МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «Полоцкий государственный университет»

Факультет информационных технологий

Кафедра технологий программирования

**РЕФЕРАТ**

**по дисциплине “Алгоритмы и структуры данных”**

ВЫПОЛНИЛ студент группы 21-ИТ-1

Макеёнок Д.И.

ПРОВЕРИЛ преподаватель

Деканова М.В.

Новополоцк, 2022 г.

**Раздел 1**

**Тема 1.1 Введение в дисциплину. Общие понятия и определения.**

**Структура данных** – это множество элементов данных и внутренних связей между ними.

**Объект данных** – представляет объект, имеющий данные.

Тип данных – это способ классификации различных типов данных, таких как целочисленные, строковые и т. д., Который определяет значения, которые можно использовать с данными соответствующего типа, тип операций, которые можно выполнять с данными соответствующего типа.

Есть два типа данных –

* Встроенный тип данных
* Тип произвольных данных

Те типы данных, для которых язык имеет встроенную поддержку, называются встроенными типами данных. Например, большинство языков предоставляют следующие встроенные типы данных –

* Целые
* Логическое (верно, неверно)
* Плавающая (десятичные числа)
* Персонажи и Струны

Те типы данных, которые не зависят от реализации, поскольку они могут быть реализованы тем или иным способом, называются производными типами данных. Эти типы данных обычно создаются путем сочетания основных или встроенных типов данных и связанных с ними операций.

Например –

* Список
* Массив
* Стек
* Очередь

**Основные операции:**

Данные в структурах данных обрабатываются определенными операциями. Конкретная выбранная структура данных в значительной степени зависит от частоты операции, которая должна быть выполнена над структурой данных.

**Тема 1.2 Статические структуры данных.**

**Статические структуры** относятся к разряду не примитивных структур, которые, фактически, представляют собой структурированное множество примитивных, базовых, структур.

Например, вектор может быть представлен упорядоченным множеством чисел.

Статические структуры в языках программирования связаны со **структурированными типами**.

**Структурированные типы** в языках программирования являются теми средствами интеграции, которые позволяют строить структуры данных сколь угодно большой сложности.

К таким типам относятся:

* массивы,
* записи (в некоторых языках - структуры)
* и множества (этот тип реализован не во всех языках).

**Массивы** являются наиболее широко используемыми структурами данных и предусмотрены во всех языках программирования.

Массив состоит из элементов одного типа, называемого **базовым**, поэтому структура массива однородна.

**Базовый тип** может быть, как скалярным, так и структурированным, т.е. элементами массива могут быть числа, символы, строки, структуры, в том числе и массивы.

**Свободными** называют двухмерные массивы, размер каждой из строк, которых может быть различным.

Иногда при использовании матриц имеет смысл хранить только какую-либо часть матрицы.

К таким матрицам можно отнести **треугольные** и **разреженные матрицы**.

В треугольной матрице Аn\*n все элементы выше или ниже главной диагонали являются нулевыми.

Если нулевыми являются элементы выше диагонали (нижняя треугольная матрица), то ненулевыми будут

элементы

a11 ; a21 , a22 ; a31 , a32 ; a33 ; …; an1 , an2 , …, ann .

Нижнетреугольную матрицу целесообразно хранить в виде одномерного массива (вектора) В с числом

элементов **m=n\*(n+1)/2**.

**Разреженными** называются матрицы, содержащие большое количество нулевых элементов.

Они обычно встречаются в научных приложениях, при представлении графов в программах.

Представление таких матриц в виде двухмерных массивов приводит к нерациональному использованию

памяти и неэффективности выполняемых над ними операций.

**Запись** представляет собой совокупность ограниченного числа логически связанных компонент, принадлежащих к разным типам.

Компоненты записи называются **полями**, каждое из которых определяется именем. Сами поля, в свою очередь, могут быть составными.

**Мощность** составляющего типа есть произведение мощностей составляющих его типов.

Каждый элемент в записи имеет свое уникальное имя, но такое же имя может использоваться в других записях или для обозначения других объектов программы.

**Множество** – это структурированный тип данных, представляющий собой набор взаимосвязанных по какому-либо признаку или группе признаков неповторяющихся объектов, которые можно рассматривать как единое целое.

Каждый объект в множестве называется **элементом множества**.

Все элементы множества должны принадлежать одному из скалярных типов, кроме вещественного.

Этот тип называется **базовым типом** множества. Базовый тип задается диапазоном или перечислением.

**Тест по первому разделу**

1.Какой тип данных существует?

А) Внешний тип данных.

Б) Внутренний тип данных.

**В)** Встроенный тип данных.

Г) Пустой тип данных.

2. Каким типом данных являются десятичные числа?

А) Целые.

Б) Логическое.

**В)** Плавающая.

Г) Персонаж и Струны.

3. Объект данных это?

**А)** объект, имеющий данные.

Б) объект, имеющий значения только цифр.

В) объект, имеющий всю информацию.

Г) объект, имеющий значения.

4. Что из перечисленного не является произвольным типом данных?

А) Очередь.

Б) Стек.

В) Список.

**Г)** Логический тип данных.

5. Какая структура данных состоит из элементов одного типа?

А) Запись.

**Б)** Массив.

В) Множество.

Г) Матрицы.

6. Как называются элементы одного типа в массиве?

А) Структурированный тип.

**Б)** Базовый тип.  
В) Скалярный тип

Г) Обычный тип.

7. Как называют двухмерный массив, размер каждой из строк, которых может быть различным?

**А)** Свободным.

Б) Произвольным.

В) Разно размерный.

Г) Открытый.

8. В какой матрице все элементы выше или ниже главной диагонали являются нулевыми?

А) Квадратная.

Б) Разряженная.

В) Диагональная.

**Г)** Треугольная.

9. Компоненты какой структуры данных называются полями?

А) Множество.

Б) Массив.

**В)** Запись.

Г) Матрицы.

10. Какой тип данных представляет собой набор взаимосвязанных по какому-либо признаку или группе признаков неповторяющихся объектов?

**А)** Множество.

Б) Массив.

В) Запись.

Г) Матрицы.

**Раздел 2**

**Тема 2.1 Список.**

**Список** – это линейная динамическая структура данных, у каждого элемента может быть только один предок и только один потомок.

По сути своей это очень похоже на обыкновенный массив, с той

лишь разницей, что размер его не имеет ограничений.

Элемент списка доступен в программе через указатель.

В программе список задается посредством **заголовка** – указателя на первый

элемент списка.

**Порядок следования элементов** определяется последовательностью связей

между элементами.

**Логический (порядковый) номер элемента** списка также задается его

естественной нумерацией в цепочке элементов.

Список является **структурой данных с последовательным доступом**.

Список обладает свойством **локальности изменений**: при вставке/удалении

элемента изменения касаются только текущего и его соседей.

Преимущества списков проявляются в таких структурах данных, где операции изменения порядка превалируют над операциями доступа и поиска.

списки бывают следующих видов:

* односвязные;
* двусвязные;
* циклические.

**Односвязный линейный список** – это структура данных, представляющая собой последовательность элементов, в каждом из которых хранится значение и указатель на следующий элемент списка. В последнем элементе указатель на следующий элемент равен NULL.

**Двусвязный линейный список** (ДЛС) – Каждый узел ДЛС содержит

два поля указателей: на следующий и на предыдущий узел. Поле указателя на следующий узел последнего узла указывает на NULL. Поле указателя на предыдущий узел первого узла также указывает на NULL.

**Тема 2.2 Стек.**

**Стеком** называется упорядоченный набор элементов, в котором размещение новых и удаление существующих происходит с одного конца,

называемого вершиной.

**Дисциплина обслуживания** — это совокупность правил (упорядочение и

алгоритм) обслуживания элементов динамической структуры данных. В

зависимости от дисциплины обслуживания различают те или иные структуры динамических данных.

Принцип работы стека сравнивают со стопкой листов бумаги: чтобы взять

второй сверху, нужно снять верхний.

В стеке реализуется дисциплина обслуживания LIFO:

• LAST— последний

• INPUT — вошел

• FIRST — первый

• OUTPUT — вышел

Различают аппаратный и программный стек.

**Аппаратный стек** используется для хранения адресов возврата из функций

и их аргументов.

**Программный стек** – это пользовательская модель (структура) данных.

Основные операции, производимые со стеком:

* создание стека;
* печать (просмотр) стека;
* добавление элемента в вершину стека;
* извлечение элемента из вершины стека;
* проверка пустоты стека;
* очистка стека.

**Тема 2.3 Очередь**

**Очередью** называется упорядоченный набор элементов, которые могут удаляться с её начала и помещаться в её конец.

Очередь организована, в отличие от стека, согласно дисциплине обслуживания, FIFO:

• FIRST — первый

• INPUT — вошел

• FIRST — первый

• OUTPUT — вышел

В очереди доступны два элемента (две позиции): начало очереди и конец очереди.

Поместить элемент можно только в конец очереди, а взять элемент только из ее начала.

Примером может служить обыкновенная очередь в магазине.

В программировании очереди применяются очень широко — например, при моделировании, буферизованном вводе-выводе или диспетчеризации задач в операционной системе.

В программе очередь можно реализовывать в виде:

* статического массива с ограничением на размер в очереди;
* динамического массива;
* односвязного списка;
* двусвязного списка.

Очередь с приоритетами может быть реализована одним из двух способов:

**Очередь с приоритетным включением.** В этом случае элемент, который

добавляется в очередь, сразу размещается в ней в соответствии с его

приоритетом.

**Очередь с приоритетным исключением.** В этом случае новый элемент

просто добавляется в конец очереди. А вытягивание элемента из очереди

осуществляется по его приоритету, то есть вытягивается элемент с

наивысшим приоритетом.

**Операции над очередями:**

* добавление элемента в очередь (помещение в хвост);
* удаление элемента из очереди (удаление из головы);
* проверка, пуста ли очередь;
* очистка очереди.

**Дек** является особым видом очереди.

**Дек** – это структура данных, представляющая собой последовательность элементов, в которой можно добавлять и удалять в произвольном порядке элементы с двух сторон.

Первый и последний элементы дека соответствуют входу и выходу дека.

**Тест по второму разделу**

1. Как называется динамическая структура данных, у каждого элемента которой может быть только один предок и только один потомок?

**А)** Список.

Б) Очередь.

В) Стек.

Г) Массив.

2.Какого вида списков не существует?

А) Односвязный список.

**Б)** Несвязный список.  
В) Двусвязный список.

Г) Циклический список.

3. В каком списке каждый узел содержит два поля указателей: на следующий и на предыдущий узел?

А) Односвязный список.

Б) Односвязный циклический список.  
**В)** Двусвязный список.

Г) Циклический список.

4. Как называется совокупность правил обслуживания элементов динамической структуры данных?

**А)** Дисциплина обслуживания.

Б) Правила обслуживания.

В) Порядок обслуживания.

Г) Правила порядка.

5. Что организовано согласно дисциплине обслуживания, FIFO?

А) Список.

**Б)** Очередь.

В) Стек.

Г) Массив.

6. Что организовано согласно дисциплине обслуживания, LIFO?

А) Список.

Б) Очередь.

**В)** Стек.

Г) Массив.

7. Какой стек используется для хранения адресов возврата из функций

и их аргументов?

А) Логический.

Б) Вычислительный.

В) Программный.

**Г)** Аппаратный.

8. Как называется очередь с приоритетом в которой элемент добавляется в очередь, сразу размещается в ней в соответствии с его приоритетом?

**А)** Очередь с приоритетным включением.

Б) Очередь с приоритетным исключением.

В) Очередь с приоритетной заменой.

Г) Очередь с приоритетным перемещением.

9. Как называется очередь с приоритетом в которой элемент добавляется в очередь, и после вытягивается в соответствии с его приоритетом?

А) Очередь с приоритетным включением.

**Б)** Очередь с приоритетным исключением.

В) Очередь с приоритетной заменой.

Г) Очередь с приоритетным перемещением.

10. Как называется структура данных в которой можно добавлять и удалять в произвольном порядке элементы с двух сторон?

А) Список.

Б) Очередь.

В) Стек.

**Г)** Дек.

**Раздел 3**

**Тема 3.1 Деревья.**

**Дерево –** это нелинейная иерархическая структура данных. Она состоит из узлов и ребер, которые соединяют узлы.

**Узел —** это объект, в котором есть ключ или значение и указатели на дочерние узлы. Узлы, у которых нет дочерних узлов, называют листами или терминальными узлами. Узлы, у которых есть хотя бы один дочерний узел, называются **внутренними**.

**Ребро** связывает два узла.

**Корень** — это самый верхний узел дерева. Его ещё иногда называют корневым узлом.

**Высота узла** — это максимальная длина пути от этого узла к самому нижнему узлу (листу).

**Глубина вложенности узла** — длина пути от корня до этого узла.

**Высота дерева** — это высота корневого узла или глубина самого

глубокого узла.

**Степень узла** — это общее количество ребер, которые соединены с этим узлом.

Дерево, в котором задан порядок называется **упорядоченным деревом** со структурой данных, заданной перед именованием и называемой структурой данных упорядоченного дерева.

**Обход дерева** — вид [обхода графа](https://ru.wikipedia.org/w/index.php?title=%D0%9E%D0%B1%D1%85%D0%BE%D0%B4_%D0%B3%D1%80%D0%B0%D1%84%D0%B0&action=edit&redlink=1), обусловливающий процесс посещения каждого узла [структуры дерева данных](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_(%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)) ровно один раз

Существует три основных способа обхода в глубину.

**Прямой** (pre-order) Посетить корень, обойти левое поддерево, обойти правое поддерево.

**Симметричный** или поперечный (in-order) Обойти левое поддерево, посетить корень, обойти правое поддерево.

**В обратном порядке** (post-order) Обойти левое поддерево, обойти правое поддерево, посетить корень.

Дерево, у которого узлам сопоставлены метки, называется **помеченным деревом**.

**Метка узла** - это не имя узла, а значение, которое "хранится" в узле.

В некоторых приложениях требуется изменять значение метки, поскольку имя узла сохраняется постоянным.

**Деревья выражений** представляют код в виде древовидной структуры, где каждый узел является выражением

**Тема 3.2 АТД «Дерево».**

**Дерево** — это абстрактный тип данных (АТД) для иерархического хранения элементов. За исключением элемента во главе дерева (root), каждый элемент структуры имеет родителя (parent) и ноль или более дочерних элементов (children).

Операторы, выполняемые над деревьями:

**PARENT(n, Т).**

Эта функция возвращает родителя (parent) узла n в дереве Т. Если n является корнем, который не имеет родителя, то в этом случае возвращается Λ ("нулевой узел“), что указывает на то, что мы выходим за пределы дерева.

**LEFTMOST\_CHILD(n, Т).**

Данная функция возвращает самого левого сына узла n в дереве Т. Если n является листом (и поэтому не имеет сына), то возвращается Λ.

**RIGHT\_SIBLING(n, Т).**

Эта функция возвращает правого брата узла n в дереве Т и значение Λ, если такового не существует. Для этого находится родитель р узла n и все сыновья узла р, затем среди этих сыновей находится узел, расположенный непосредственно справа от узла n.

**Двоичное дерево** – древовидная структура данных, в которой у родительских узлов не может быть больше двух детей.

Характеристики дерева Хаффмана -

* Для того же набора весов доступное дерево Хаффмана не обязательно уникально.
* Левое и правое поддеревья дерева Хаффмана можно поменять местами, поскольку это не влияет на взвешенную длину пути дерева.
* Все узлы с весами - это конечные узлы, а узлы без весов - все корневые узлы суббинарного дерева.
* Узлы с большими весами находятся ближе к корневому узлу дерева Хаффмана, а узлы с меньшими весами находятся дальше от корневого узла дерева Хаффмана.
* В дереве Хаффмана есть только листовые узлы и узлы со степенью 2, и нет узлов со степенью 1.
* Дерево Хаффмана с n листовыми узлами имеет 2n-1 узлов.

Этапы построения дерева Хаффмана.

1. Рассмотрим заданные n весов как n бинарных деревьев только с корневыми узлами (без левого и правого потомков), чтобы сформировать множество HT. Вес каждого дерева - это вес узла.
2. Выберите два двоичных дерева с наименьшими весами из набора HT, чтобы сформировать новое двоичное дерево, вес которого является суммой весов двух двоичных деревьев.
3. Удалите два двоичных дерева, выбранных на шаге 2, из набора HT, и добавьте вновь полученное двоичное дерево на шаге 2 в набор HT.
4. Повторяйте шаги 2 и 3 до тех пор, пока набор HT не будет содержать только одно дерево, которое является деревом Хаффмана.

**Тема 3.3 Основные виды деревьев**

**Бинарное дерево поиска** — структура данных для работы с упорядоченными множествами. Бинарное дерево поиска обладает следующим свойством: если узел бинарного дерева с ключом, то все узлы в левом поддереве должны иметь ключи, меньшие, а в правом поддереве большие.

**АВЛ-дерево** – сбалансированное по высоте двоичное дерево поиска, в котором у любой вершины высота левого и правого поддеревьев различаются не более чем на 1.

**2-3-дерево** — структура данных, являющаяся B-деревом Степени 1, страницы которого могут содержать только 2-вершины (вершины с одним полем и 2 детьми) и 3-вершины (вершины с 2 полями и 3 детьми).

Свойства 2-3 дерева.

* Все не листовые вершины содержат одно поле и 2 поддерева или 2 поля и 3 поддерева.
* Все листовые вершины находятся на одном уровне (на нижнем уровне) и содержат 1 или 2 поля.
* Все данные отсортированы (по принципу двоичного дерева поиска).

Не листовые вершины содержат одно или два поля, указывающие на диапазон значений в их поддеревьях.

**B-дерево** — это особый тип сбалансированного дерева поиска, в котором каждый узел может содержать более одного ключа и иметь более двух дочерних элементов. Из-за этого свойства B-дерево называют сильно ветвящимся.

используются Б-деревья:

* В базах данных и файловых системах.
* Для хранения блоков данных (вторичные носители).
* Для многоуровневой индексации.

**Красно-черные деревья** относятся к сбалансированным бинарным деревьям

поиска.

Как бинарное дерево, красно-черное обладает свойствами:

1) Оба поддерева являются бинарными деревьями поиска.

2) Для каждого узла с ключом k выполняется критерий

упорядочения:

ключи всех левых потомков <= k < ключи всех правых потомков.

Это неравенство должно быть истинным для всех потомков узла, а не

только его дочерних узлов.

**Тест по третьему разделу**

1. Как называется нелинейная иерархическая структура данных состоящая из узлов и ребер, которые соединяют узлы?

**А)** Дерево.

Б) Очередь.

В) Стек.

Г) Массив.

2.Что связывает два узла?

А) Прямая.

Б) Дочерний узел.

В) Корень.

**Г)** Ребро.

3. Как называется самый верхний узел дерева?

А) Высота дерева.

Б) Дочерний узел.

**В)** Корень.

Г) Ребро.

4. Как называется максимальная длина пути от узла к самому нижнему узлу?

А) Высота дерева.

**Б)** Высота узла.

В) Высота Корня.

Г) Высота Ребра.

5. Как называется значение, которое "хранится" в узле?

А) Кусочек узла.

Б) Название узла.

В) Значение узла.

**Г)** Метка узла.

6. Какая функция возвращает родителя узла n в дереве Т?

A)LEFTMOST\_CHILD(n, Т).

**Б)** PARENT(n, Т).

В)RIGHT\_SIBLING(n, Т).

Г) LABEL(n, Т).

7. Как называется древовидная структура данных, в которой у родительских узлов не может быть больше двух детей?

А)АВЛ-дерево.

Б) 2-3-дерево.

В) B-дерево.

**Г)** Двоичное дерево.

8. Как называется сбалансированное по высоте двоичное дерево поиска, в котором у любой вершины высота левого и правого поддеревьев различаются не более чем на 1?

**А)** АВЛ-дерево.

Б) 2-3-дерево.

В) B-дерево.

Г) Двоичное дерево поиска.

9. Как называется особый тип сбалансированного дерева поиска, в котором каждый узел может содержать более одного ключа и иметь более двух дочерних элементов?

А) Красно-черное дерево.

Б) 2-3-дерево.

**В)** B-дерево.

Г) Двоичное дерево поиска.

10. Как называется структура данных, являющаяся B-деревом Степени 1, страницы которого могут содержать только 2-вершины и 3-вершины.

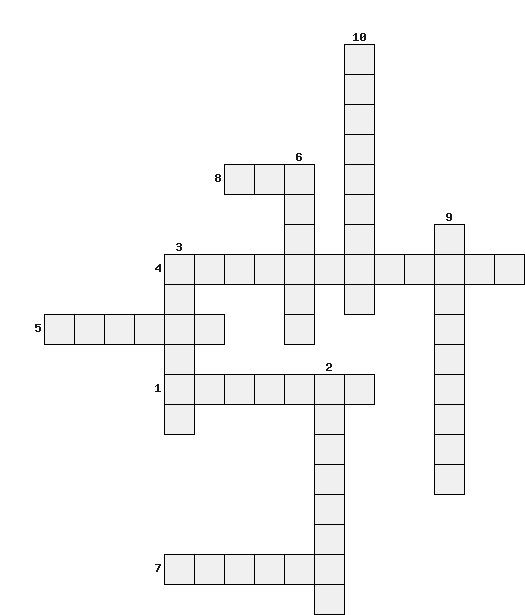
А) Красно-черное дерево.

**Б)** 2-3-дерево.

В) B-дерево.

Г) Двоичное дерево поиска.

**Кроссворд**



1. Как называется упорядоченный набор элементов в котором они могут удаляться с его начала и помещаться в его конец?

2. Как называется дерево, в котором у родительских узлов, не может быть больше двух детей?

3. Как называется динамическая структура данных, у каждого элемента которой может быть только один предок и только один потомок?

4. Обойти левое поддерево, посетить корень, обойти правое поддерево. Как называется такой обход?

5. Как называется [структура данных](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), хранящая набор значений, идентифицируемых по индексу или набору индексов?

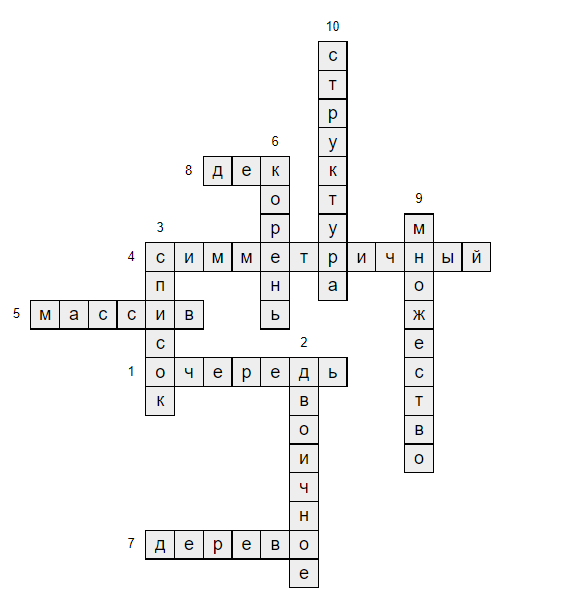
6. Как называется самый верхний узел дерева?

7. Как называется нелинейная иерархическая структура данных состоящая из узлов и ребер, которые соединяют узлы?

8.Как называется структура данных в которой можно добавлять и удалять в произвольном порядке элементы с двух сторон?

9. Какой тип данных представляет собой набор взаимосвязанных по какому-либо признаку или группе признаков неповторяющихся объектов?

10. Что называется множеством элементов данных и внутренних связей между ними?



**Список литературы**

1.Теоретичекий лекционным материал.

2. Методические указания к лабораторным работам.

3. Структуры данных и основные понятия алгоритма:

<https://coderlessons.com/tutorials/kompiuternoe-programmirovanie/izuchite-strukturu-dannykh-i-algoritmy/struktury-dannykh-i-osnovnye-poniatiia-algoritma>

4. Дерево (структура данных):

<https://ru.wikipedia.org/wiki/Дерево_(структура_данных)#Упорядочивание_деревьев> ,

<https://codechick.io/tutorials/dsa/dsa-trees>

5. Код Хаффмана:

<https://ru.wikipedia.org/wiki/Код_Хаффмана>

6. Обход дерева:

<https://ru.wikipedia.org/wiki/Обход_дерева>

7. Массив:

<https://ru.wikipedia.org/wiki/Массив_(тип_данных)>

8. Списки:

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

9. Стек:

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

10. Очередь:

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

11. Двоичное дерево:

<https://codechick.io/tutorials/dsa/dsa-binary-tree>

12. АВЛ-дерево:

<https://ru.wikipedia.org/wiki/АВЛ-дерево>