1. Линейные структуры данных – это структуры данных, в которых переход от одного элемента данных к другому не зависит от каких-либо логических условий, т.е. в линейных структурах используются лишь безусловные связи элементов.

1.1 Список Может всё то же самое, что и массив, но позволяет добавлять элементы в любое место, удалять элементы из любого места и получать текущее количество элементов.

1.2 Ассоциативный массив

1.3 Хеш-таблица - это обычный массив с необычной адресацией, задаваемой хеш-функцией. Лучший выбор, если не нужна сортировка информации, а только быстрый доступ к ней. Тратится дополнительная память.

преимущества:

Важное свойство хеш-таблиц состоит в том, что, при некоторых разумных допущениях, все три операции (поиск, вставка, удаление элементов) в среднем выполняются за время O(1), время для наихудшего случая - O(n).

недостатки:

Итерация не в порядке возрастания ключей

Необходимость «перехеширования» при увеличении числа хранимых объектов (?)

нельзя реализовать быстро работающие дополнительные операции MIN, MAX и алгоритм обхода всех хранимых пар в порядке возрастания или убывания ключей (?)

не поддерживает упорядоченности, и не сохраняет порядок следования элементов (?)

возможность коллизий

реализация:

C#: Hashtable

1.4 Стек Набор элементов одного типа, упорядоченных таким образом, чтобы добавлять и доставать элементы можно было только с одного конца. Операции: добавить элемент в стек, достать из стека последний добавленный элемент, проверить, является ли стек пустым.

1.5 Очередь Набор элементов одного типа, упорядоченных таким образом, чтобы добавлять их можно было только в один конец, а получать - с другого конца. Операции: добавить элемент в конец очереди, достать первый элемент из очереди, получение размера очереди.

1.5.1 Очередь с приоритетом

1.6 Дек - особый вид очереди. Дек (от англ. deq - double ended queue,т.е очередь с двумя концами) - это такой последовательный список, в котором как включение, так и исключение элементов может осуществляться с любого из двух концов списка. Частный случай дека - дек с ограниченным входом и дек с ограниченным выходом.

поддерживаемые операции:

включение элемента справа;

включение элемента слева;

исключение элемента справа;

исключение элемента слева;

определение размера;

очистка.

1.7 Буферное окно

2. Граф

2.1 Список рёбер

2.2 Деревья

2.2.1 2-3-дерево

2.2.2 Дерево отрезков

2.2.3 Красно-чёрное дерево

2.2.4 BSP-дерево

2.2.5 B-дерево

Основное предназначение: B-дерево предназначено для хранения информации на жёстком диске. Время произвольного доступа к жёсткому диску очень велико (миллисекунды), поскольку оно определяется скоростью вращения диска и перемещения головок. Поэтому важно уменьшить количество узлов, просматриваемых при каждой операции, то есть высоту дерева, что достигается путём высокой ветвистости.

преимущества:

Во всех случаях полезное использование пространства вторичной памяти занимает свыше 50 %. С ростом степени полезного использования памяти не происходит снижения качества обслуживания.

Произвольный доступ к записи реализуется посредством малого количества подопераций (обращения к физическим блокам).

В среднем достаточно эффективно реализуются операции включения и удаления записей; при этом сохраняется естественный порядок ключей с целью последовательной обработки, а также соответствующий баланс дерева для обеспечения быстрой произвольной выборки.

Неизменная упорядоченность по ключу обеспечивает возможность эффективной пакетной обработки

недостатки:

Основной недостаток В-деревьев состоит в отсутствии для них средств выборки данных по вторичному ключу.

2.2.6 Двоичное дерево поиска

2.2.6.1 Самобалансирующееся дерево поиска

2.2.6.1.1 АВЛ-дерево

2.2.6.1.1.1 Дерево Фибоначчи

2.2.6.1.2 Красно-чёрное дерево

2.2.6.1.3 Расширяющееся дерево

2.2.7 Куча

2.2.7.1 Двоичная куча

2.2.7.2 Биномиальная куча

2.2.7.3 Фибоначчиева куча

2.2.7.4 Сливаемая куча

2.2.8 Суффиксное дерево

2.2.8 Префиксное дерево

Существует еще следующее разделение структур:

Статические структуры данных

Полустатические структуры данных

Динамические структуры данных

Нелинейные структуры данных

общий вид описания структур:

-основное предназначение, описание

-поддерживаемые операции

-преимущества

-недостатки

-готовая реализация в языке программирования (название функции или класса)

условные обозначения