



Объектно-ориентированное программирование

Петрусович Денис Андреевич

Доцент Кафедры Высшей математики

petrusevich@mirea.ru



Тема 0. Знакомство со структурами данных

Последовательные контейнеры
(сохранен порядок, в котором
добавлены элементы)

Ассоциативные контейнеры
(элементы упорядочены, пары
«ключ — значение»)



Тема 0. Работа с массивом

- Массив – непрерывный участок памяти, состоящий из одинаковых по структуре элементов
- Есть эффективная индексация $[i]$



«Стоимость» операций

Производительность операций определяется по порядку величин, зависящих от длины массива (чаще n)

Выделяют оценки $\Theta()$, $O()$, $\Omega()$

Оценка операции в среднем: $O()$

$f(n)$ – оценка количества действий

$O(n)$: $\lim f(n) / n = \text{const}$

$O(1)$: $f(n) \leq A, A = \text{const}$



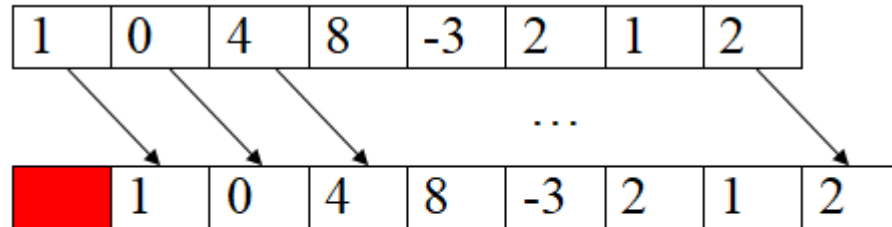
Операции над массивом

- Добавление в конец: $O(1)$

1	0	4	8	-3	2	1	
---	---	---	---	----	---	---	--

1	0	4	8	-3	2	1	2
---	---	---	---	----	---	---	---

- Добавление в начало: $O(n)$



3	1	0	4	8	-3	2	1	2
---	---	---	---	---	----	---	---	---

- Обращение к i -му элементу
(индексация $[i]$)




Операции над массивом

- Обращение к i -му элементу (индексация $[i]$)

$$\&\text{arr}[i] = \&\text{arr}[0] + i * \text{sizeof}(\text{arr}[0])$$

Не требуется перечислять
элементы в массиве

3	1	0	4	8	-3	2	1	2
---	---	---	---	---	----	---	---	---





Работа с массивом

Поиск в неотсортированном массиве —
линейный поиск, $O(n)$

Поиск элемента *elem_to_find* в массиве *arr*
длиной *len*

```
for (i=0; i<len; i++)  
    if (arr[i]==elem_to_find)  
        ...
```



Работа с массивом

Поиск в отсортированном массиве – бинарный поиск. Поиск элемента «3»

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----

1:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----



2:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----



3:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----



Размер рассматриваемой части массива
уменьшается в два раза на каждой итерации



Работа с массивом

Бинарный поиск. Поиск элемента «12»

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----

1:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----

2:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----

3:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----

4:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----



Работа с массивом

Бинарный поиск. Оценка максимального
числа шагов

Размер рассматриваемой части уменьшается в
2 раза на каждой итерации, пока не
останется 1 элемент: $n / 2$, $n / 4$, $n / 8$, $n / 16$,
... $n / 2^k$

$\max k = ?$

$n / 2^k \sim 1$

$k \sim \log n$



Работа с массивом

Индексация [i] $O(1)$

Вставка/удаление $O(n)$

Добавление в конец $O(1)$

Поиск в

отсортированном массиве $O(\log n)$

Поиск в

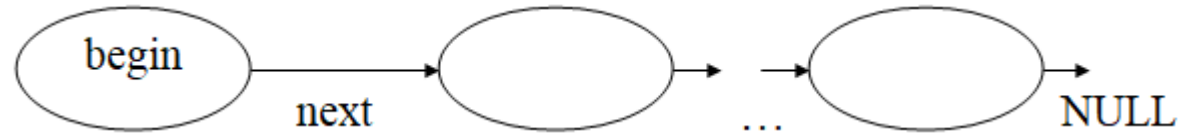
неотсортированном массиве $O(n)$

n — число элементов в массиве



СВЯЗНЫЕ СПИСКИ

Участок памяти не обязан быть непрерывным



Односвязный список: в каждом элементе есть указатель на следующий next

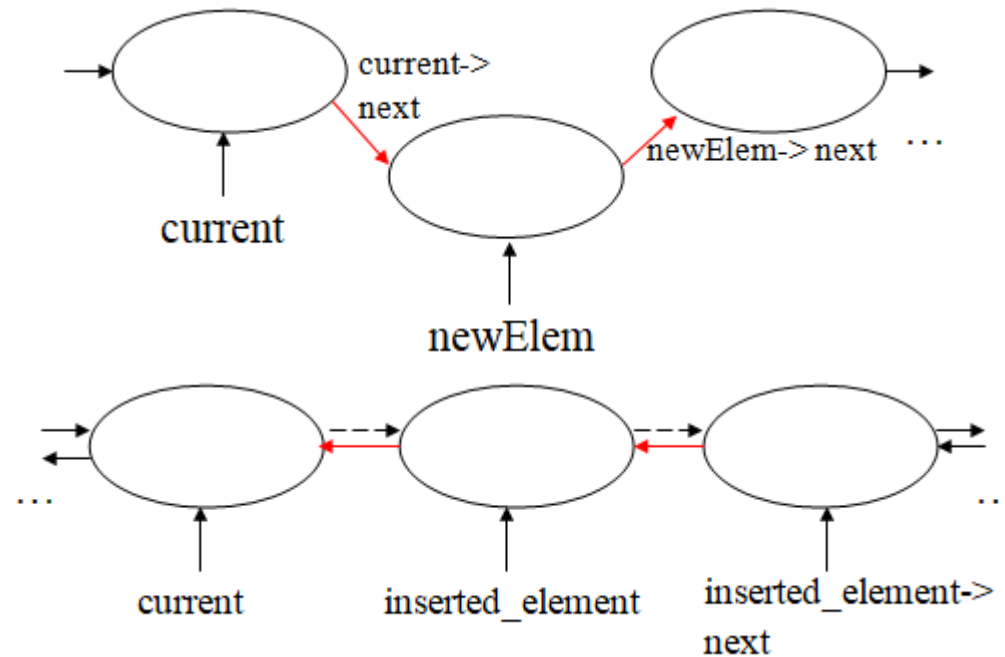
Двусвязный список: вдобавок, есть указатель на предыдущий элемент previous





СВЯЗНЫЕ СПИСКИ

В такой структуре удобно вставлять/удалять элементы:



Вставка/удаление $O(1)$

Индексация $O(n)$

Линейный поиск $O(n)$

Бинарный поиск хуже линейного



СВЯЗНЫЕ СПИСКИ

Примеры применения:

1. Структура файловой системы FAT
2. Список объектов с одинаковым значением хэш-функций в хэш-таблице
3. Список объектов, в который часто надо добавлять/удалять элементы



Стек/очередь

Стек: LIFO (Last In First Out: добавление в конец /
извлечение с конца, напоминает стопку бумаги)

Очередь: FIFO (First In First Out: добавление в конец
/ извлечение с начала)



Литература. Связные списки

- Шилдт. Самоучитель C++ (о наследовании)
- Топп, Форд. Структуры данных (о связных списках). Глава 9.
- Вирт. Структуры данных (о связных списках). Пункт 4.3.
- Кормен. Алгоритмы: построение и анализ (о связных списках). Пункт 10.2.
- Weiss. Пункт 3.2
- Das. Глава 5
- Лекции Data Structures. Unit 1
- На openedu.ru доступен курс ИТМО «Алгоритмы и структуры данных». Неделя 4. Элементарные структуры данных.
<https://openedu.ru/course/ITMOUniversity/PADS/>



Спасибо за внимание!

Петрусович Денис Андреевич

Доцент Кафедры Высшей математики

petrusevich@mirea.ru