Теория

T1

Структуры данных

<https://proglib.io/p/8-data-structures/>

Структура данных – это контейнер, который хранит информацию в определенном виде.

Данные являются самой важной сущностью в информатике, а структуры позволяют хранить их в организованной форме.

1. Массив (Array)
2. Стек (Stack)
3. Очередь (Queue)
4. Связный список (Linked List)
5. Дерево (Tree)
6. Граф (Graph)
7. Префиксное дерево (Trie)
8. Хэш-Таблица (Hash Table)

Типы:

Массив – это самая простая и наиболее широко используемая из структур. Стеки и очереди являются производными от массивов.

* Одномерные массивы (как на картинке).
* Многомерные массивы (массивы массивов).

Операции:

* Insert – вставка.
* Get – получение элемента.
* Delete – удаление.
* Size – получение общего количества элементов в массиве.

Найти второй минимальный элемент

var a = [1,61,2,67,-30,12,-77,100,126,-102,6];

var a\_min = a[0];

for (i = 1; i < a.length; i++) {

   if (a[i] <= a\_min) {

      a\_min = a[i];

   }

}

alert("Минимальное значение = " + a\_min);

Первые не повторяющиеся целые числа

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | mas=**new** array(20);  mas(0)=math.floor(math.random\*20);  b=1;  **for** (i=1;i++;i=20) {  **do** {  b=1;  a=math.floor(math.random\*20);  **for** (j=0;j++;j=i) {  **if** (mas(j)==a) b=0;  } }  while (b==0)  mas(i)=a;  } |

* [Объединить два отсортированных массива](https://pas1.ru/arrayconflux).

var arr1 = [1, 3, 5, 6];

var arr2 = [3, 6, 10, 11, 12];

arr1.concat(arr2.filter(ele => !arr1.includes(ele)));

console.log(arr1);

output :- [1, 3, 5, 6, 10, 11, 12]

Стеки

### **Основные операции со стеками**

* Push – вставка элемента наверх стека.
* Pop – получение верхнего элемента и его удаление.
* isEmpty – возвращает true, если стек пуст.
* Top – получение верхнего элемента без удаления.

 Так работает метод LIFO (Last In First Out, последним пришел – первым ушел).

Книги

## Очереди

очередь реализует метод FIFO (First in First Out, первым пришел – первым ушел).

 пример этих структур в реальной жизни – очереди людей в билетную кассу

### **Основные операции с очередями**

* Enqueue – вставка в конец.
* Dequeue –  удаление из начала.
* isEmpty – возвращает true, если очередь пуста.
* Top – получение первого элемента.

## Связный список

сеть узлов, каждый из которых содержит данные и указатель на следующий узел в цепочке. Также есть указатель на первый элемент – head

[Типы связных списков](https://cyberpedia.su/3xefd6.html):

* Однонаправленный
* Двунаправленный

### **Основные операции со связными списками**

* InsertAtEnd – вставка в конец.
* InsertAtHead – вставка в начало.
* Delete – удаление указанного элемента.
* DeleteAtHead – удаление первого элемента.
* Search – получение указанного элемента.
* isEmpty – возвращает true, если связный список пуст.

### **Часто задаваемые вопросы о связных списках**

## Графы

Граф представляет собой набор узлов, соединенных друг с другом в виде сети. Узлы также называются вершинами. Пара (**x**, **y**) называется ребром, которое указывает, что вершина **x**соединена с вершиной **y**. Ребро может содержать вес/стоимость, показывая, сколько затрат требуется, чтобы пройти от **x** до **y**.

Типы графов:

* Неориентированный
* Ориентированный

В языке программирования графы могут быть представлены в двух формах:

* [Матрица смежности](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B0_%D1%81%D0%BC%D0%B5%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D0%B8)
* [Список смежности](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_%D1%81%D0%BC%D0%B5%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D0%B8)

Общие [алгоритмы обхода графов](http://www.intuit.ru/studies/courses/648/504/lecture/11474?page=2):

* В ширину
* В глубину

## Деревья

Дерево – это иерархическая структура данных, состоящая из вершин (узлов) и ребер, соединяющих их. Они похожи на графы, но есть одно важное отличие: в дереве не может быть цикла.

## Префиксное дерево

[Префиксные деревья](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D1%84%D0%B8%D0%BA%D1%81%D0%BD%D0%BE%D0%B5_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE) (tries) – древовидные структуры данных, эффективные для решения задач со строками. Они обеспечивают быстрый поиск и используются преимущественно для поиска слов в словаре, автодополнения в поисковых системах и даже для IP-маршрутизации.

## Хеш-Таблица

Хеширование – это процесс, используемый для уникальной идентификации объектов и хранения каждого из них в некотором предварительно вычисленном уникальном индексе – ключе. Итак, объект хранится в виде пары **ключ-значение**, а коллекция таких элементов называется **словарем**.

Связный список и массив - это различные структуры данных, которые не привязаны к конкретному языку программирования.

**Массив**

Массив - это совокупность однотипных данных, расположенных непрерывно в памяти. Доступ к элементу осуществляется по индексу за O(1) - мы обращаемся непосредственно к нужному участку памяти.

**Связанный список**

Доступ к элементу в связном списке в среднем занимает O(N) путем перебора элементов в поисках нужного. Способы доступа к элементам отличаются по реализации и от языка программирования. Например, на Java в стандартном классе LinkedList в зависимости от ситуации проход элементов может начинаться как с начала, так и с конца списка. И поиск элемента может осуществляться как по индексу, так и по сравнению элементов.

Связный список требует больших расходов памяти при прочих равных условиях за счет хранения указателей на следующий/предыдущий элементы и особенностей внутренней реализации.

<https://prog-cpp.ru/data-list/>

T3

Системное

программирование

В простейшем случае интерпретатор читает исходный текст программы по одной строке за раз, выполняет эту строку и только после этого переходит к следующей. Так работали ранние версии языка Basic. В языках типа Java исходный текст программы сначала конвертируется в промежуточную форму, а затем интерпретируется. В этом случае программа также интерпретируется в процессе выполнения.

Компилятор читает сразу всю программу и конвертирует ее в *объектный код*, то есть транслирует исходный текст программы в форму, более пригодную для непосредственного выполнения компьютером. Объектный код также называют *двоичным* или *машинным кодом*. Когда программа скомпилирована, в ее коде уже нет отдельных строк исходного кода.

В общем случае интерпретируемая программа выполняется медленнее, чем скомпилированная. Необходимо помнить, что компилятор преобразует исходный текст программы в объектный код, который выполняется компьютером непосредственно. Значит, потеря времени на компиляцию происходит лишь единожды, а в случае интерпретации — каждый раз при очередной компиляции фрагмента программы в процессе ее выполнения.

Навыки

Git

<https://git-scm.com/book/ru/v2>

ссылки

<https://cyberleninka.ru/article/n/28-magicheskih-mer-razrabotki-bezopasnogo-programmnogo-obespecheniya/viewer>

<https://www.securitylab.ru/blog/personal/crypto-anarchist/312897.php>

<https://geekbrains.ru/posts/ide_negative>

<https://www.internet-technologies.ru/articles/10-luchshih-ide.html>

<https://tproger.ru/translations/sorting-for-beginners/>

<https://medium.com/nuances-of-programming/%D0%B2%D1%81%D0%B5-%D1%87%D1%82%D0%BE-%D0%BD%D1%83%D0%B6%D0%BD%D0%BE-%D0%B7%D0%BD%D0%B0%D1%82%D1%8C-%D0%BE-%D0%B4%D1%80%D0%B5%D0%B2%D0%BE%D0%B2%D0%B8%D0%B4%D0%BD%D1%8B%D1%85-%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0%D1%85-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-d750444a77ec>

<https://qna.habr.com/q/28207>

<https://proglib.io/p/painful-git/>

<http://10.131.72.243:15871/cgi-bin/blockpage.cgi?ws-session=18446744073659251607>

<https://pai-bx.com/wiki/more/2343-git-comands/>

<https://software-testing.ru/library/testing/testing-automation/2552-unit-tests>

<https://habr.com/ru/post/336030/>

<https://metanit.com/sharp/aspnet5/22.1.php>

<https://ru.stackoverflow.com/questions/436370/%D0%9A%D0%B0%D0%BA-%D0%BD%D0%B0%D0%B9%D1%82%D0%B8-%D1%81%D1%80%D0%B5%D0%B4%D0%BD%D0%B5%D0%B5-%D0%B0%D1%80%D0%B8%D1%84%D0%BC%D0%B5%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%BE%D0%B5-%D0%B2%D1%81%D0%B5%D1%85-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D0%BE%D0%B2-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%B0>

<https://habr.com/ru/post/339656/>

<https://habr.com/ru/company/mailru/blog/316634/>

<https://habr.com/ru/post/335920/>

<https://proglib.io/p/sort-algorithms/>

https://proglib.io/p/sort-gif/