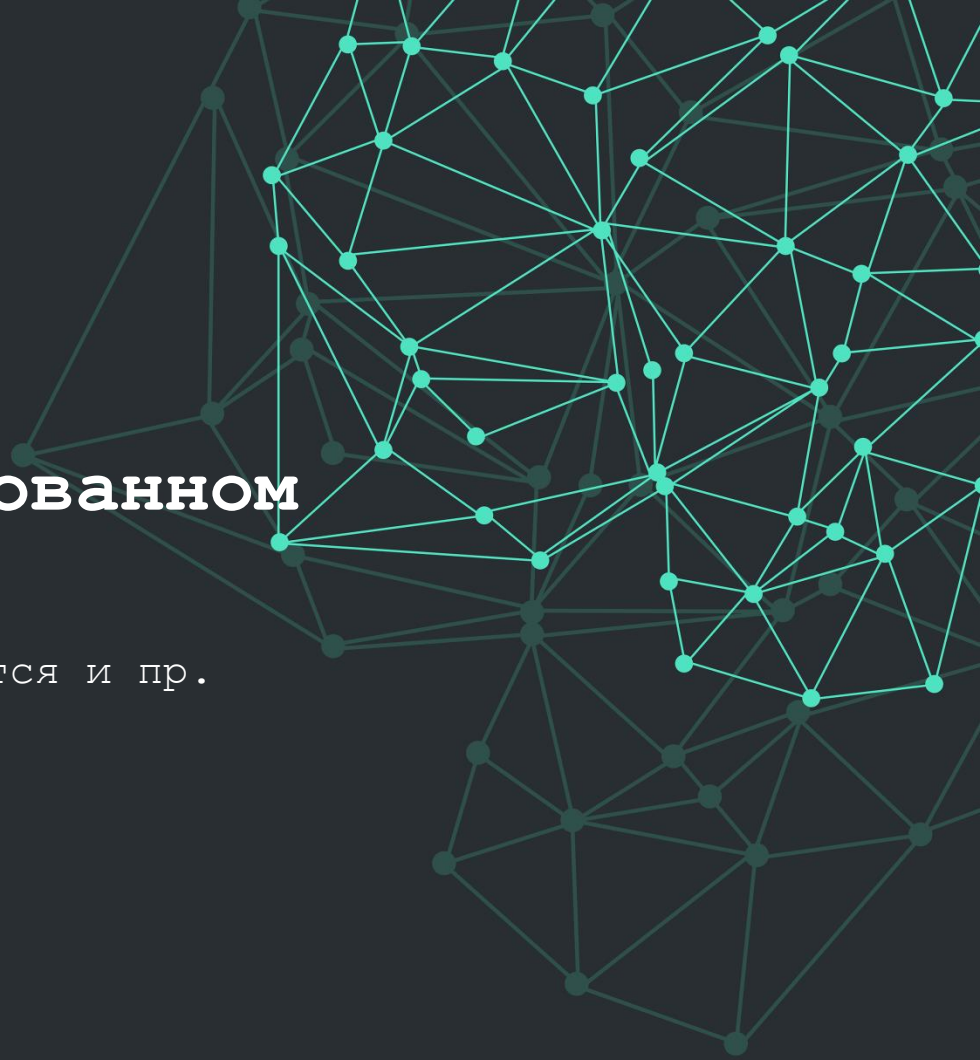
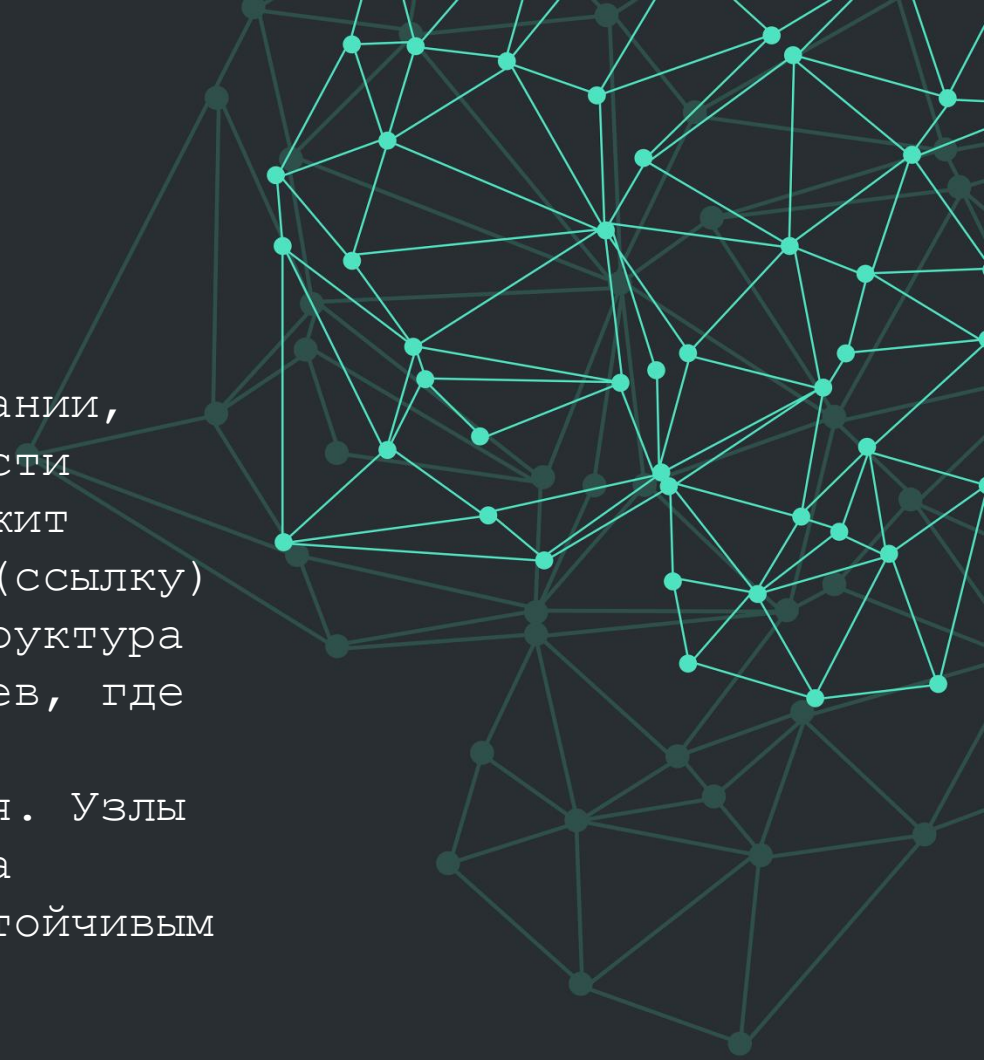
Односвязные списки в объектно-ориентированном программировании

Как появились, для чего используются и пр.



Введение

Односвязный список — это фундаментальная динамическая структура данных в объектно-ориентированном программировании, состоящая из последовательности узлов, где каждый узел содержит значение данных и указатель (ссылку) только на следующий узел. Структура отлично подходит для сценариев, где количество элементов заранее неизвестно или часто меняется. Узлы не требуют непрерывного блока памяти, что делает список устойчивым к фрагментации.



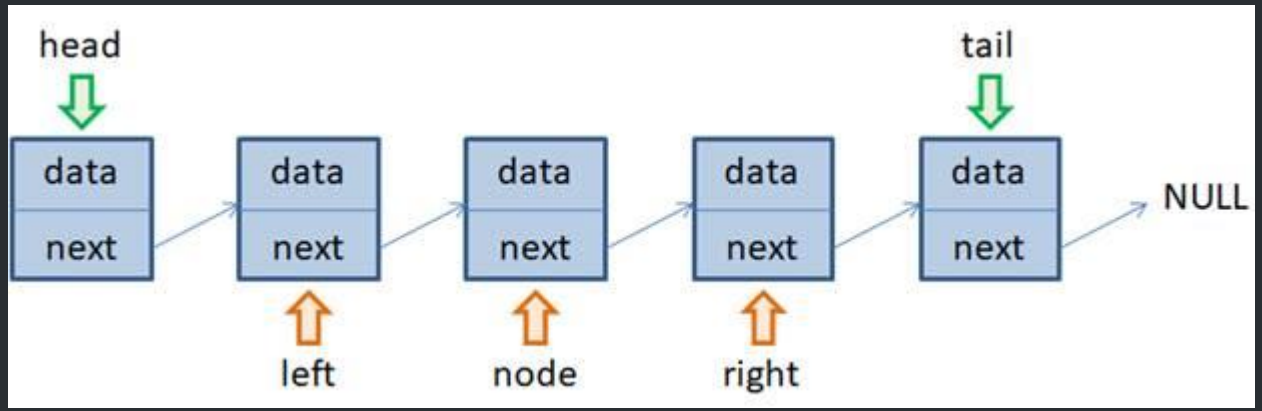
Краткая история

Односвязные списки возникли в 1950-1960-х годах с развитием высокоуровневых языков программирования, таких как Lisp, где они стали основой для рекурсивных структур и обработки символических выражений.

Изначально списки использовались для символьных манипуляций и графов, а их популярность выросла с появлением структур данных в 1970-х. В ООП они интегрировались в Java и C++.

Структура

Каждый узел односвязного списка — это объект с двумя полями: данными (data любого типа) и указателем next на следующий узел, что обеспечивает линейную связь без доступа к предыдущему. Голова (head) — входная точка для операций, хвост (tail) упрощает добавление в конец, ссылаясь на null для завершения.



Реализация

Вот так выглядит реализация узла:

```
struct Node {
   string val;
   Node* next;
   Node(string _val) : val(_val), next(nullptr){}
};
```

В узле есть:

- Значение, которое будет задавать пользователь
- Указатель на следующий элемент (по умолчанию nullptr)
- Конструктор

А это реализация односвязного списка:

```
struct list {
    Node* first;
    Node* last;

    list() : first(nullptr), last(nullptr) {}
};
```

В списке есть:

- Указатель на первый узел
- Указатель на последний узел
- Конструктор

В чём его отличие от массива?

В отличие от классического массива, где данные в памяти расположены строго последовательно, в односвязном списке, наоборот, данные расположены хаотично и связывание узлов списка происходит посредством ссылок. За счёт этой особенности в односвязный список можно добавлять произвольное число элементов, однако доступ будет осуществляться только последовательно. Произвольного доступа к элементам в односвязном списке нет.

Таблица сравнения с другими данными

Характеристика	Односвязный список	Массив	Двусвязный список
Доступ по индексу	O(n) последовательный	O(1) прямой	O(n)
Вставка начало/конец	O(1) с tail	O(n) сдвиг	O(1) bidirectional
Удаление	O(n) поиск	O(n) сдвиг	O(1) с prev/next
Память на элемент	data + 1 указатель	data	data + 2 указателя
Кэш-дружественность	Низкая (разрозненно)	Высокая	Средняя
Итерация	Только вперед	Полная	В обе стороны

Преимущества

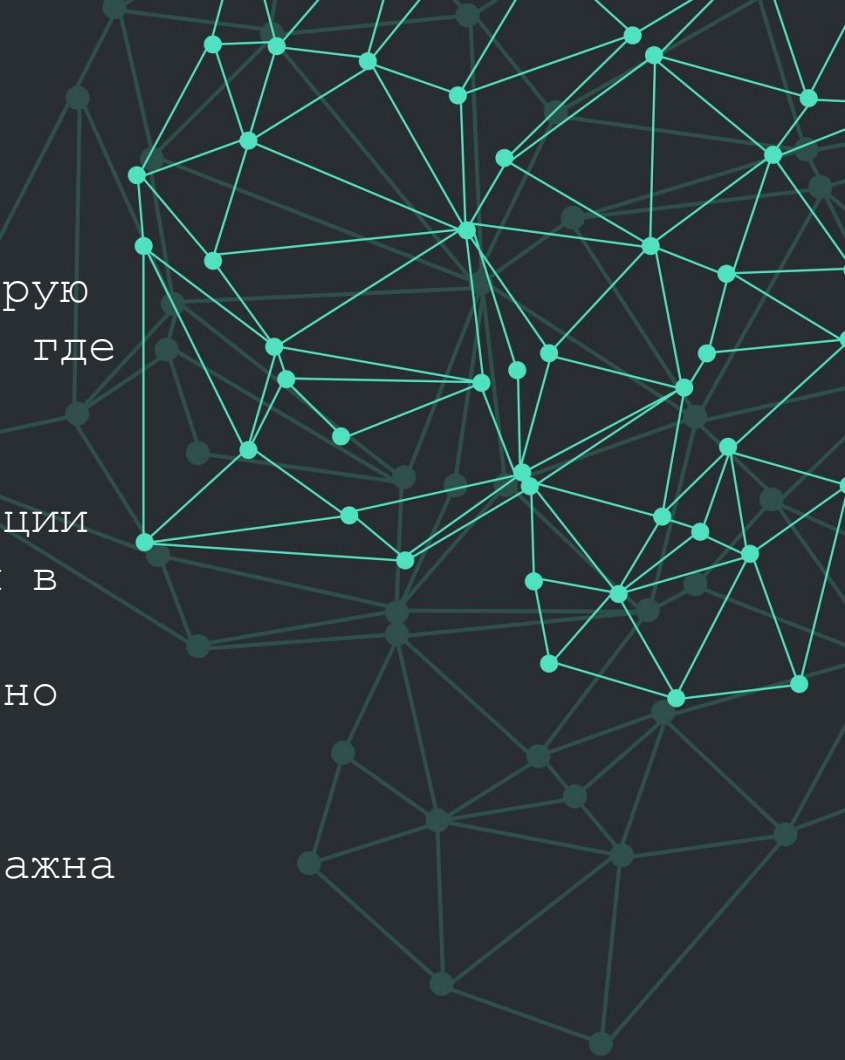
- Позволяют легко добавлять и удалять элементы в любое время, без необходимости передвигать другие данные, что удобно при динамическом изменении количества элементов.
- Не требуют выделения большого непрерывного блока памяти, можно хранить данные в разных местах, что помогает эффективно использовать память.
- Просты в реализации и хорошо подходят для таких структур, как стеки и очереди.

Недостатки

- Нельзя быстро получить доступ к произвольному элементу списка — нужно проходить все элементы последовательно, начиная с головы.
- Занимают дополнительную память под указатели на следующий элемент.
- Менее эффективны при задачах, где важен быстрый произвольный доступ к элементам, например, по индексу.

Вывод

Односвязный список — ключевая динамическая структура в ООП, которую лучше всего применять в ситуациях, где нужно часто добавлять или удалять элементы, а размер данных заранее неизвестен — например, при реализации стеков, очередей, истории действий в приложениях или списков задач в многозадачных системах. Они особенно удобны для потоковой обработки информации, парсинга текстов или моделирования путей в играх, где важна гибкость и простота изменений без перемещения данных.



Источники

https://education.yandex.ru/handbook/algorithms/article/odnosvyaznyj-spisok

https://habr.com/ru/articles/717572/

https://habr.com/ru/sandbox/153128/

https://foxminded.ua/ru/svyaznii-spisok-python/