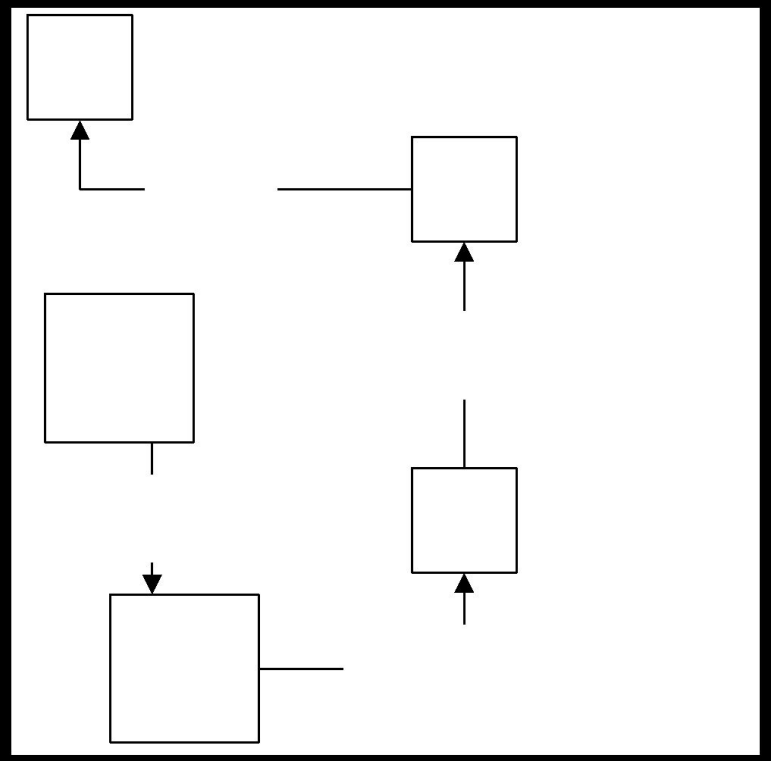
СТЕКИ

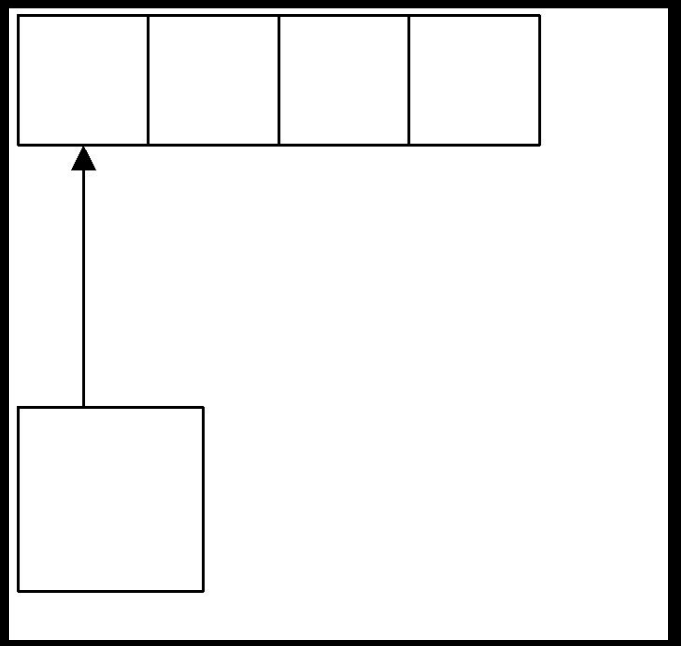
СТЕКИ

Стек – это структура данных, организованная по принципу LIFO – последний вошел, первый вышел.

СТЕКИ



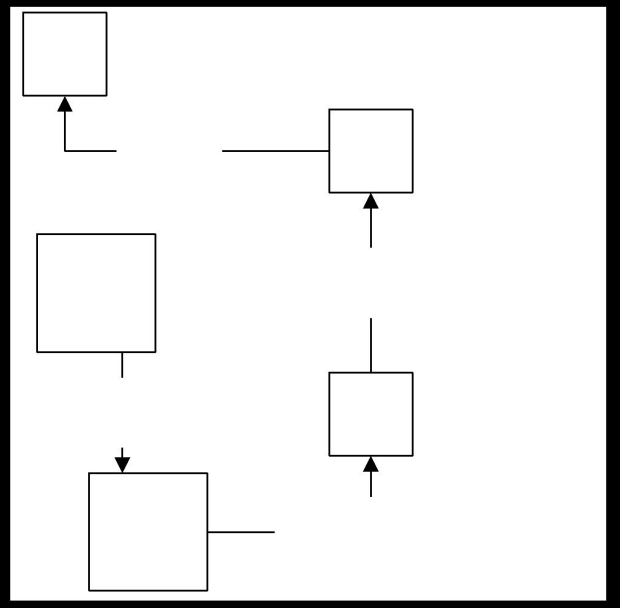
Для стека не важно фактическое местоположение элементов в памяти. В этом его отличие от массива, хранящего элементы последовательно.

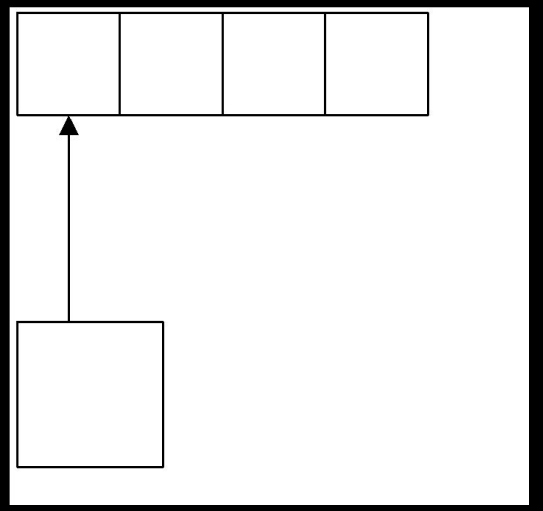


СТЕКИ

Массив хранит элементы последовательно, поэтому получение любого из них занимает O(1).

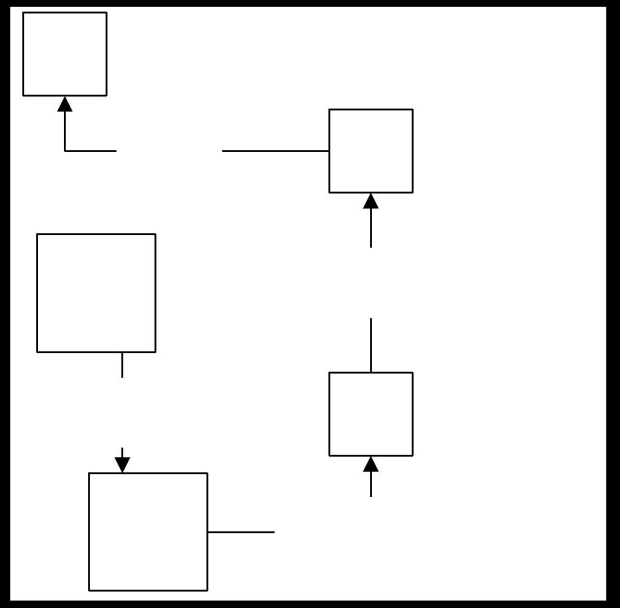
Каждый элемент стека хранит данные только о себе и положении следующего за ним элементе, поэтому время получения каждого из них - O(n).

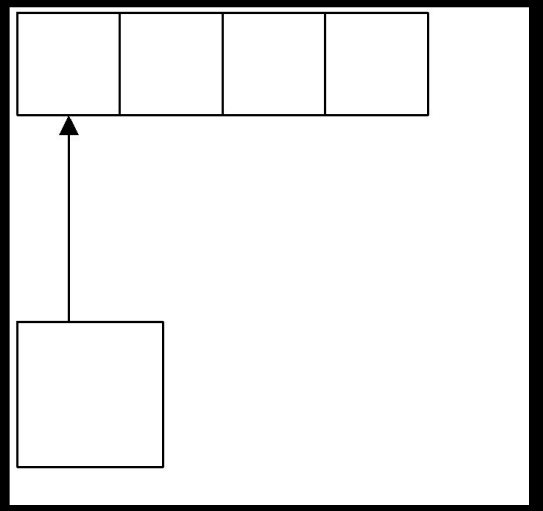




СТЕКИ

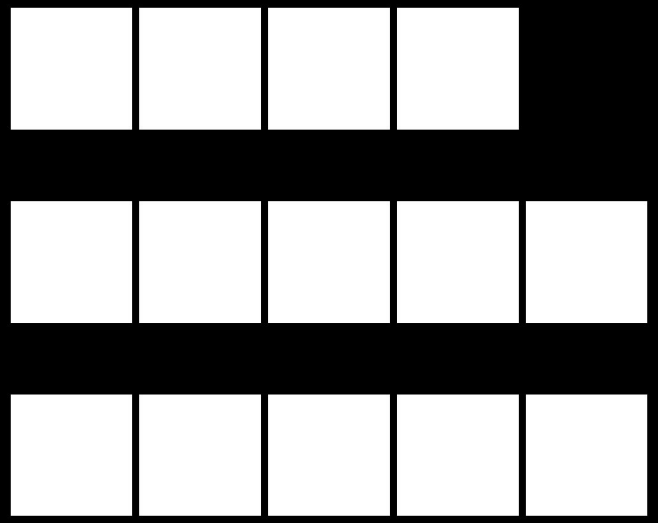
Поиск элемента и в массиве, и в стеке в худшем случае сводится к полному перебору элементов – время поиска будет O(n).



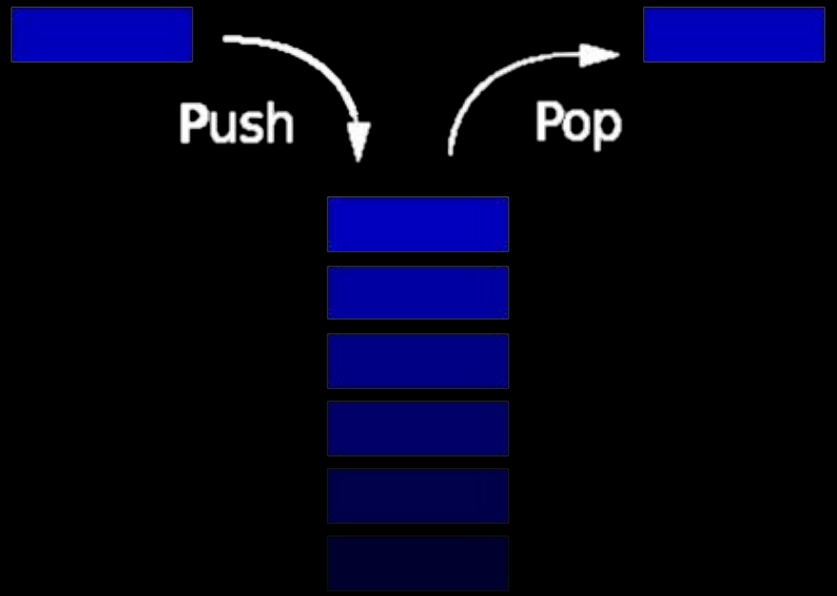


СТЕКИ

Вставка или удаление элемента в массиве требует времени O(n)

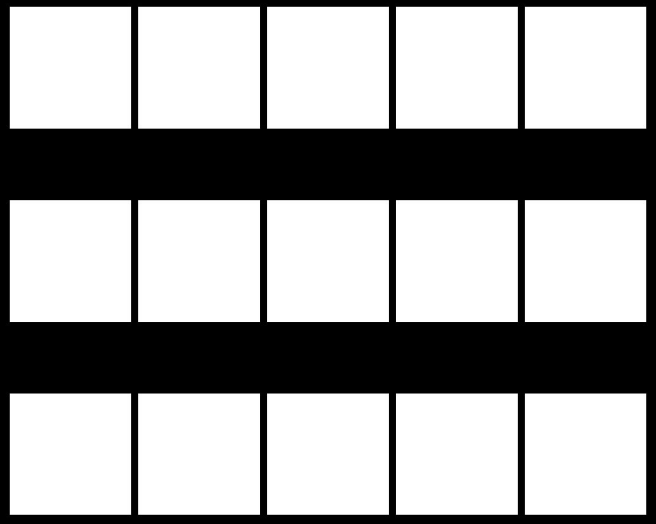


В стеке данные добавление и удаление данных происходит только в конце и занимает время O(1).



СТЕКИ

При удалении элемента из массива память, занимаемая удаленным элементом, не освобождается.



При удалении элемента из стека, занимаемая им память уменьшится.

СТЕКИ

Вставка элемента в массив может потребовать дополнительной памяти до размера ещё одного массива.

Вставка элемента в стек требует дополнительной памяти лишь для нового элемента.

СТЕКИ

Операции над стеком:

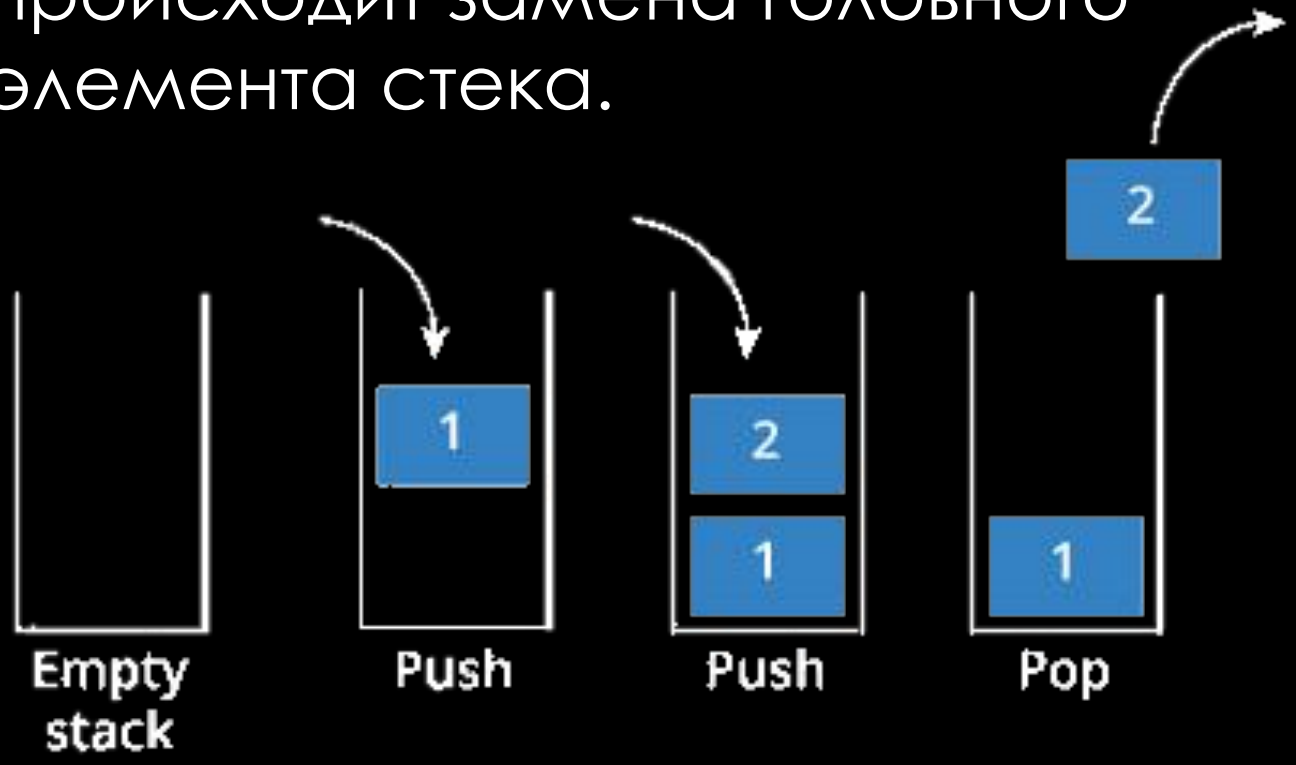
Добавление нового элемента – push

Удаление последнего элемента – pop

Чтение вершины – top

PUSH/POP

В операциях вставки/удаления происходит замена головного элемента стека.

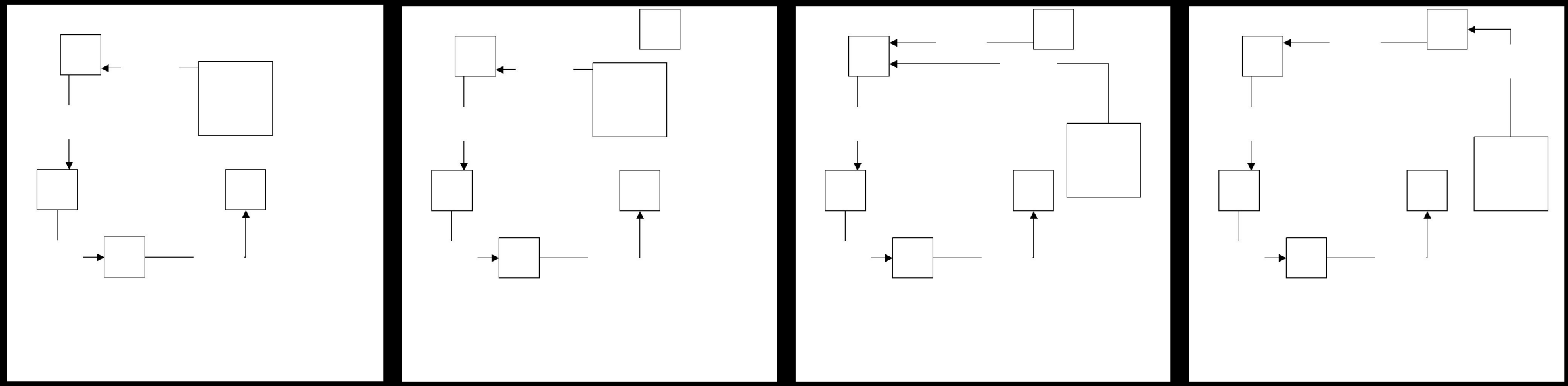


При добавлении, новый элемент становится головным.
При удалении из стека удаляется ссылка на последний элемент, а предпоследний элемент становится головным.

PUSH/POP

Push

Pop



ПРОБЛЕМЫ

- Возможно, что элементы, добавленные в стек в начале, так и не будут прочитаны.
- Данные не упорядочены.
- Получение произвольного элемента может быть длительным.

STL:STACK

```
#include <iostream>
#include <stack>
 using namespace std;
int main()
{
   stack <int> st;
   return 0;
}
```

STL:STACK

- Операции:
- pop – удаление элемента
- empty – проверка на пустоту
- swap– обмен содержимого с другим стеком
- size – количество элементов в стеке
- push – добавление элемента в стек
- emplace – создание и добавление элемента в стек

STL:STACK - EMPLACE

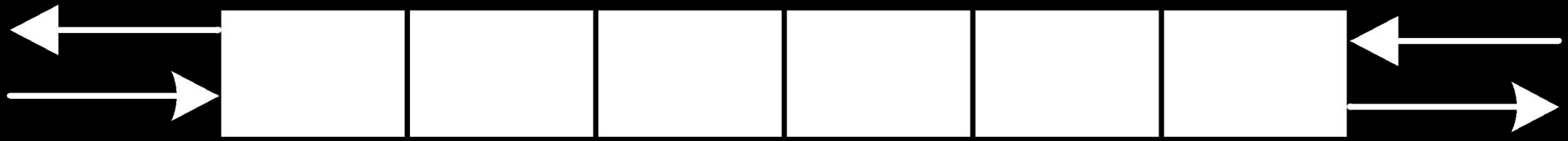
```
#include <iostream>
#include <stack>
using namespace std;
struct some{
int a, b, c;
some(int a, int b, int c){
this->a=a;
this->b=b;
this->c=c;}};
```

```
int main()
{    stack <some> st;
     st.push(f);
     //st.push(1,2,3); - не заработает
     st.emplace(1,2,3);// - создаст структуру с полями 1,2,3 и добавит её в стек
}
```

ДЕКИ

Деки располагаются в памяти так же, как и стеки.

В отличии от стека, в деке возможно добавление и в начало, и конец.



ДЕКИ

- Вставка и удаление данных производится за O(1).
- Получение, вставка и удаление произвольного элемента производится за O(n).
- Поиск элементов происходит за O(n).

ДЕКИ

- Деки требуют меньше памяти для вставки, чем массивы.
- Это связано с тем, что элементы в деке расположены не в одной области памяти.

ДЕКИ

- Операции над деком:
- Вставка в начало и конец дека – push_front, push_back
- Удаление из начала и конца – pop_front, pop_back
- Чтение первого и последнего элемента – front, back
- Вставка и удаление элемента в произвльном месте – insert, erase
- Число элементов и максимальное число элементов – size, max_size
- Изменение размера – resize
- Высвобождение памяти от неиспользуемых элементов – shrink_to_fit
- Семейство функций emplace
- Обмен содержимого деков - swap