сортировка  $\Rightarrow$  ASCII код  $\Rightarrow$  сортировка.

числа с плавающей точкой  
приручают ошибки?

Но ведь не хочется сравнивать  
числа с double, потому что точности  
не хватает.  $\rightarrow$  Не использовать hash map  
double/float

Например:  $\epsilon = 0,005$

$0,002 \uparrow$   
 $0,001 \downarrow$

3.) Равномерность. — равномерное  
распределение гаджетов при равномерном  
погоне бояра.

(+) поиск, удаление, вставка  $\rightarrow O(1)$

В другом случае.  $\leftarrow$  Сложность  
операции.

Если соединяют хэш-коды файлов  
объектов в таблице — конфликт  
(много, если есть много)  $\rightarrow$  (много фальшивых,  
занесенных в)

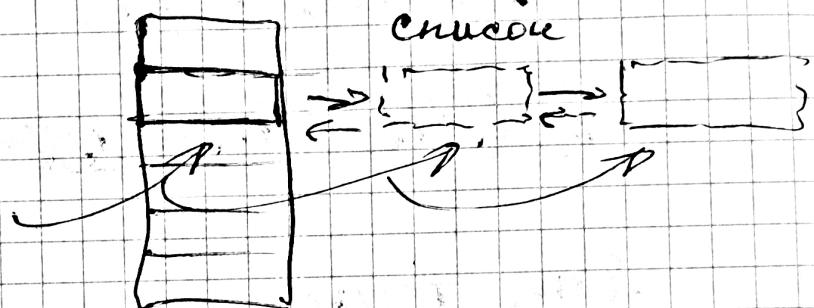
Это быстрое хранение, что делает

такое хранилище используемо.

быстро:

1) Добавка  $\rightarrow$  список, cost  $O(1)$  мес. времени.

Много уровней (ярусов, ячеек, строк)  
(list, forward-list)



2) Удалять можно не только в соседство.

Много отработки  
израсходовано.

Сложность  $O(1) \mapsto O(n)$



Время поиск константе

$$P_{\text{искн.}} \approx \frac{N}{L} = L$$

$n$ -байт (128)

$L_{\text{поск}} = 1.0 > L$  ( $L_{\text{поск}} - \max \text{ load factor}$ )

Аналогично с памятью

DR

коэффициент

чтобы  $\Delta \rightarrow \Delta_{\text{натур.}} \Rightarrow$  убирается  $\Delta$

$$\Delta = \Delta h$$

перемножение.

,  $O(N) \rightarrow \%h + \text{непос.}$

Было правило  $\Delta \approx 0,7$

135 кр. •  $0,7 \approx 100$  кр.

## БЫСТРЫЕ СПОСОБЫ ХОДИ-ГАБЛ

Дан. Таблица

1

хранимые

B	C	A	D	E	F
---	---	---	---	---	---

1) std::set

2). multiset (хорош)

3) vector

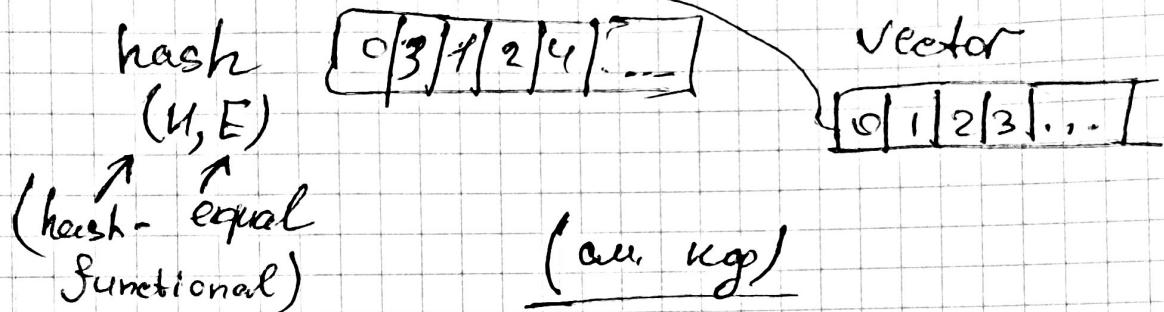
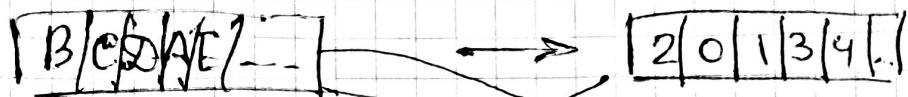
одномерные

[boost::multi\_index]

set - хранят в твои непара, в которых  
контейнер имеет единообразием.

Есть алгоритм сортировки, и с помощью  
девято биндер set.

set  $\rightleftharpoons$  lower\_bound  
upper\_bound



## Boost: bimap

std::map  $\leftrightarrow$  boost::bimap

$C_n \cdot \tau_0 \log_2 n$

Симметрия в языке наращивания

(Интервалы в виде left/right between  
first/second)