

The background is an abstract composition. On the left, there is a close-up, slightly blurred image of a computer keyboard, showing keys with Cyrillic characters. This transitions into a large, smooth, light-colored curved shape that dominates the center and right. In the top right corner, there is a small, circular inset showing a blue and white circuit board pattern. The overall lighting is soft and diffused.

# Структуры и алгоритмы обработки данных

## *Лекция 2*

### Классификация структур данных

# Схема процесса создания программ для решения прикладных задач



Математическая модель

- Неформальный алгоритм

Абстрактное описание данных

- Формальный алгоритм

Описание данных на языке программирования

- Программа на языке программирования

- ❖ Создание программы можно рассматривать как процесс последовательного преобразования информации от первоначальной неформальной постановки задачи, до получения завершенной программы на языке программирования
- ❖ В процессе преобразования информации от постановки задачи до получения компьютерной программы выделяются два взаимосвязанных объекта – данные и алгоритм их преобразования



# Понятие структуры данных



**ЭВМ в настоящее время:**

- считывает и выполняет определенные алгоритмы
- хранит значительные объемы информации, к которой нужно быстро обращаться

**Эта информация** - абстракция фрагмента реального мира, состоит из определенного множества данных, относящихся к какой-либо проблеме

# Понятие структуры данных



## Любые данные в памяти ЭВМ :

- представляются последовательностью двоичных разрядов, или битов
- их значениями являются соответствующие двоичные числа
- имеют очень простую организацию - слабо структурированы

*Для человека - описывать и исследовать сколько-нибудь сложные данные в терминах последовательностей битов весьма неудобно*

*Более крупные и содержательные, чем бит, «строительные блоки» для организации произвольных данных получаются на основе понятия структуры данного*

# Понятие структуры данных



**Данные** – это значения или наборы значений, описывающие любую информацию, которую можно обработать на компьютере

**Данные** могут иметь разный уровень сложности или организованности (от самых простых - числа или символы, заканчивая достаточно сложными образованиями, включающими как **сами элементы данных**, так и **информацию об их количестве и взаимосвязях между этими элементами**)

Модель организованности данных принято называть **структурой данных**

**Структура данных** в общем случае –  
множество элементов данных и множество связей между ними

# Понятие структуры данных



## Пример

*Т.к. данные – это некоторые сообщения, слова в некотором заданном алфавите, то*

- ❖ *число **123** – данное, представляющее собой слово в алфавите из десяти натуральных цифр;*
- ❖ *число **12,34** – данное, представляющее собой слово в алфавите из десяти натуральных цифр и десятичной запятой;*
- ❖ *текст "**математика и информатика – нужные дисциплины**", – данное в алфавите из символов русского языка и знаков препинания, включая пробел*

# Понятие структуры данных



## УРОВНИ ПРЕДСТАВЛЕНИЯ СТРУКТУР ДАННЫХ:

- ◆ абстрактный (математический) уровень
- ◆ логический уровень
- ◆ физический уровень

# Понятие структуры данных



## УРОВНИ ПРЕДСТАВЛЕНИЯ СТРУКТУР ДАННЫХ:

- ❖ **абстрактный (математический) уровень** - определяет характер организованности структуры данных

**Организованность** - описывается математической моделью, может быть представлена множеством **элементов**, каждый из которых является **значениями данных**, и **отношениями** между ними, свойства которых определяют различные типы данных

**Тип данных** – это математическая модель организованности данных и различные операторы, определенные в рамках этой модели

# Понятие структуры данных



## УРОВНИ ПРЕДСТАВЛЕНИЯ СТРУКТУР ДАННЫХ:

- ❖ **Логический уровень –**  
*представление структуры данных на языке программирования*

*Простейшие структуры, представляющие собой один элемент, определяются простыми типами*

*В языках программирования для переменных простого типа уже определены множества допустимых значений и набор допустимых операций*

*Определенные в языке программирования типы данных называют базовыми.* Их характер организованности – простейший

# Понятие структуры данных



## УРОВНИ ПРЕДСТАВЛЕНИЯ СТРУКТУР ДАННЫХ:

### ❖ Логический уровень –

*представление структуры данных на языке программирования*

*Составные структуры данных - совокупность простых структур и отношения между ними, могут быть определены программистом в виде структурных типов (например, массив)*

*Набор допустимых значений зависит от простых типов, на основе которых построена структура данных и ее типа*

# Понятие структуры данных



## УРОВНИ ПРЕДСТАВЛЕНИЯ СТРУКТУР ДАННЫХ:

### ❖ Логический уровень –

*представление структуры данных на языке программирования*

*Для других структур данных, например **список, стек, очередь, дерево, таблица и др.**, нет соответствующих типов, определяющих организованность этих данных и допустимые операции*

*Такие **структуры** данных называются **производными** и реализуются непосредственно программистами*

# Понятие структуры данных



## УРОВНИ ПРЕДСТАВЛЕНИЯ СТРУКТУР ДАННЫХ:

- ❖ **Физический уровень –**  
отображение на память ЭВМ информационного объекта в соответствии с логическим описанием

*На этом уровне определяются*

- область и объём памяти, необходимый для хранения экземпляра структуры данных,
- форматы и интерпретация внутреннего представления

**Физическая структура данных** - способ физического представления данных в памяти машины и называется еще **структурой хранения**, **внутренней структурой** или **структурой памяти**

# Классификация структур данных

## Данные

### Данные статической структуры

- ▶ *могут быть простыми и составными*
- ▶ *формируются из простых структур по определенному закону*
- ▶ *взаимное расположение и взаимосвязь элементов структуры всегда остаются постоянными*

### Данные динамической структуры

- ▶ *внутреннее строение данных формируется по определенному закону*
- ▶ *количество элементов, их взаиморасположение и взаимосвязи могут динамически изменяться во время выполнения программы*

# Классификация структур данных

## Типы данных

### Типы данных линейной структуры

► *определяют список элементов, упорядоченных по положению*

- ◆ *тип данных с прямым доступом*
- ◆ *тип данных с индексным доступом*
- ◆ *тип данных с последовательным доступом*

### Типы данных нелинейной структуры

► *определяют элементы без позиционного упорядочивания*

- ◆ *могут иметь иерархическую или групповую структуру*

# Классификация структур данных

## Типы данных линейной структуры

### Тип данных с прямым доступом

► позволяет выбирать элемент непосредственно, не обращаясь сначала к предшествующим элементам в списке

### Тип данных с индексным доступом

► с записью данных связывается некоторый ключ, использующийся для доступа к записи

### Типы данных с последовательным доступом

► *Динамические структуры со свойствами*

◆ *непостоянство и непредсказуемость размера;*

◆ *отсутствие физической смежности элементов структуры в памяти*

# Классификация структур данных



## Типы данных линейной структуры

### Пример

*Зададим простые типы данных "специальность", "студент", "вуз" следующим перечислением:*

- ❖ *специальность = (филолог, историк, математик, медик);*
- ❖ *студент = (Петров, Николаев, Семенов, Иванова, Петрова);*
- ❖ *вуз = (КФУ, МГУ, СевНТУ)*

*Значением типа «студент» может быть Петров*

# Классификация структур данных



## Структурированный тип данных

### Пример

*Зададим простые типы данных "специальность", "студент", "вуз" следующим перечислением:*

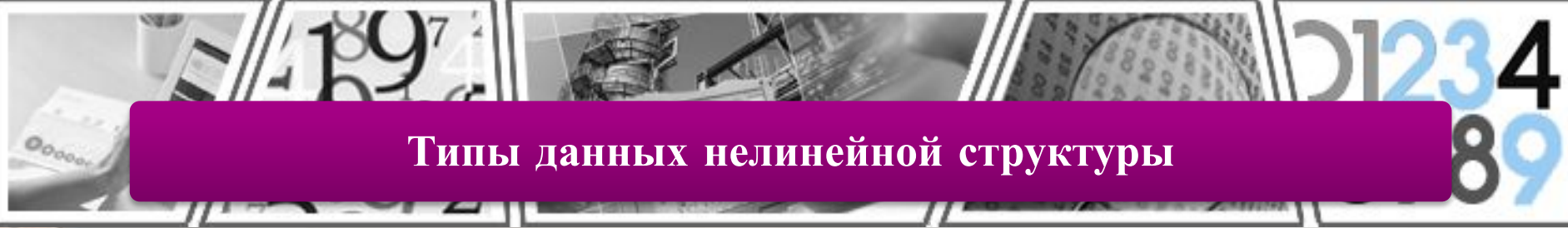
- ❖ *специальность = (филолог, историк, математик, медик);*
- ❖ *студент = (Петров, Николаев, Семенов, Иванова, Петрова);*
- ❖ *вуз = (ТНУ, МГУ, СевНТУ)*

*Структурированный тип данных "специальность\_студента":*

- ❖ *специальность\_студента = (специальность, студент)*

*Значением типа "специальность\_студента" может быть пара (историк, Семенов)*

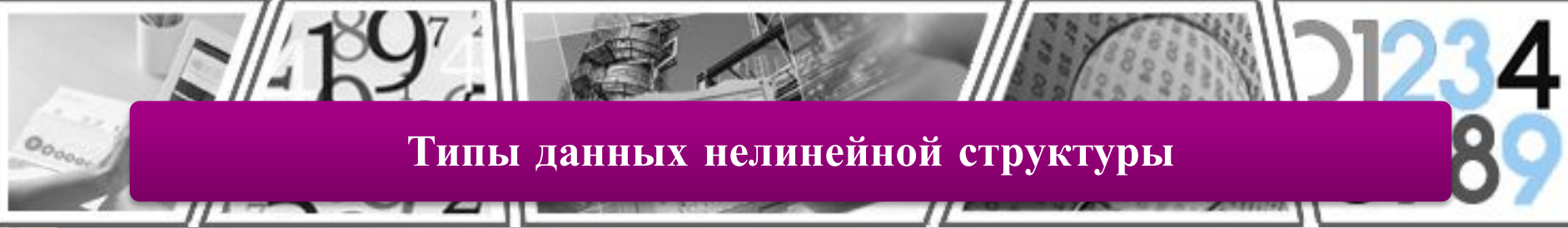
# Классификация структур данных



## Типы данных нелинейной структуры

- *могут иметь иерархическую или групповую структуру*
  - ❖ *Иерархическая структура* – это совокупность элементов, которые разделяются по уровням, при этом элементы на данном уровне структуры могут иметь несколько наследников на следующем уровне
  - ❖ *Групповая структура* представляет собой нелинейную структуру, которая содержит элементы без какого-либо упорядочения

# Классификация структур данных



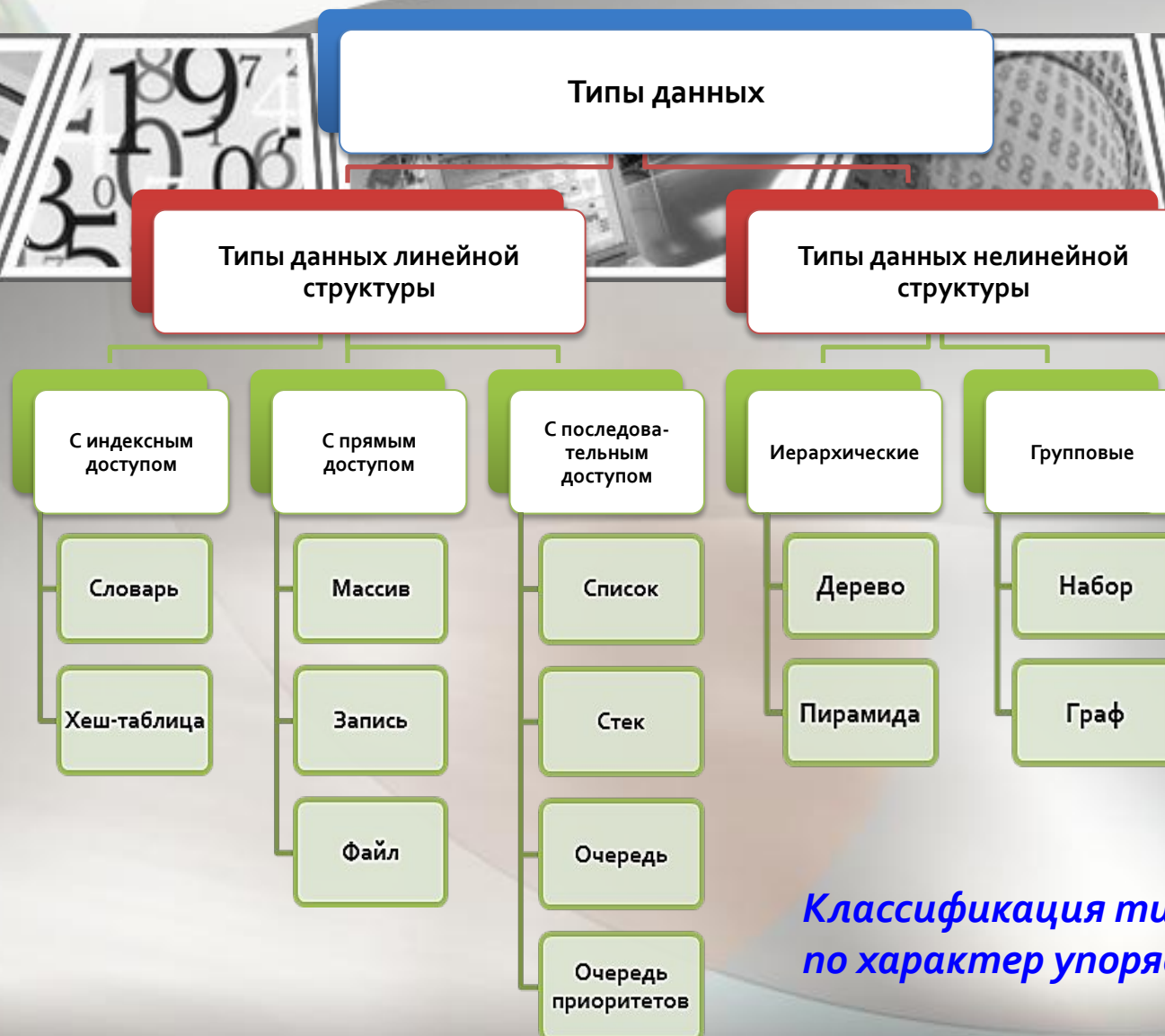
## Типы данных нелинейной структуры

### Пример

*Структуру "вуз" можно задать иерархической структурой, состоящей, например, из следующих уровней:*

- ❖ ректорат
- ❖ деканаты и подразделения кафедры
- ❖ отделы
- ❖ преподаватели и сотрудники, студенты

# Классификация структур данных



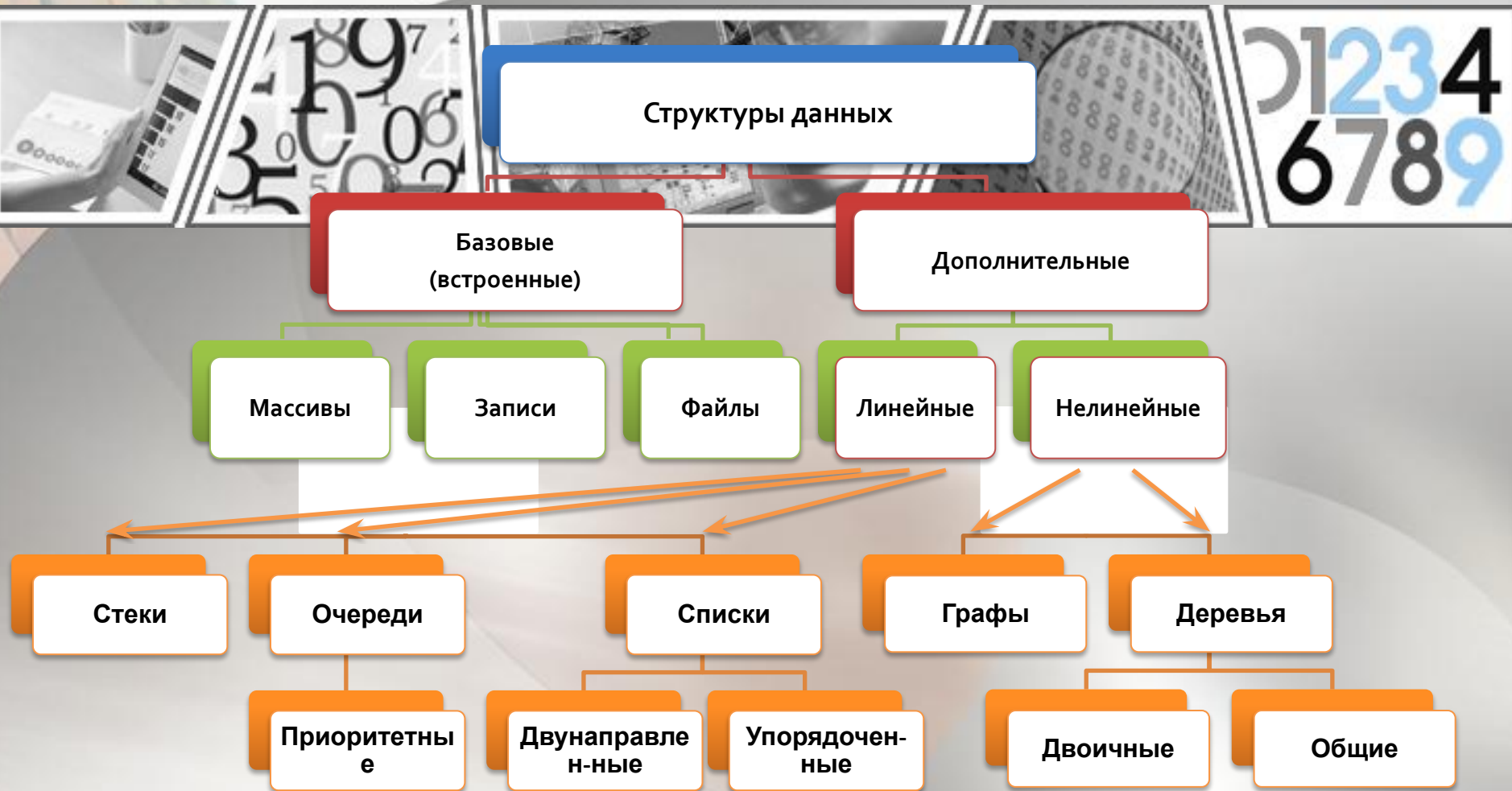
*Классификация типов данных  
по характер упорядоченности*

# Классификация структур данных



*Классификация структур данных в зависимости от размещения физических структур и доступа к ним*

# Классификация структур данных



*Классификация базовых и дополнительных структур данных*

# Структуры данных



## ❖ Массив –

- *линейная структура данных*
- *состоит из конечного, фиксированного и упорядоченного набора элементов, имеющих один и тот же тип*
- *с прямым доступом посредством целого индекса – номера элемента в последовательности*

*Для доступа к элементу на логическом уровне достаточно указать имя массива и индекс элемента*

# Структуры данных

| 0 | 1 | 2  | 3  | 4  | 5 | 6 | 7  | 8  | 9 |
|---|---|----|----|----|---|---|----|----|---|
| 7 | 3 | -5 | 12 | 34 | 2 | 5 | 56 | -7 | 4 |

a)

|   | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
|---|----|----|----|----|----|----|----|----|----|----|
| 0 | 7  | 3  | 5  | 12 | 34 | 2  | 5  | 56 | 13 | 4  |
| 1 | 0  | -8 | -4 | 10 | 54 | 33 | -4 | 98 | 12 | 12 |
| 2 | 34 | 12 | -6 | 6  | 94 | 52 | 1  | 0  | 10 | 10 |
| 3 | 2  | 3  | 4  | 7  | 4  | 55 | 12 | 12 | 34 | 77 |
| 4 | 34 | 12 | -6 | 0  | -6 | 1  | 1  | 0  | 54 | 3  |
| 5 | 14 | 3  | -7 | -4 | 88 | -3 | 12 | 67 | 24 | -2 |
| 6 | 3  | -2 | 34 | -5 | 6  | -5 | 3  | 30 | 8  | 11 |
| 7 | 44 | -5 | 14 | 28 | 13 | -6 | 53 | 23 | 71 | 5  |
| 8 | 12 | 3  | 39 | 68 | 38 | 0  | 1  | 58 | 0  | 22 |
| 9 | 33 | 23 | -7 | 21 | 5  | 2  | 67 | 34 | 8  | 9  |

b)

|   | 2  | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
|---|----|----|----|----|----|----|----|----|----|----|----|
| 1 |    | 2  | -4 | 22 | -4 | 17 | -4 | 61 | 67 | 12 | 34 |
| 0 |    | 7  | 3  | 5  | 12 | 34 | 2  | 5  | 56 | 13 | 4  |
|   | 33 | -2 | 7  | -5 | 23 | 54 | 33 | -4 | 98 | 12 | 12 |
|   | 2  | 3  | 4  | 7  | 4  | 94 | 52 | 1  | 0  | 10 | 10 |
|   | 34 | 12 | -6 | 6  | 94 | 4  | 55 | 12 | 12 | 34 | 77 |
|   | 2  | 3  | 4  | 7  | 4  | 55 | 6  | 77 | 1  | 0  | 54 |
|   | 34 | 12 | -6 | 0  | -6 | 77 | 1  | 0  | 1  | 1  | 1  |
|   | 14 | 3  | -7 | -4 | 88 | 35 | 12 | 67 | -3 | 12 |    |
|   | 3  | -2 | 34 | -5 | 6  | 14 | 3  | 30 | -5 | 3  |    |
|   | 44 | -5 | 14 | -6 | 13 | 4  | 53 | 23 | -6 | 53 |    |
|   | 12 | 3  | 39 | -0 | 38 | 19 | 1  | 58 | 0  | 1  |    |
|   | 33 | 23 | -7 | 21 | 5  | 2  | 67 | 34 | 2  | 67 |    |

c)

| 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
|----|----|----|----|----|----|----|----|----|----|
| 33 | 23 | -7 | 21 | 5  | 2  | 67 | 34 | 8  | 9  |
| 45 | 3  | 38 | 88 | 38 | 0  | 1  | 58 | 0  | 22 |
| 44 | -5 | 14 | 28 | 13 | -6 | 53 | 23 | 71 | 5  |

d)

| 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
|----|----|----|----|----|----|----|----|----|----|
| 33 | 23 | -7 | 21 | 5  | 2  | 67 | 34 | 8  | 9  |
| 45 | 3  | 38 | 88 | 38 | 0  | 1  | 58 | 0  | 22 |
| 44 | -5 | 14 | 28 | 13 | -6 | 53 | 23 | 71 | 5  |

e)

❖ Массив –

- линейная структура данных

# Структуры данных



## ❖ Массив –

*Пример.* Последовательность чисел 89, –65, 9, 0, –1.7 может образовывать одномерный вещественный массив размерности 5, например, с именем  $x$  вида:  $x[1] = 89$ ,  $x[2] = -65$ ,  $x[3] = 9$ ,  $x[4] = 0$ ,  $x[5] = -1.7$ .

*Значение порядкового номера элемента массива называется индексом элемента.*

*Можно ссылаться на элемент  $x[4]$ , элемент  $x[i]$ , элемент  $x[4+j]$  массива  $x$ . При текущих значениях переменных  $i = 2$  и  $j = 1$  эти индексы определяют, соответственно, 4-й, 2-й и 5-й элементы массива*

# Структуры данных



## ❖ *Запись (структура) –*

- *тип данных линейной структуры*
- *содержит конечный и фиксированный набор элементов (полей), возможно, имеющих различные типы*
- *с прямым доступом к элементу посредством имени поля*

*Для доступа к полю структуры на логическом уровне достаточно указать имя структурной переменной и имя ее поля.*

# Структуры данных

День Победы:

Полёт Гагарина:

| День | Месяц  | Год  |
|------|--------|------|
| 9    | май    | 1945 |
| 12   | апрель | 1961 |

- ❖ **Записи (структуры)** – структуры, аналогичные строкам таблицы
- ❖ Компоненты записей принято называть **полями**
- ❖ Различные поля (столбцы таблицы) могут быть разных типов
- ❖ Для доступа к отдельным полям записи используются их фиксированные и неизменные имена
  - Например: **День Победы. Месяц := май**
- ❖ Поля могут выбираться для обработки в произвольном порядке, поэтому говорят, что **доступ к компонентам записи прямой**

# Структуры данных



*Традиционный пример структуры – строка платежной ведомости:*

- ❖ *содержит сведения о служащем:*
  - *полное имя*
  - *адрес*
  - *номер карточки социального страхования*
  - *зарплата и т.д.*
- ❖ *некоторые из этих характеристик сами могут быть структурами:*
  - *полное имя состоит из нескольких компонент (фамилии, имени и отчества)*
  - *адрес*
  - *зарплата*

# Структуры данных



## ❖ Словарь (таблица) –

- тип данных линейной структуры с индексным доступом
- состоит из элементов вида "ключ – значение", называемых ассоциациями  
Например, ключом может быть слово, а значением – строка, указывающая определение слова
- по значению в ассоциации осуществляется прямой доступ с использованием ключа в качестве индекса

В результате словарь подобен массиву, за исключением того, что индексы не должны быть целыми значениями

# Структуры данных



## ❖ Хеш-таблица —

- тип данных линейной структуры с индексным доступом
- предназначенный для хранения данных, связанных с ключом

*Позволяет хранить пары (ключ, значение) и выполнять три операции:*

- операцию добавления новой пары
- операцию поиска
- операцию удаления пары по ключу

# Структуры данных

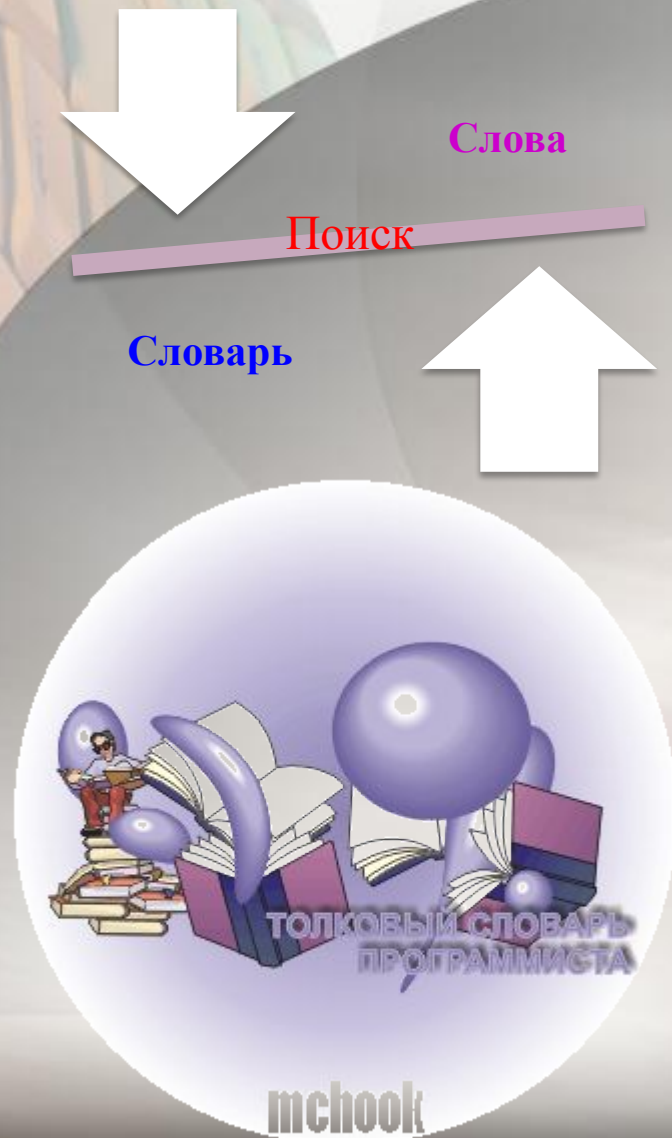


**Хеш-таблица** представляет собой обобщение обычного массива

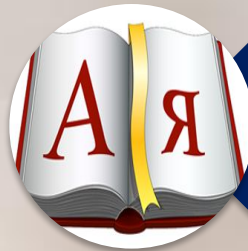
|             |                                                              |
|-------------|--------------------------------------------------------------|
| Массив      | Ключ – только число                                          |
| Хеш-таблица | Ключ - любой объект, для которого<br>можно вычислить хеш-код |

# Структуры данных

## Хеширование. Словарь



Словарь будем представлять в виде массива слов



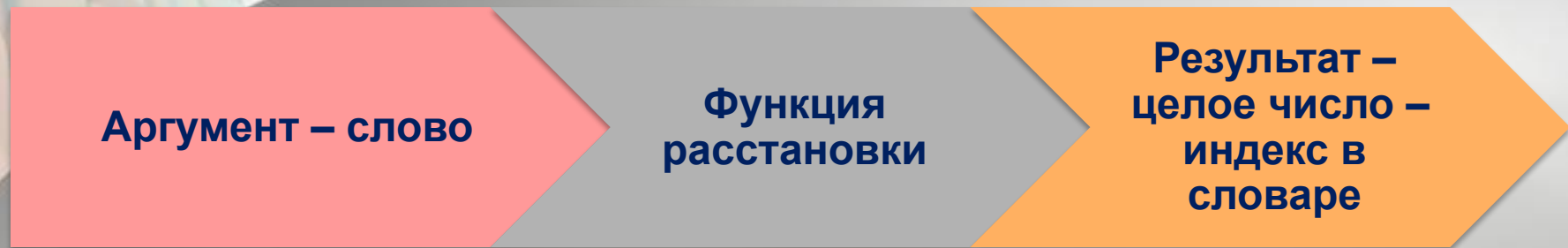
Помещать слова будем не по алфавиту



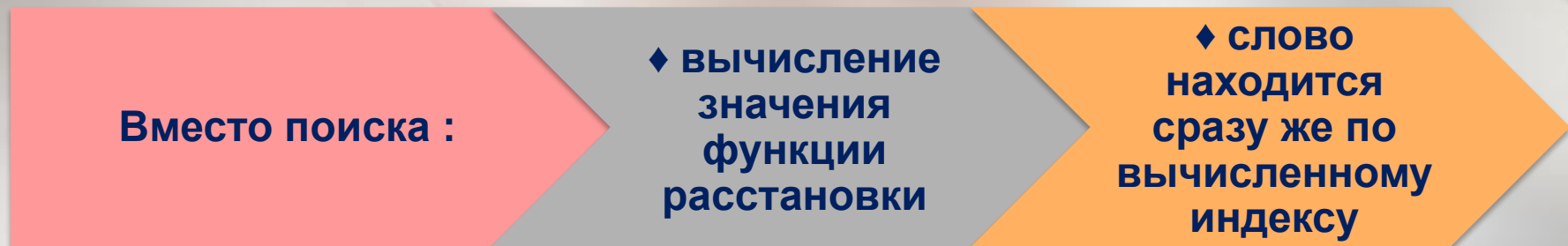
Размещаем в соответствии со значениями простой функции, вычисленной над словом

# Структуры данных

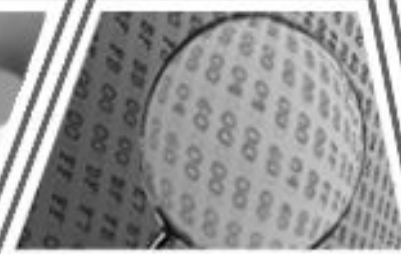
## Хеширование. Словарь



Если каждому слову будет соответствовать свое значение функции, то поиск в словаре становится ненужным



# Структуры данных



## ❖ **Стек**

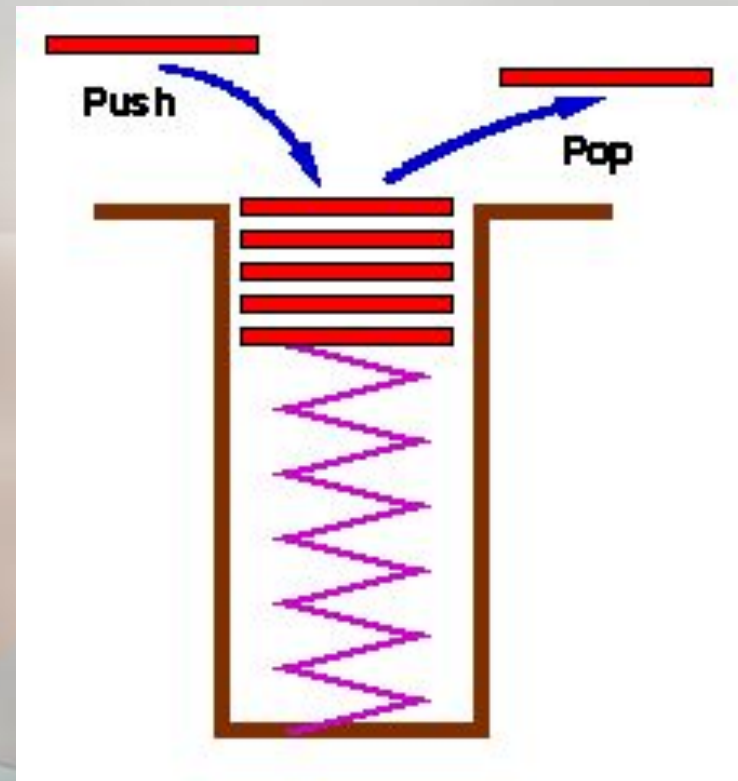
– это упорядоченный набор элементов, в котором добавление новых и удаление существующих производится с одного конца, называемого **вершиной стека**

*Стеки иногда называют магазинами*  
- по аналогии с магазином в огнестрельном оружии (стрельба начнётся с патрона, заряженного последним)

Для обозначения стеков часто используется аббревиатура

**LIFO – last in, first out**

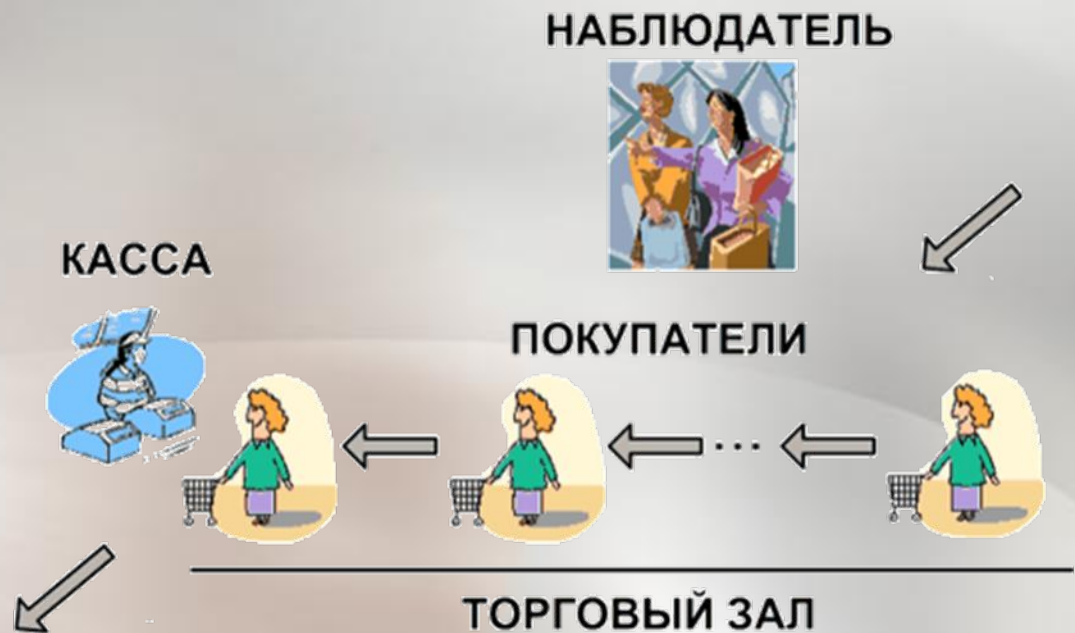
*«Последним пришел, первым ушел»*



# Структуры данных

**Очередь** – это последовательный список переменной длины, в котором включение элементов производится с одного конца списка (с хвоста), а исключение элементов производится с другого конца (из головы)

Данное определение можно сравнить с обычной очередью в магазине — если покупатель пришел первым, то он и обслуживается первым



Для обозначения очереди часто используется аббревиатура

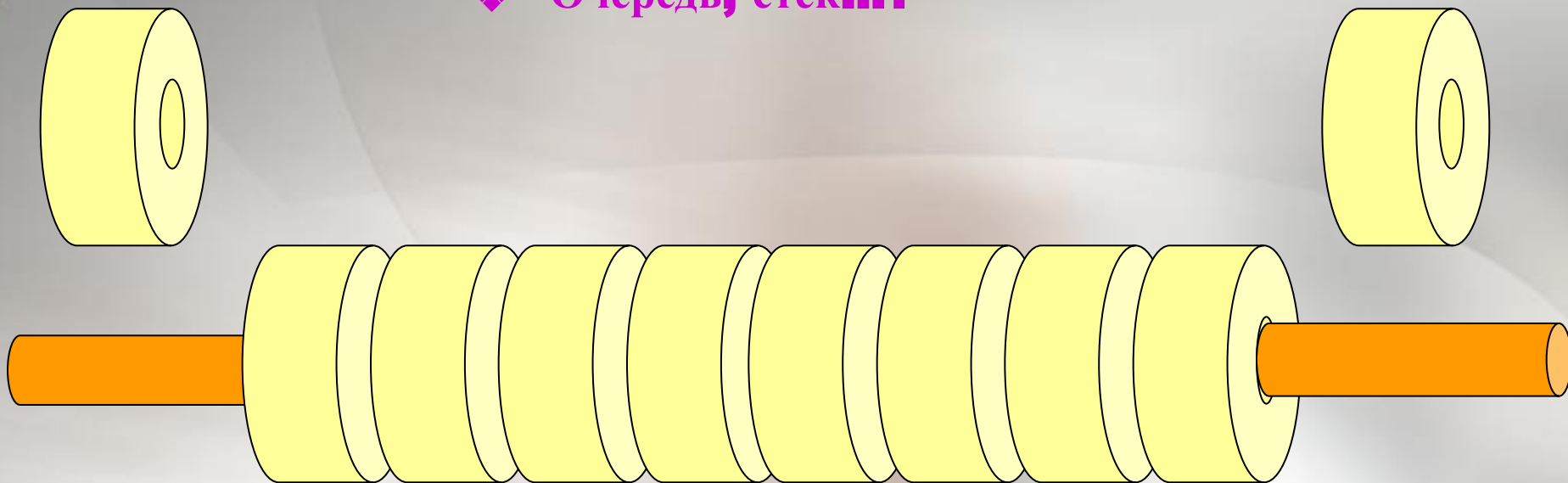
**FIFO – First in First out**

«Первый зашел, первый вышел»

# Структуры данных

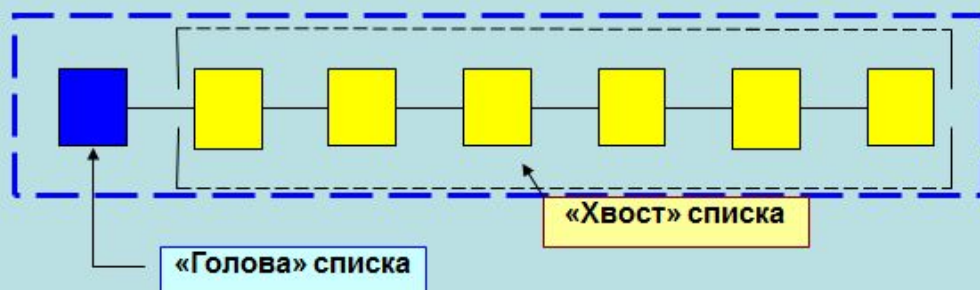


◆ Очередь, стек...?

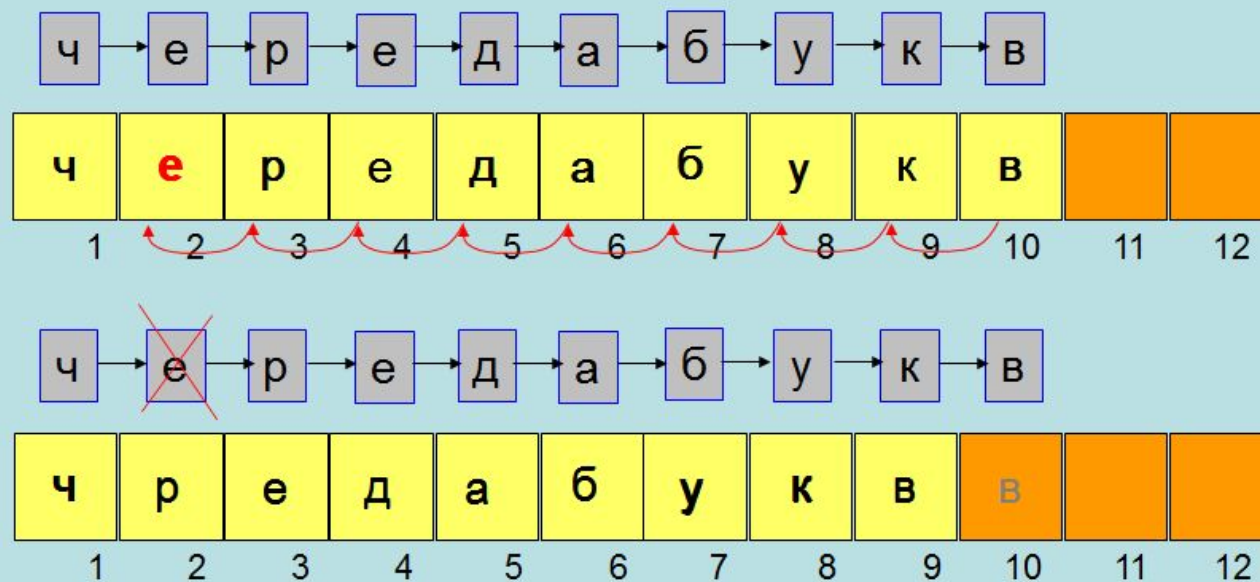


# Структуры данных

## Модельное представление линейного списка



Пример операции «удалить элемент списка»:



# Структуры данных



## ◆ Файл –

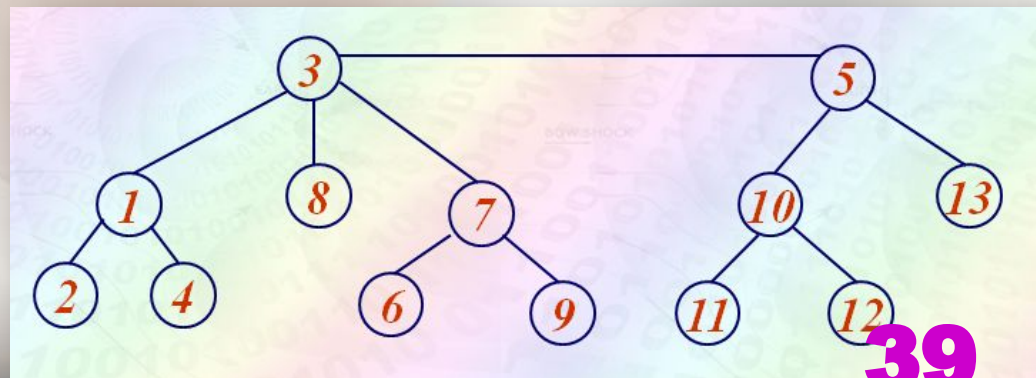
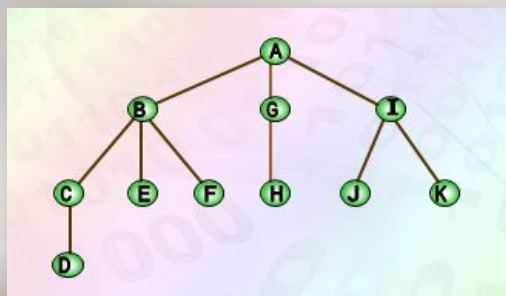
- *тип данных линейной структуры с прямым или последовательным доступом*
- *представляет собой последовательность байтов, приравниваемую к потоку (последовательность байтов, перемещаемая от одного устройства к другому)*
- *прямой доступ осуществляется только к дисковому файлу*

# Структуры данных



## ◆ Дерево –

- *нелинейная иерархическая структура данных*
- *все элементы происходят от одного источника, называемого корнем*
- *каждый элемент, за исключением корня, имеет единственного предка*



# Структуры данных



## ◆ Набор (множество) –

- *нелинейная групповая структура данных*
- *находит применение, когда данные являются неупорядоченными, и каждый элемент данных является единственным в своем роде, уникальным*

# Структуры данных



## ❖ Граф –

- нелинейная групповая структура данных
- задает набор вершин и набор связей, соединяющих вершины

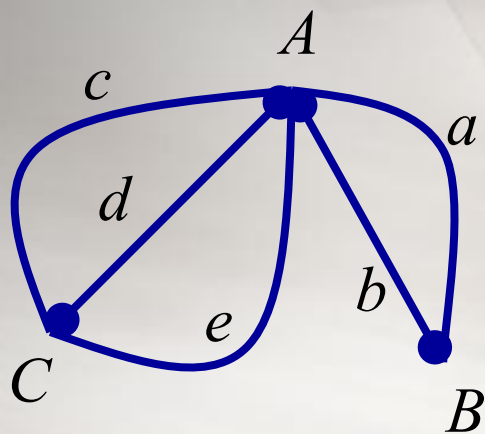
*Важную роль при обработке данных играют следующие операции:*

- обход структуры: доступ к каждому элементу структуры с целью его последующей обработки;
- поиск: нахождение места расположения элемента с данным значением;
- вставка: включение нового элемента в структуру;
- удаление: исключение элемента из структуры.

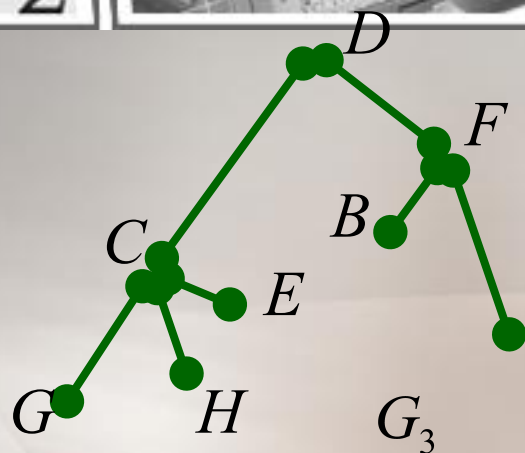
# Структуры данных



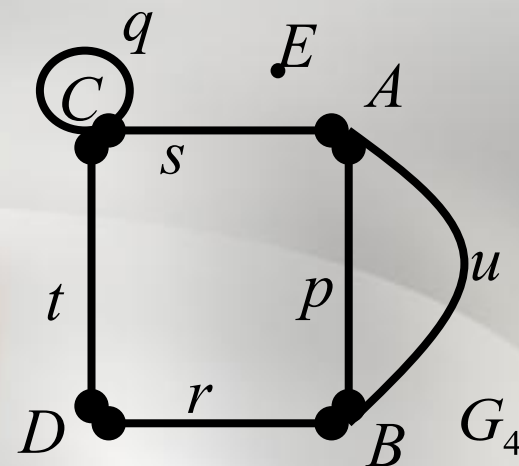
◆ Граф  
Примеры



$G_1$



$G_2$

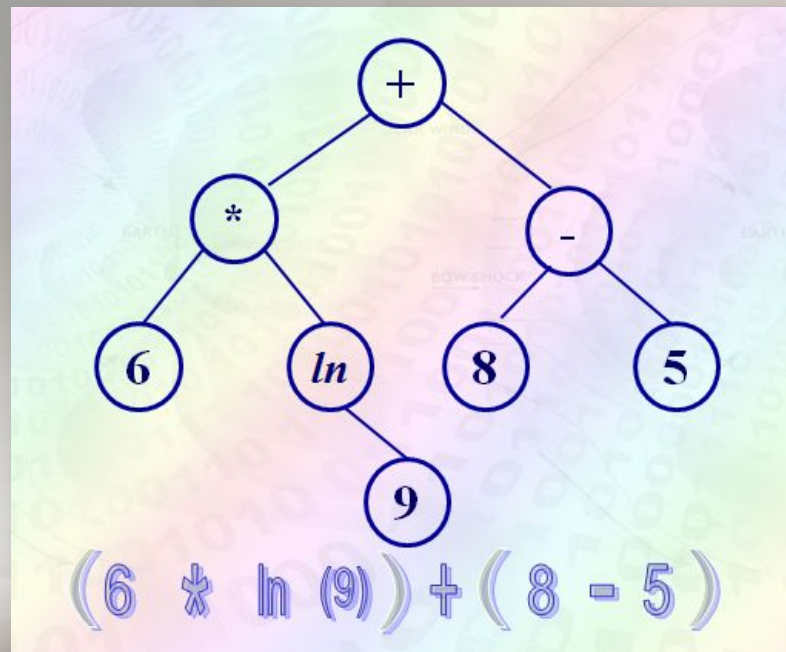


# Структуры данных

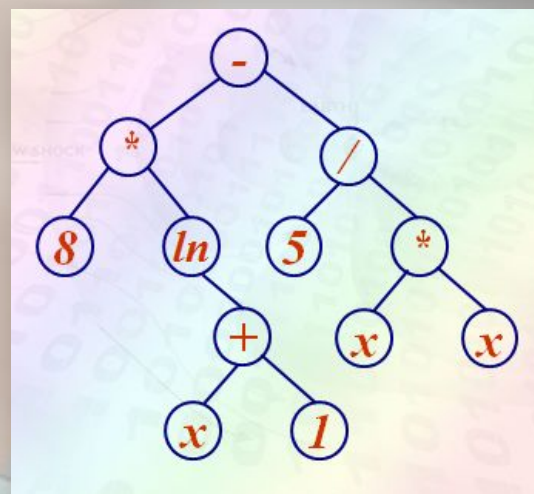


## ◆ Граф

Запись математических выражений



$$y = 8 * \ln(x + 1) - 5 / x^2$$



# Типы и структуры данных



*В языках программирования понятие «структуры данных» тесно связано с понятием «типы данных»*

*Любые данные, т.е. константы, переменные, значения функций или выражения, характеризуются своими типами*

*Информация по каждому типу однозначно определяет:*

- структуру хранения данных указанного типа, т. е. выделение памяти, представление данных в ней и метод доступа к данным;
- множество допустимых значений, которые может иметь тот или иной объект описываемого типа;
- набор допустимых операций, которые применимы к объекту описываемого типа

