Структуры данных

Структура данных — программная единица, позволяющая хранить и обрабатывать множество однотипных и/или логически связанных данных в вычислительной технике, формируются с помощью типов данных, ссылок и операций над ними в выбранном языке программирования.

Многие классические структуры данных представлены в стандартных библиотеках языков программирования или непосредственно встроены в языки программирования.

Структуры делятся на

- *Линейные*, элементы образуют последовательность или линейный список, обход узлов линеен. Примеры: Массивы. Связанный список, стеки и очереди.

- *Нелинейные*, если обход узлов нелинейный, а данные не последовательны. Пример: граф и деревья.

Основные структуры данных.

- Массивы
- Стеки
- Очереди
- Связанные списки
- Графы
- Деревья
- Префиксные деревья
- Хэш таблицы

Начнём с самого простого примера.

Массивы

309	18	24	65	43	1201

Пример массива, состоящего из 6 элементов

Каждому элементу данных присваивается положительное числовое значение (индекс), который соответствует позиции элемента в массиве.

Большинство языков определяют начальный индекс массива как 0. Можно обращаться к любому элементу через индекс.

Количество элементов в простом массиве- фиксированное, менять можно только содержимое ячеек. Но есть динамические массивы (vector), в которых эта функция добавлена.

- *Одномерные,* как показано выше. Представляют собой ряд ячеек.
- *Многомерные*, массивы внутри массивов.

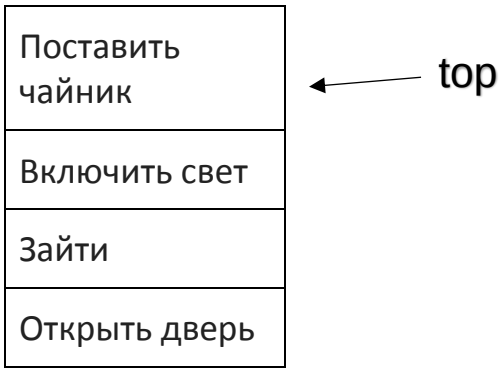
Например, основной массив – список из 3 имён (строки), каждое имя в свою очередь – массив, где элементы- буквы (столбцы).

	0	1	2	3
0	И	в	а	н
1	П	ё	т	р
2	В	е	р	а

Стеки

— абстрактный тип данных, представляющий собой список элементов, организованных по принципу LIFO (англ. last in — first out, «последним пришёл — первым вышел»).

Этот принцип заключается в том, что действия совершаются над элементом, расположенным самым первым. Чтобы сделать что-нибудь с элементами ниже, надо удалять всё, что до них. Например, функция «Отменить» в приложениях работает по LIFO.



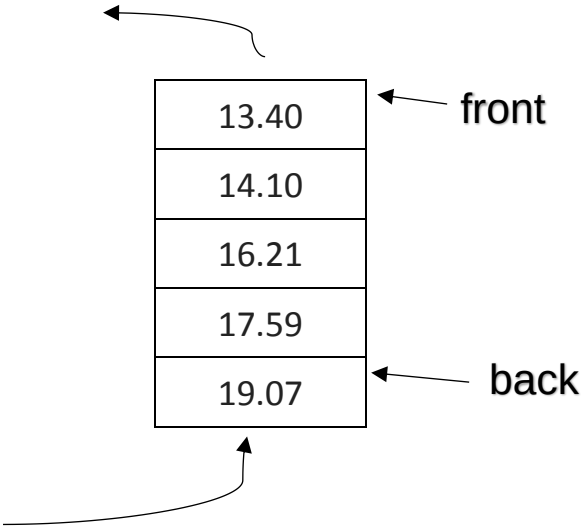
Основные операции:

- Push-вставляет элемент сверху
- Pop-возвращает верхний элемент после удаления из стека
- isEmpty-возвращает true, если стек пуст
- Top-возвращает верхний элемент без удаления из стека

Очереди

— хранит элемент последовательным образом с использованием принципа FIFO (First in First Out).

Элементарный пример очереди – очередь людей. Или посложнее-последовательность выполнения заказов в кафе. Тот, что поступил раньше, будет раньше готов.



Операции: в отличие от стека, можно вставлять элементы и в начало, и в конец очереди.

- Enqueue—) — вставляет элемент в конец очереди
- Dequeue () — удаляет элемент из начала очереди
- isEmpty () — возвращает значение true, если очередь пуста
- Top () — возвращает первый элемент очереди.

Связанный список

– массив где каждый элемент является отдельным объектом и состоит из двух элементов – данных и ссылки на следующий узел.

Принципиальным преимуществом перед массивом является структурная гибкость: порядок элементов связного списка может не совпадать с порядком расположения элементов данных в памяти компьютера, а порядок обхода списка всегда явно задаётся его внутренними связями

- *Однонаправленный*, каждый узел хранит адрес или ссылку на следующий узел в списке и последний узел имеет следующий адрес или ссылку как NULL.

  1->2->3->4->NULL

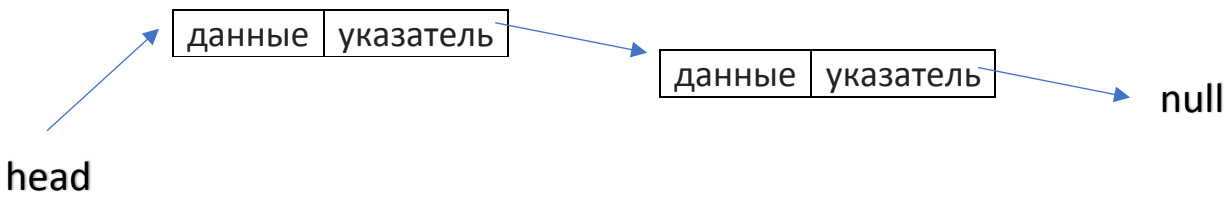
- *Двунаправленный*, две ссылки, связанные с каждым узлом, одним из опорных пунктов на следующий узел и один к предыдущему узлу.

  NULL<-1<->2<->3->NULL

- *Круговой*, все узлы соединяются, образуя круг. В конце нет NULL. Циклический связанный список может быть одно-или двукратным циклическим связанным списком.

  1->2->3->1

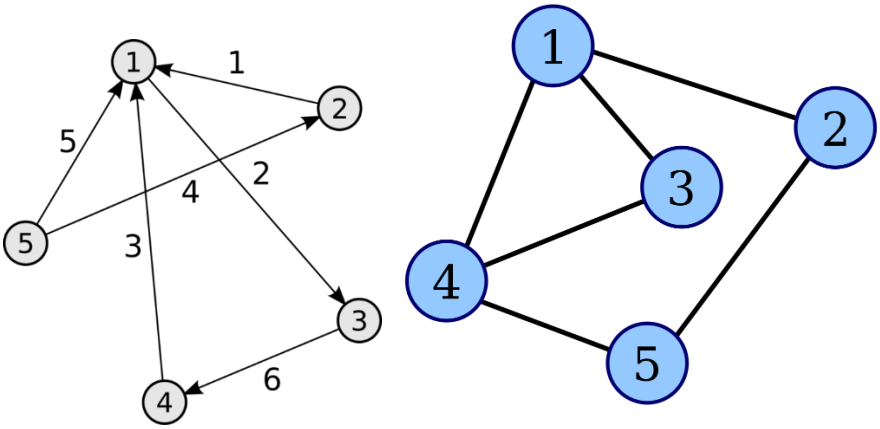
  Самое частое, линейный однонаправленный список. Пример – файловая система.



Графы

-это набор узлов (вершин), которые соединены друг с другом в виде сети ребрами (дугами).

- *Ориентированный*, ребра являются направленными, т.е. существует только одно доступное направление между двумя связными вершинами.
- *Неориентированные*, к каждому из ребер можно осуществлять переход в обоих направлениях.
- *Смешанные*



Встречаются в форме

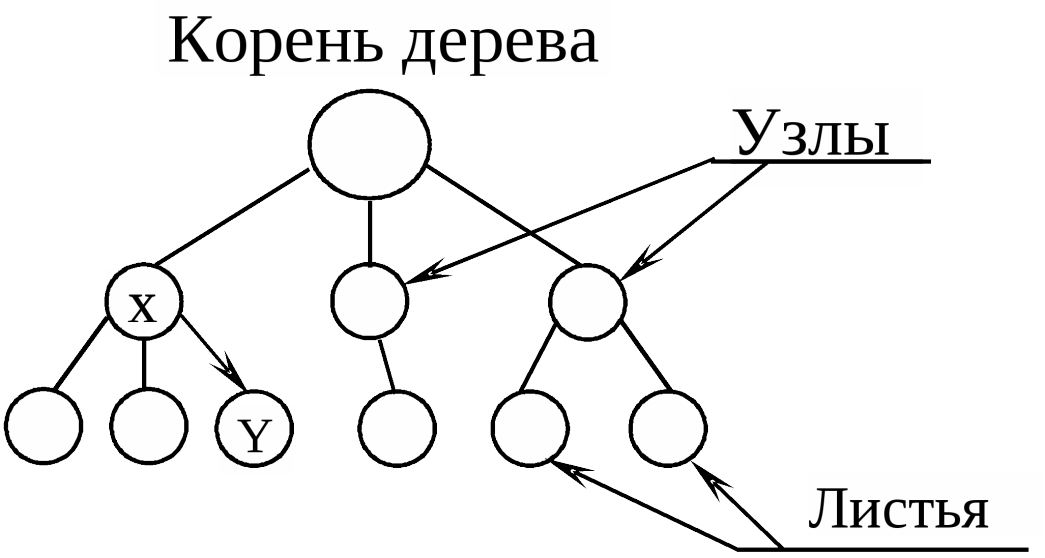
- Матрица смежности
- Список смежности

Обход графа можно совершать 2 способами:

- Поиск в ширину – обход по уровням
- Поиск в глубину – обход по вершинам

Деревья

-это иерархическая структура данных, состоящая из узлов (вершин) и ребер (дуг). Деревья по сути связанные графы без циклов.

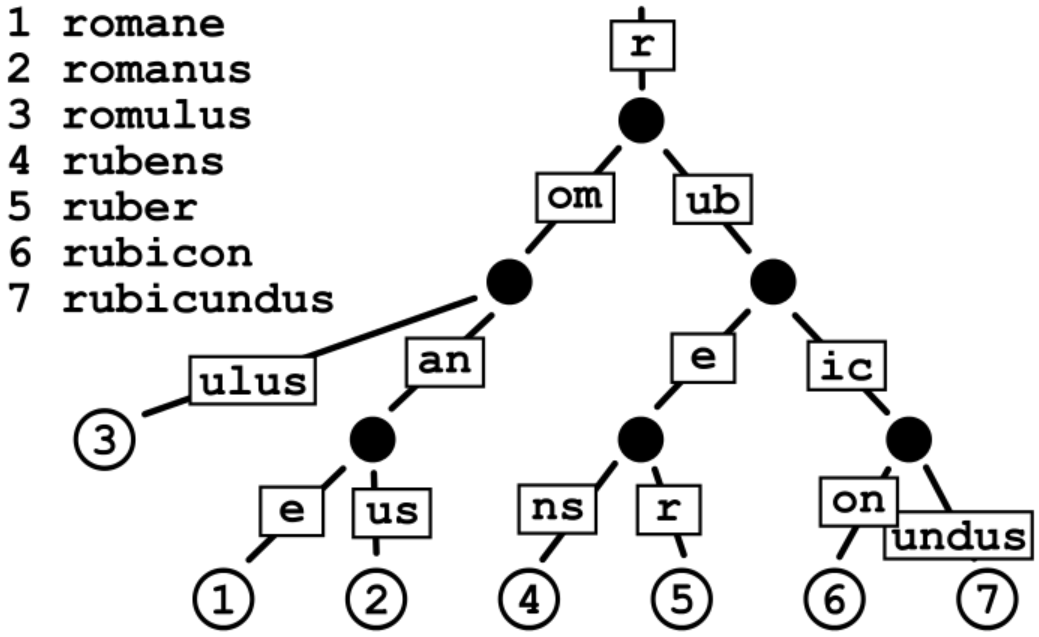


Типы деревьев:

- N дерево
- Сбалансированное дерево
- Бинарное дерево
- Дерево Бинарного Поиска
- AVL дерево
- 2-3-4 деревья

Trie (префиксное деревое)

Разновидность дерева для строк, быстрый поиск. Словари. T9.



Обход дерева может быть:

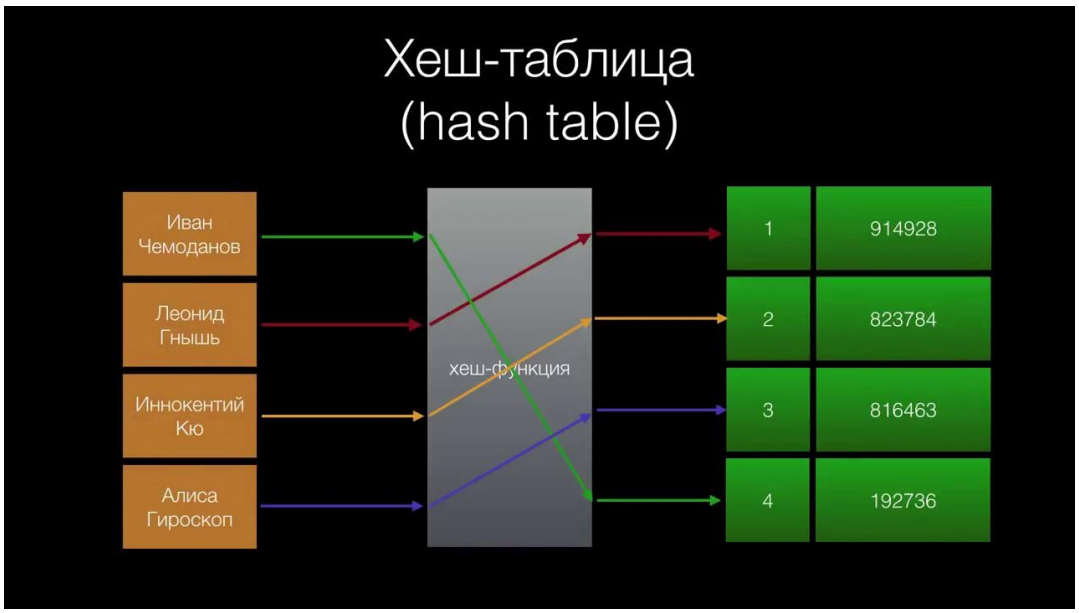
- В прямом порядке (сверху вниз) — префиксная форма.
- В симметричном порядке (слева направо) — инфиксная форма.
- В обратном порядке (снизу вверх) — постфиксная форма.

Хэш таблицы

Хэширование — это процесс, используемый для уникальной идентификации объектов и хранения каждого объекта в заранее рассчитанном уникальном индексе (ключе).
Объект хранится в виде пары «ключ-значение», а коллекция таких элементов называется «словарем». Каждый объект можно найти с помощью этого ключа.
По сути это массив, в котором ключ представлен в виде хеш-функции.

Эффективность хеширования зависит от

- Функции хеширования
- Размера хэш-таблицы
- Метода борьбы с коллизиями


Хеш-таблица
(hash table)
Иван Чемоданов
Леонид Гнышь
Иннокентий Кю
Алиса Гироскоп
хеш-функция
1
914928
2
823784
3
816463
4
192736