Структуры данных

- **Структура данных - организационная схема данных, в соответствии с которой они упорядочены, с тем, чтобы их можно было интерпретировать и выполнять над ними определенные операции.**
- **Структуры данных могут быть *однородными* и *неоднородными*.**
- **В *однородных* структурах все элементы данных представлены записями одного типа. В *неоднородных* - элементами одной структуры могут являться записи разных типов.**

- **Различные структуры данных предоставляют и различный доступ к своим элементам: в одних структурах доступ возможен *к любому элементу*, в других - *к строго определенному*.**
- **В памяти компьютера данные могут иметь *последовательное* или *связанное* представление. Соответственно различают структуры хранения, использующие *последовательное представление данных* и *связанное представление данных*.**

Последовательное представление:

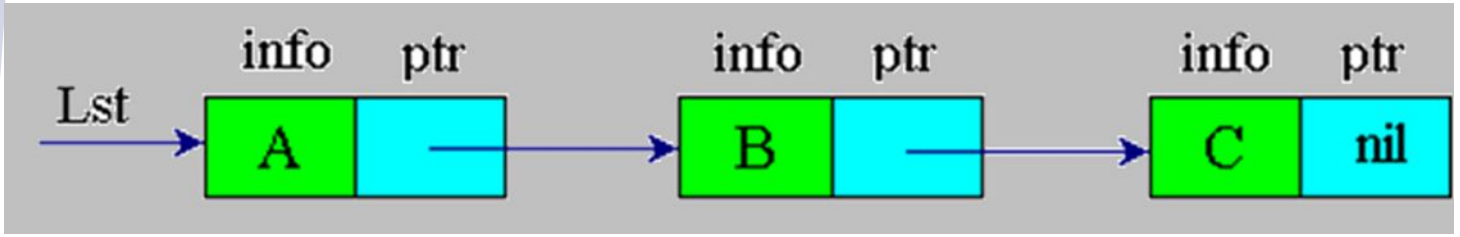
- данные в памяти компьютера размещаются в соседних последовательно расположенных ячейках
- физический порядок следования записей полностью соответствует логическому порядку, определяемому логической структурой
- совокупность записей, размещенных в последовательно расположенных ячейках памяти, называют *последовательным списком*.

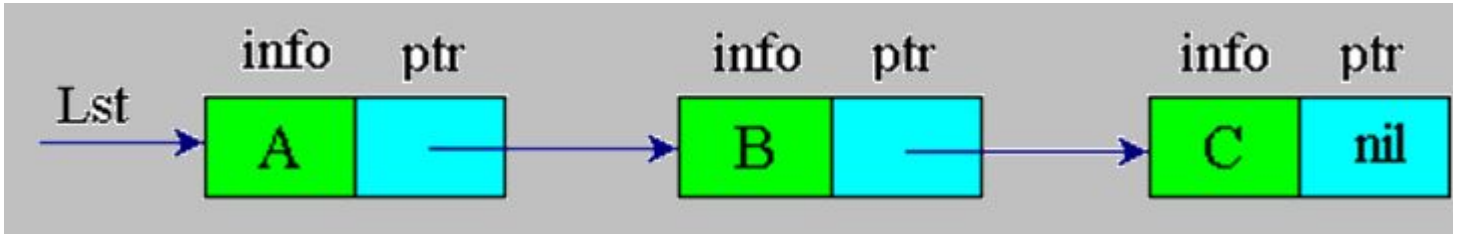
Связанное представление данных:

- записи располагаются в любых свободных ячейках и связываются указателями, указывающими на место расположения записи, логически следующей за данной.
- в каждой записи предусматривается дополнительное поле, в котором размещается указатель (ссылка).

- структуры хранения, основанные на связанном представлении данных, называют *связанными списками*.
- если каждая запись содержит лишь один указатель, то список *односвязный*, при большем числе указателей - список *многосвязный*

Пример односвязного списка:





- **Элемент односвязного списка содержит два поля: информационное поле (info) и поле указателя (ptr)**
- **Указатель дает только адрес последующего элемента списка**
- **Поле указателя последнего элемента в списке является пустым (NIL)**
- **LST - указатель на начало списка. Список может быть пустым, тогда LST = NIL**
- **Доступ к элементу списка осуществляется только от его начала, то есть обратной связи в этом списке нет**

Пример многосвязного списка:

- В двусвязном списке у любого элемента есть два указателя
- Один указывает на предыдущий (левый) элемент (L), другой указывает на последующий (правый) элемент (R)

Структуры данных делятся на *линейные* и *нелинейные*.

К линейным структурам относят:

- **массив**
- **стек**
- **очередь**
- **таблица**

В **нелинейных структурах** связь между элементами структуры (записями) определяется отношениями подчинения или какими-либо логическими условиями.

К нелинейным структурам относят:

- деревья
- графы
- многосвязные списки
- списковые структуры

Линейные (статические) структуры данных

Массив **- это линейная структура данных фиксированного размера, реализуемая с использованием последовательного представления данных.**

Каждый элемент массива идентифицируется одним или несколькими индексами.

Индекс **- это целое число, значение которого определяет позицию соответствующего элемента в массиве и используется для осуществления доступа к этому элементу.**

Стек - это линейная структура переменного размера.

В отличии от структуры массива позволяет включать или исключать элементы, то есть объем данных в стеке может динамически расти и сокращаться во время выполнения программы.

Информация обрабатывается по принципу "последним пришел, первым ушел".

Очередь - это линейная структура переменного размера.

Исключение элементов из очереди допускается с одного конца - с начала очереди.

Включение элементов возможно лишь в противоположный конец - в конец очереди.

Данные обрабатываются в порядке их поступления по принципу: "первым пришел, первым ушел".

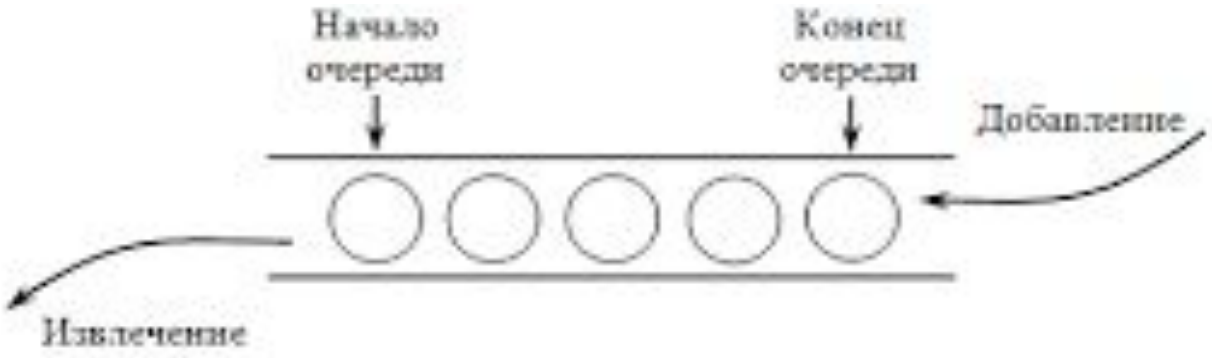
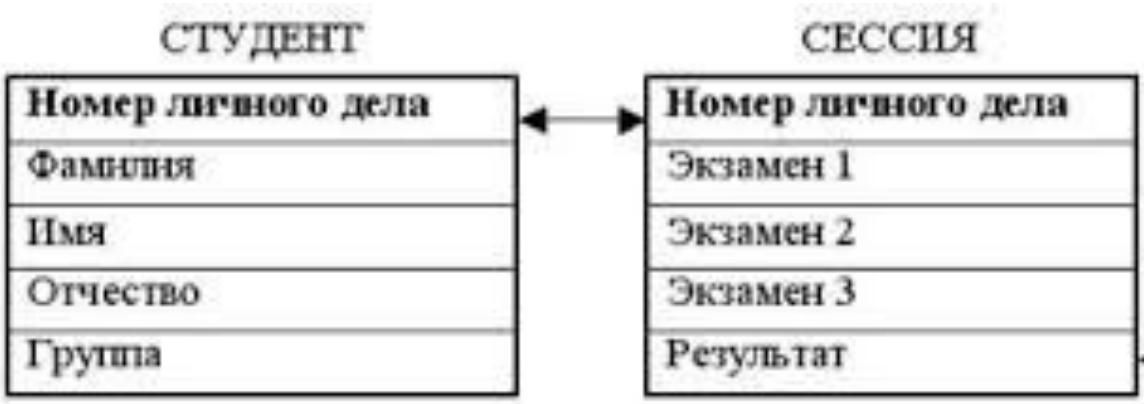


Таблица - это линейная структура данных, каждый элемент которой характеризуется определенным значением ключа и доступ к элементам которой осуществляется по ключу.



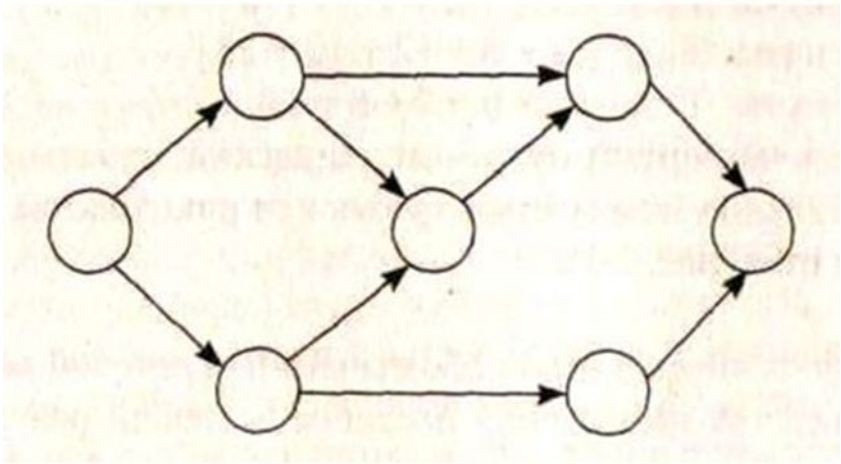
Нелинейные структуры данных

- Отношения между объектами реального мира часто носят нелинейный характер.
- Это могут быть отношения, определяемые логическими условиями, отношениями типа "один-ко-многим" или отношениями типа "многие-ко-многим".

Отношения "один-ко-многим" носят иерархический характер и отображаются *древовидными* структурами.

Пример: в виде дерева может быть представлена структура учебных подразделений ВУЗа.

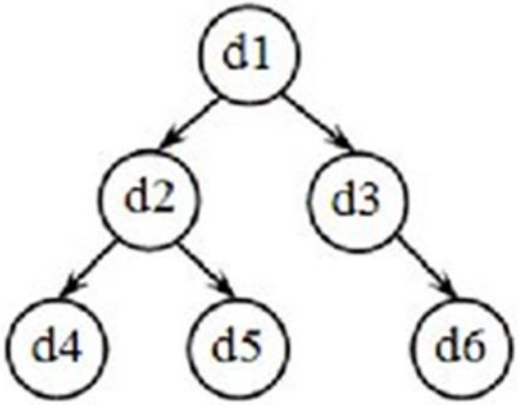
- **Граф общего вида состоит из ряда вершин (узлов) и ребер, связывающих пары вершин.**
- **Если в понятия "ребро" и "вершина" вложить определенную смысловую нагрузку, то графы можно использовать для представления данных.**



Дерево – это граф с некоторыми ограничениями, т.е. ориентированный граф, не имеющий циклов.

Вершины (узлы) дерева организованы по уровням и подчинены определенной иерархии.

Любой узел дерева связан с единственным узлом более высокого уровня - ***порождающим*** - и с *m* узлами ближайшего уровня - ***порожденными***.



На самом верхнем уровне имеется единственный узел, называемый **корнем**.

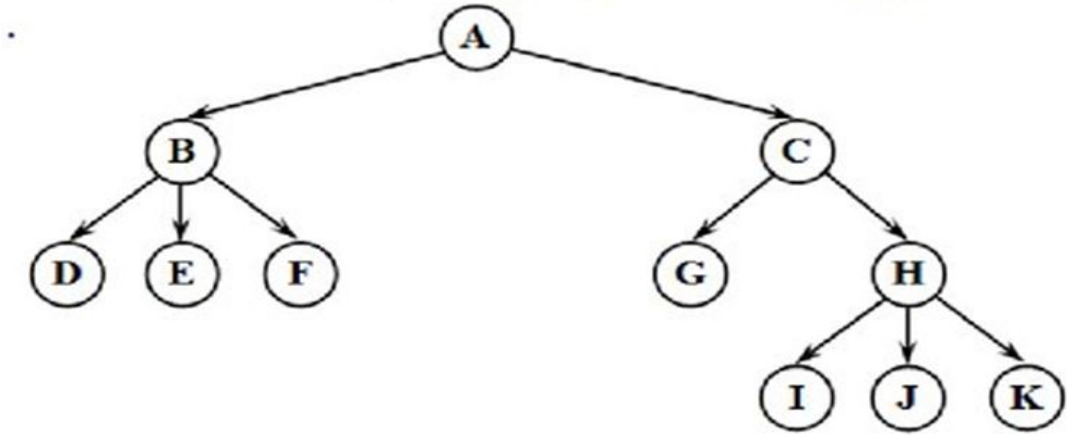
Узлы, расположенные в конце каждой ветви дерева и не имеющие порожденных, называются **листьями**.

В деревьях направление обязательно от порождающего к порожденному.

Длина пути от корня до некоторого узла равна **уровню** этого узла.

Количество уровней дерева определяет **высоту дерева**.

Бинарное (двоичное) дерево **– это динамическая структура данных, представляющая собой дерево, в котором каждая вершина имеет не более двух потомков.**



Уровни представления данных

На *логическом уровне* оперируют с логическими структурами данных, отражающими реальные отношения, которые существуют между объектами (таблицами) БД и их характеристиками.

Единицей хранения на этом уровне является логическая запись.

На логическом уровне представления данных не учитывается техническое и математическое обеспечение системы.

На уровне хранения оперируют со структурами хранения, то есть представлениями логической структуры данных в памяти ПК.

Единицей хранения информации на этом уровне также является логическая запись.

Одна и та же логическая структура данных может быть реализована в памяти компьютера различными структурами хранения (например, массив, запись, таблица).

На физическом уровне представления данных оперируют с физическими структурами данных.

На этом уровне решается задача реализации структуры хранения в памяти компьютера.

Единицей хранения информации является физическая запись, представляющая собой участок носителя, на которой размещается одна или несколько логических записей.

Физическая независимость от данных означает, что любые изменения в физическом расположении данных или техническом обеспечении БД не должны отражаться на логических структурах и прикладных программах, то есть не должны вызывать в них изменений.

Логическая независимость от данных означает, что изменения в структурах хранения не должны вызывать изменения в логических структурах данных и в прикладных программах.

Виртуальные данные существуют лишь на логическом уровне.

Пользователю эти данные представляются реально существующими, и он оперирует ими при необходимости.

Каждый раз при обращении к этим данным система определенным образом их генерирует на основании других данных, физически существующих в системе.

Прозрачные данные представляются несуществующими на логическом уровне.

Это позволяет скрыть от пользователя многие сложные механизмы, используемые при преобразовании логических структур данных в физические (примером могут служить индексы).