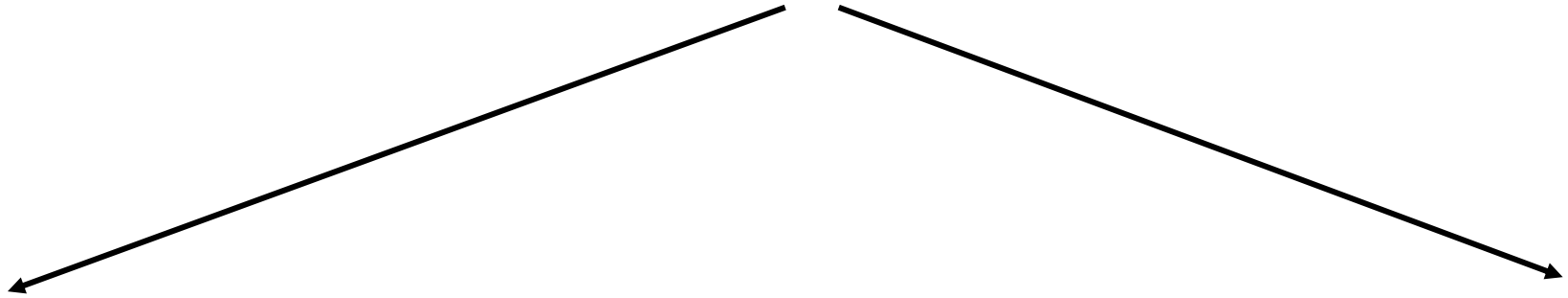
Ассоциативные контейнеры STL

Контейнеры STL



Последовательные

array
vector
deque
forward_list
list

Ассоциативные

set
map
multiset
multimap
unordered_map
unordered_set

Ассоциативный массив

Ассоциативный массив — абстрактный тип данных, позволяющий хранить пары вида «(ключ, значение)» и поддерживающий операции **добавления** пары, а также **поиска** и **удаления** пары по ключу.

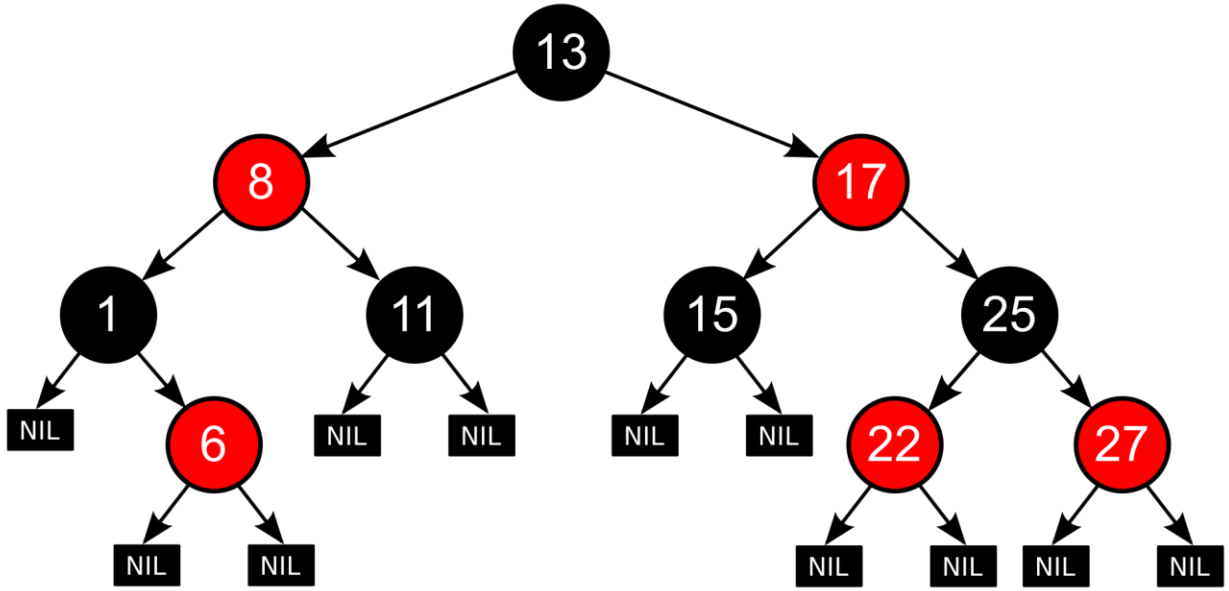
Std::map

```
#include <iostream>
#include <map>
#include <string>

int main()
{
    std::map<std::string, int> m;
    m["Math"] = 5;
    m["Literature"] = 4;
    m["Russian"] = 4;
    print_map(m);

}
```

Отсортированный ассоциативный контейнер, позволяющий хранить пары [ключ — значение]. Значение ключа – уникально.

Полный код см. в репозитории

Красно-черное дерево: реализация



1. Узел может быть либо красным, либо чёрным и имеет двух потомков;
2. Корень — как правило чёрный. Это правило слабо влияет на работоспособность модели, так как цвет корня всегда можно изменить с красного на чёрный;
3. Все листья, не содержащие данных — чёрные.
4. Оба потомка каждого красного узла — чёрные.
5. Любой простой путь от узла-предка до листового узла-потомка содержит одинаковое число чёрных узлов.

Красно-чёрное дерево — двоичное дерево поиска, в котором каждый узел имеет атрибут *цвета*.

Полный код см. в репозитории

Std::unordered_map

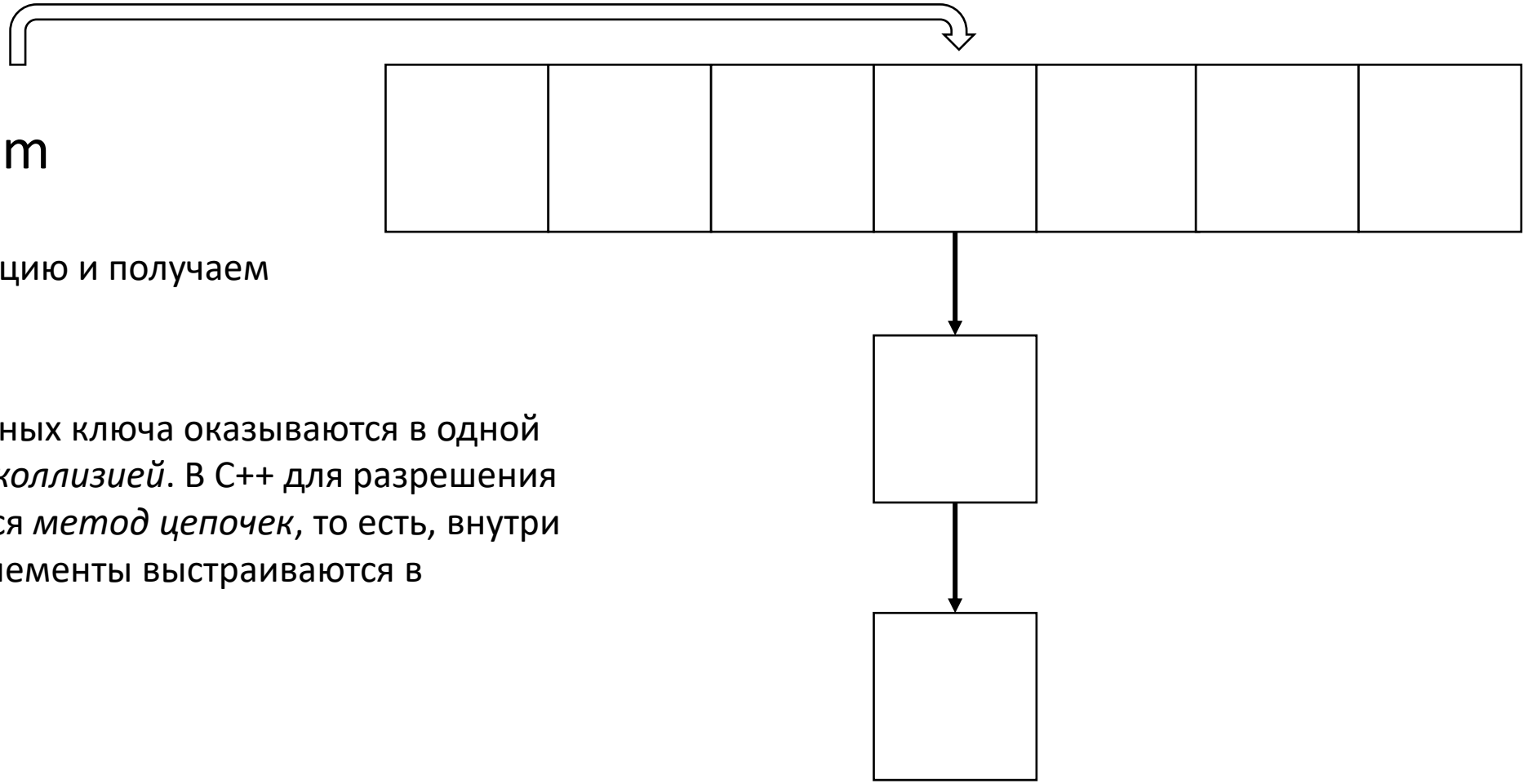
```cpp
#include <iostream>
#include <unordered_map>
#include <string>

int main()
{
    std::unordered_map<std::string, int> m;
    m["Math"] = 5;
    m["Literature"] = 4;
    m["Russian"] = 4;
    print_map(m);

}
```

Ассоциативный контейнер, позволяющий хранить пары [ключ — значение]. Представлен в виде хэш-таблицы

Полный код см. в репозитории

Хэш-таблица: реализация

Hash: key -> num



Применяем хэш функцию и получаем уникальное число.

Случай, когда два разных ключа оказываются в одной корзине, называется *коллизией*. В C++ для разрешения коллизий используется *метод цепочек*, то есть, внутри одной корзины все элементы выстраиваются в односвязный список.

Полный код см. в репозитории

Std::multimap

```
#include <iostream>
#include <map>
#include <string>

int main()
{
    std::multimap<std::string, int> m;
    m.insert(std::pair<std::string, int>("Math",5));
    m.insert(std::pair<std::string, int>("Math", 4));
    m.insert(std::pair<std::string, int>("Russian", 5));
    print_map(m);
    m.erase("Math");
    print_map(m);
}
```

Отсортированный ассоциативный контейнер, позволяющий хранить пары [ключ — значение]. Значение ключа – может быть не уникальным.

Полный код см. в репозитории

Std::set

```
#include <iostream>
#include <set>

int main()
{
    std::set<int> s;
    s.insert(5);
    s.insert(34);
    s.insert(321);
    s.insert(5);
    print_set(s);
}
```

set — это контейнер, который автоматически сортирует добавляемые элементы в порядке возрастания. Но при добавлении одинаковых значений, set будет хранить только один его экземпляр. По другому его еще называют множеством.

Полный код см. в репозитории

Std::multiset

```
#include <iostream>
#include <map>

int main()
{
    std::multiset<int> s;
    s.insert(5);
    s.insert(34);
    s.insert(321);
    s.insert(5);
    print_set(s);
}
```

multiset — это контейнер, который будет содержать элементы в отсортированном порядке при добавлении, но он хранит повторяющееся элементы, по сравнению с множеством set.

Полный код см. в репозитории

Зачем все это нужно?

Container	Insert Head	Insert Tail	**Insert**	Remove Head	Remove Tail	**Remove**	Index Search	Find
vector	n/a	O(1)	O(n)	O(1)	O(1)	O(n)	O(1)	O(log n)
list	O(1)	O(1)	O(1)	O(1)	O(1)	O(1)	n/a	O(n)
deque	O(1)	O(1)	n/a	O(1)	O(1)	O(n)	n/a	n/a
queue	n/a	O(1)	n/a	O(1)	n/a	n/a	O(1)	O(log n)
stack	O(1)	n/a	n/a	O(1)	n/a	n/a	n/a	n/a
map	n/a	n/a	O(log n)	n/a	n/a	O(log n)	O(1)	O(log n)
multimap	n/a	n/a	O(log n)	n/a	n/a	O(log n)	O(1)*	O(log n)
set	n/a	n/a	O(log n)	n/a	n/a	O(log n)	O(1)	O(log n)
multiset	n/a	n/a	O(log n)	n/a	n/a	O(log n)	O(1)*	O(log n)

Сложность алгоритмов

O(1) – для операции требуется константное время
O(n) – полный перебор
O(log(n)) – бинарный поиск
O(n^2) – сортировка пузырьком (двойной полный перебор)