КОМПЬЮТЕРНЫЕ МЕТОДЫ ОБРАБОТКИ ИНФОРМАЦИИ

Лекция 02 (С)

Методы компьютерного представления информации. Цифровое изображение и компьютерное зрение.

Содержание лекции

1. ПО и программирование. Парадигмы программирования. Роль способа представления данных.
2. Методы компьютерного представления и передачи информации.
3. Цифровое изображение. Сжатие видеоинформации.
4. Компьютерное зрение. Области приложения и примеры систем.
5. Возможные темы курсовых работ.

Программное обеспечение

Программное обеспечение принято по назначению подразделять на системноеПрограммное обеспечение принято по назначению подразделять на системное, прикладноеПрограммное обеспечение принято по назначению подразделять на системное, прикладное и инструментальное.

BIOS

Системное ПО

Операционная система

- Общего назначения
- Реального времени
- Сетевая
- Встраиваемая

Загрузчик операционной системы

Драйвер устройства

Инструментальное ПО

Средство разработки программного обеспечения

- Среда разработки
- RAD
- SDK

Система управления базами данных (СУБД)

- РеляционнаяРеляционная (DB2Реляционная (DB2, MySQLРеляционная (DB2, MySQL, Oracle, и т.д.)
- Объектно-ориентированнаяОбъектно-

Программное обеспечение

Прикладное ПО

Офисное приложение

Текстовый редактор
Текстовый процессор

- Табличный процессор
- Редактор презентаций

Корпоративная информационная система

- Аудиторская программа
- Бухгалтерская программа
- Система MRP
- Система MRP II
- Система ERP
- Система CRM
- Система POS

Система управления версиями (VCS)
Система управления проектами (Project Management)
Система автоматизации документооборота (EDM)
Финансово-аналитическая система
Система управления архивами документов (DWM)
Корпоративный портал

Программное обеспечение

Прикладное ПО

Система проектирования и производства

- Система автоматизации проектных работ (САПР, CAD)
- CAE-система
- CAM-система
- PDM-система

PLM-система
АСУТП (SCADA)
АСТПП (MES)

Система логистической поддержки изделий

Система анализа логистической поддержки (LSA)

- Система создания ИЭТР (IETM)

Система обработки и хранения медицинской информации

Система передачи, обработки, хранения и архивации изображений Радиологическая информационная сеть (РИС)

Госпитальная информационная сеть (ГИС)

Программное обеспечение

Прикладное ПО

Научное ПО

- Система математического и статистического расчёта и анализа
- Система компьютерного моделирования

Информационные системы

Геоинформационная система (ГИС)
Система поддержки принятия решений (СППР)

Система управления IT-инфраструктурой
Справочно-правовая система (СПС)

Клиент для доступа к интернет-сервисам

- Электронная почта
- Веб-браузер
- Система мгновенного обмена сообщениями
- IRC

IP-телефония
Пиринговая сеть
Потоковое мультимедиа
Банк-клиент

Программное обеспечение

Прикладное ПО

Мультимедиа

- Компьютерная игра
- Музыкальный редактор
- Графические программы

Видеоредактор
Аудиоредактор
Медиаплеер

Программные средства защиты

- Криптошлюз
- Средство аутентификации
- Средство мониторинга и аудита
- Сканер защищённости
- Средство разграничения доступа

- Система криптографической защиты, шифрования и ЭЦП
- Антивирусная программа
- Антиспамовая программа
- Межсетевой экран

Парадигмы программирования

Агентно-ориентированная
Компонентно-ориентированная
Конкатенативная
Декларативная
Декларативная (контрастирует с Императивной)
- Ограничениями
- Функциональная
- Потоком данных
  - Таблично-ориентированная (электронные таблицы)
  - Реактивная
- Логическая

Событийно-ориентированная
- Сервис-ориентированная

Комбинаторная
Императивная
Императивная (контрастирует с Декларативной)
- Процедурная

Парадигмы программирования

- Предметно-ориентированная
- Метапрограммирование
  - Автоматизация процесса программирования
  - Обобщённое программирование
  - Рефлексивно-ориентированная
  - Итерационная
- Параллельная
- Структурная
  - Модульная
  - Рекурсивная
  - Объектно-ориентированная
    - Автоматная
    - Разделение ответственности:
      - Аспектно-ориентированная
      - Субъектно-ориентированная
    - Прототип-ориентированная

Парадигмы программирования

Декларативное программирование 1. Программа «декларативна», если она описывает *каково́* нечто, а не *как его создать*. Например, веб-страницы на HTML декларативны, так как они описывают *что* должна содержать страница, а не *как отображать* страницу на экране. Этот подход отличается от языков императивного программирования, требующих от программиста указывать алгоритм для исполнения.

Декларативное программирование 2. Программа «декларативна», если она написана на исключительно функциональном Программа «декларативна», если она написана на исключительно функциональном, логическом Программа «декларативна», если она написана на исключительно функциональном, логическом или языке программирования с ограничениями Программа «декларативна», если она написана на исключительно

Парадигмы программирования

Императивное программирование — это парадигма программирования — это парадигма программирования, которая, в отличие от декларативного программирования — это парадигма программирования, которая, в отличие от декларативного программирования, описывает процесс вычисления в виде инструкций — это парадигма программирования, которая, в отличие от декларативного программирования, описывает процесс вычисления в виде инструкций, изменяющих состояние программы. Императивная программа очень похожа на приказы, выражаемые повелительным наклонением в естественных языках — это парадигма программирования, которая, в отличие от декларативного программирования, описывает процесс вычисления в виде инструкций, изменяющих состояние программы. Императивная программа очень похожа на приказы, выражаемые

Парадигмы программирования

Функциональное программирование — парадигма программирования — парадигма программирования, в которой процесс вычисления — парадигма программирования, в которой процесс вычисления трактуется как вычисление значений функций — парадигма программирования, в которой процесс вычисления трактуется как вычисление значений функций в математическом понимании последних (в отличие от функций как подпрограмм в процедурном программировании).

Противопоставляется парадигме императивного программированияПротивопоставляется парадигме императивного программирования, которая описывает процесс вычислений как последовательное изменение состоянийПротивопоставляется парадигме императивного программирования, которая описывает процесс вычислений как последовательное изменение состояний (в значении, подобном таковому в теории автоматовПротивопоставляется парадигме императивного программирования, которая описывает процесс

Парадигмы программирования

Событийно-ориентированное программирование (англ. *event-driven programming*; СОП) — парадигма программирования; СОП) — парадигма программирования, в которой выполнение программы; СОП) — парадигма программирования, в которой выполнение программы определяется событиями; СОП) — парадигма программирования, в которой выполнение программы определяется событиями — действиями пользователя (клавиатура, мышь), сообщениями других программ и потоков, событиями операционной системы (например, поступлением сетевого пакета).

СОП можно также определить как способ построения компьютерной программы, при котором в коде (как правило, в головной функции программы) явным образом выделяется *главный цикл приложения*, тело которого состоит из двух частей: *выборки события* и *обработки события*.

Парадигмы программирования

Процедурное (императивное) программирование является отражением архитектуры является отражением архитектуры традиционных ЭВМ является отражением архитектуры традиционных ЭВМ, которая была предложена фон Нейманом является отражением архитектуры традиционных ЭВМ, которая была предложена фон Нейманом. Теоретической моделью процедурного программирования Машина Тьюринга. Выполнение программы сводится к последовательному выполнению операторов с целью преобразования исходного состояния памяти, то есть значений исходных данных, в заключительное, то есть в результаты. Таким образом, с точки зрения программиста имеются **программа и память**, причем первая последовательно обновляет содержимое последней.

Процедурный язык программирования предоставляет возможность программисту определять каждый шаг в

Парадигмы программирования

Объектно-ориентированное, или **объектное**, программирование (ООП) — парадигма программирования, программирование (ООП) — парадигма программирования, в которой основными концепциями, программирование (ООП) — парадигма программирования, в которой основными концепциями являются понятия объектов, программирование (ООП) — парадигма программирования, в которой основными концепциями являются понятия объектов и классов.

Объект — это сущность, которой можно посылать сообщения, и которая может на них реагировать, используя свои данные. Данные объекта скрыты от остальной программы. Сокрытие данных называется инкапсуляцией.

Класс описывает множество объектов одного типа. Новые

Парадигмы программирования

Парадигма программирования	**Язык программирования**
Декларативная (функциональная)	Lisp
Декларативная (логическая)	Prolog
Императивная (процедурная)	Fortran, Pascal, C
Объектно-ориентированная с событиями	C++, Delphi, Perl, Java, Smalltalk, Python, C#.
Табличная (формулы + события, связанные с изменениями данных)	Excel
Реляционная (таблицы данных + запросы)	SQL

Парадигмы программирования

Программы = Алгоритмы + структуры данных (Н. Вирт)

Паскаль и Вирт

Парадигмы программирования

Программы = Алгоритмы + структуры данных

Что нужно помнить:

1. Разные способы представления одних и тех же данных связаны с разными способами их обработки (разными алгоритмами).
2. Для решения каждой конкретной задачи всегда найдется эффективный способ представления данных

Пример-задача. Как оптимальным образом представить целое число, чтобы компьютерная программа могла определить, делится оно нацело на 3? А на 9? А на 19? А на 21?
Подсказка: Пользоваться можно любыми элементарными функциями из школьной программы.

(Кнут Д.Э. Искусство программирования. В 3-х томах. М: Вильямс, 2007. 720 с., 832 с., 824 с.)

Парадигмы программирования

Программы = Алгоритмы + структуры данных

Пример-задача. Как оптимальным образом представить целое число, чтобы компьютерная программа могла определить, делится оно нацело на 3? А на 9? А на 19? А на 21?
Подсказка: Пользоваться можно любыми элементарными функциями из школьной программы.

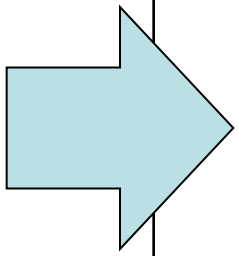
Ответ. Число N делится нацело на число K, если

$$Cos(2\pi K/N) = 1$$

(Кнут Д.Э. Искусство программирования. В 3-х томах. М: Вильямс, 2007. 720 с., 832 с., 824 с.)

Обработка информации как "прикладная математика"

Математические объекты	**Программные объекты**
элементы, множества, отношения, операции, пространства, отображения, числа, фигуры, последовательности, функции, векторы, матрицы, тензоры…	числа, функции, процедуры, массивы, таблицы, списки, указатели, объекты, классы, типы…



Числа - базовый тип данных в прикладной математике

Числовые системы	
Счётные множества	Натуральные числа ($\mathbb{N}$) • Целые ($\mathbb{Z}$) • Рациональные ($\mathbb{Q}$) • Алгебраические ($\overline{\mathbb{Q}}$) • Периоды • Вычислимые • Арифметические
Вещественные числа и их расширения	Вещественные ($\mathbb{R}$) • Комплексные ($\mathbb{C}$) • Кватернионы ($\mathbb{H}$) • Числа Кэли (октавы, октонионы) ($\mathbb{O}$) • Седенионы ($\mathbb{S}$) • Процедура Кэли-Диксона (*en*) • Дуальные • Гиперкомплексные • Superreal number (*англ.*) • Hyperreal number (*англ.*) • Surreal number (*англ.*)
Другие числовые системы	Кардинальные числа • Порядковые числа (трансфинитные, ординал) • p-адические • Супернатуральные числа
См. также	Двойные числа • Иррациональные числа • Трансцендентные • Числовой луч • Бикватернион

Определение: "Число" это такой математический объект, для которого определены 4 арифметических действия (+,-,×÷).

Основная идея: Разные типы чисел, как и все другие типы данных, появлялись в связи с необходимостью решать разные типы прикладных задач.

Подробнее см. ВикипедиЯ Свободная энциклопедия

Задачи и числа. Решение уравнений

$1,\ 2,\ \ldots$	Натуральные числа

Для решения арифметических уравнений вида

$x = a + b$

$y = ab$

Задачи и числа. Решение уравнений

$1,\ 2,\ \ldots$	Натуральные числа
$0,\ 1,\ -1,\ \ldots$	Целые числа

Для решения арифметических уравнений вида

$b = a + x \Rightarrow x = b - a$

$y = ab$

Задачи и числа. Решение уравнений

$1,\ 2,\ \ldots$	Натуральные числа
$0,\ 1,\ -1,\ \ldots$	Целые числа
$1,\ -1,\ \frac{1}{2},\ \frac{2}{3},\ 0{,}12,\ \ldots$	Рациональные числа

Для решения арифметических уравнений вида
$b = ax \Rightarrow x = b/a$

Задачи и числа. Решение уравнений

$1,\ 2,\ \ldots$	Натуральные числа
$0,\ 1,\ -1,\ \ldots$	Целые числа
$1,\ -1,\ \frac{1}{2},\ \frac{2}{3},\ 0{,}12,\ \ldots$	Рациональные числа
$1,\ -1,\ \frac{1}{2},\ 0{,}12,\ \pi,\ \sqrt{2},\ \ldots$	Вещественные числа

Для решения алгебраических уравнений вида
$b = ax^n \Rightarrow x = \sqrt[n]{(b/a)}$

Задачи и числа. Решение уравнений

$1,\ 2,\ \ldots$	Натуральные числа
$0,\ 1,\ -1,\ \ldots$	Целые числа
$1,\ -1,\ \frac{1}{2},\ \frac{2}{3},\ 0{,}12,\ \ldots$	Рациональные числа
$1,\ -1,\ \frac{1}{2},\ 0{,}12,\ \pi,\ \sqrt{2},\ \ldots$	Вещественные числа
$-1,\ \frac{1}{2},\ 0{,}12,\ \pi,\ 3i+2,\ e^{i\pi/3},\ \ldots$	Комплексные числа

Для решения любых алгебраических уравнений:

$c_0 + c_1x + c_2x^2 + c_3x^3 + \ldots. = 0$

$\Rightarrow x = a + ib$, где $i = \sqrt{-1}$.

Дальнейшее расширение понятия "числа"

$1,\ 2,\ \ldots$	Натуральные числа
$0,\ 1,\ -1,\ \ldots$	Целые числа
$1,\ -1,\ \frac{1}{2},\ \frac{2}{3},\ 0{,}12,\ \ldots$	Рациональные числа
$1,\ -1,\ \frac{1}{2},\ 0{,}12,\ \pi,\ \sqrt{2},\ \ldots$	Вещественные числа
$-1,\ \frac{1}{2},\ 0{,}12,\ \pi,\ 3i+2,\ e^{i\pi/3},\ \ldots$	Комплексные числа
$1,\ i,\ j,\ k,\ \pi j - \frac{1}{2}k,\ \ldots$	Кватернионы
	Октонионы

Подробнее см. ВикипедиЯ Свободная энциклопедия

Комплексные числа и операции с точками на плоскости

Действия над комплексными числами

- Сравнение

  $a + bi = c + di$ означает, что $a = c$ и $b = d$ (два комплексных числа равны между собой тогда и только тогда, когда равны их действительные и мнимые части).

- Сложение

  $(a + bi) + (c + di) = (a + c) + (b + d)i$.

- Вычитание

  $(a + bi) - (c + di) = (a - c) + (b - d)i$.

- Умножение

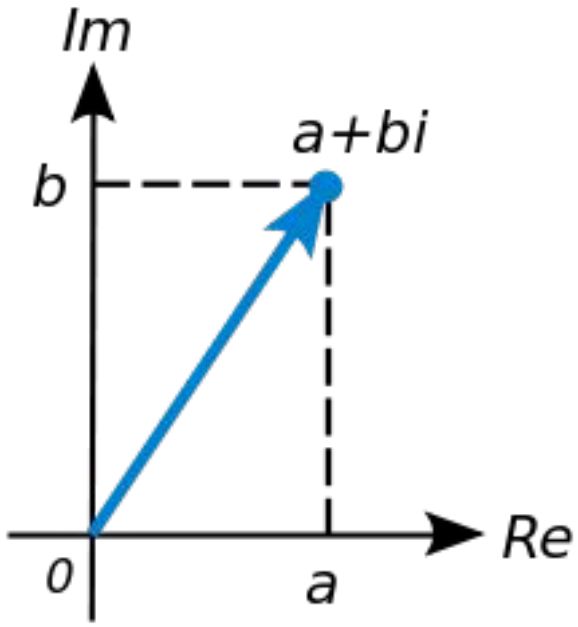
  $$(a + bi) \cdot (c + di) = ac + bci + adi + bdi^2 = (ac - bd) + (bc + ad)i.$$

- Деление

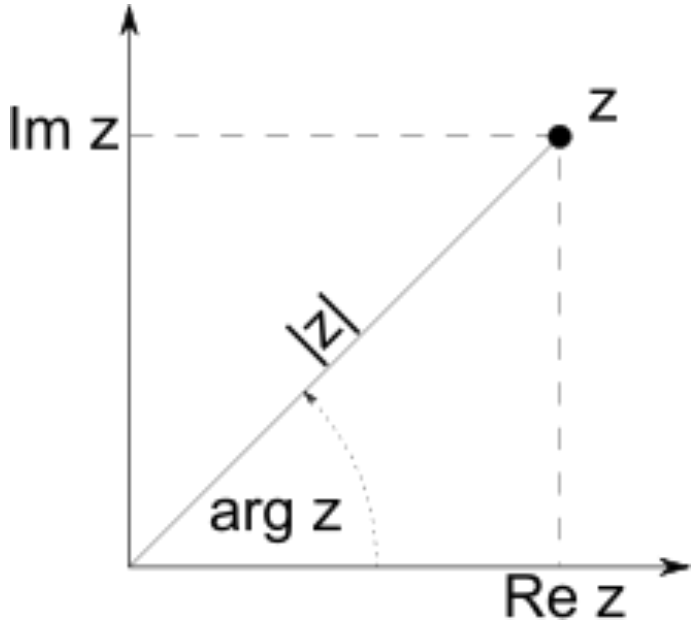
  $$\frac{a + bi}{c + di} = \frac{ac + bd}{c^2 + d^2} + \left(\frac{bc - ad}{c^2 + d^2}\right) i.$$

Комплексные числа и операции с точками на плоскости

Геометрическая модель



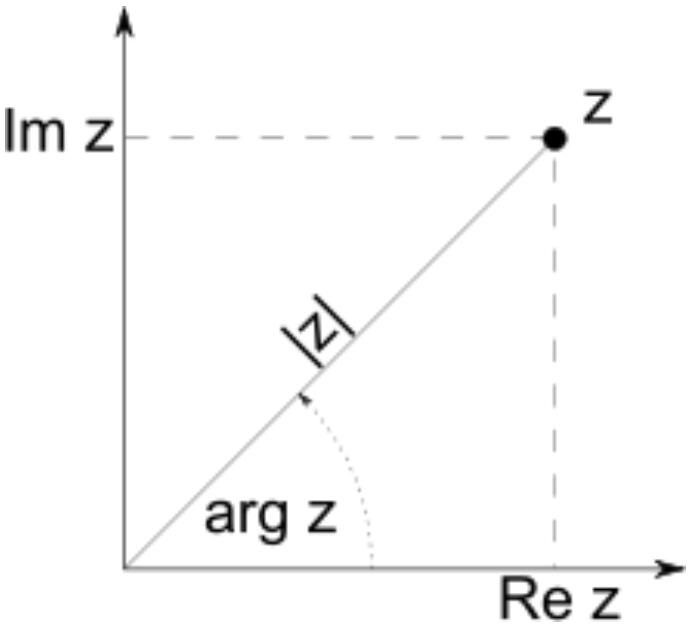
Геометрическое представление комплексного числа



Модуль, аргумент, вещественная и мнимая части

Комплексные числа и операции с точками на плоскости

Геометрическая модель



Модуль комплексного числа *z* обозначается | *z* | и определяется выражением $|z| = \sqrt{x^2 + y^2}$. Если *z* является вещественным числом, то | *z* | совпадает с абсолютной величиной этого вещественного числа.

Для любых $z, z_1, z_2 \in \mathbb{C}$ имеют место следующие свойства модуля. :

1) $|z| \geqslant 0$, причём $|z| = 0$ тогда и только тогда, когда $z = 0$;
2) $|z_1 + z_2| \leqslant |z_1| + |z_2|$ (неравенство треугольника);
3) $|z_1 \cdot z_2| = |z_1| \cdot |z_2|$
4) $|z_1 / z_2| = |z_1| / |z_2|$.

Из третьего свойства следует $|a \cdot z| = |a| \cdot |z|$, где $a \in \mathbb{R}$. Данное свойство модуля вместе с первыми двумя свойствами вводят на множестве комплексных чисел структуру двумерного **нормированного пространства** над полем $\mathbb{R}$.

Кватернионы и операции с точками пространства

Кватернион представляет собой паруКватернион представляет собой пару $(a, \vec{u})$, где $\vec{u}$ — вектор трёхмерного пространства, а a — скаляр, то есть вещественное число. Операции сложения определены следующим образом:

$$(a, \vec{u}) + (b, \vec{v}) = (a + b, \vec{u} + \vec{v})$$

Произведение определяется следующим образом:

$$(a, \vec{u})\,(b, \vec{v}) = (ab - \vec{u} \cdot \vec{v}, a\vec{v} + b\vec{u} + \vec{u} \times \vec{v})$$

где $*$ обозначает скалярное произведение, а $\times$ — векторное произведение.

Кватернионы и операции с точками пространства

Кватернионы можно определить как формальную сумму $a + bi + cj + dk$, где a, b, c, d — вещественные числа, а i, j, k — *мнимые единицы* со следующим свойством: $i^2 = j^2 = k^2 = ijk = -1$.

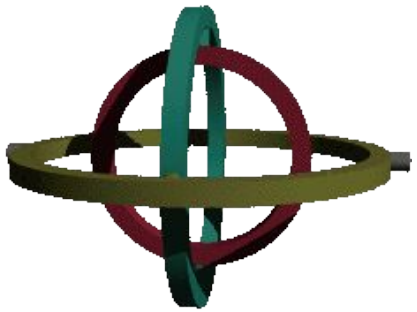
Таким образом, таблица умножения *базисных кватернионов* — — выглядит так:

.	1	*i*	*j*	*k*
1	1	i	j	k
i	i	-1	k	$-j$
j	j	$-k$	-1	i
k	k	j	$-i$	-1

например, $ij = k$, а $ji = -k$.

Так же, как и для комплексных чисел, $|q| = \sqrt{q\bar{q}} = \sqrt{a^2 + b^2 + c^2 + d^2}$ называется *модулем* q.

Кватернионы и повороты пространства



Организация трёх степеней свободы, но окончательная свобода меньших колец зависит от положения больших колец

Октонионы и операции с точками пространства

Октонион (октава, число Кэли) — это линейная комбинация элементов $\{1, i, j, k, l, il, jl, kl\}$

Каждая октава *x* может быть записана в форме

$$x = x_0 + x_1\, i + x_2\, j + x_3\, k + x_4\, l + x_5\, il + x_6\, jl + x_7\, kl.$$

с вещественными коэффициентами x_i **Норма октониона**

$$\|x\|^2 = x^* x = x_0^2 + x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2 + x_7^2.$$

Таблица умножения элементов октавы:

1	***i* (*e1*)**	***j* (*e2*)**	***k* (*e3*)**	***l* (*e4*)**	***il* (*e5*)**	***jl* (*e6*)**	***kl* (*e7*)**
***i* (*e1*)**	−1	*k*	−*j*	*il*	−*l*	−*kl*	*jl*
***j* (*e2*)**	−*k*	−1	*i*	*jl*	*kl*	−*l*	−*il*
***k* (*e3*)**	*j*	−*i*	−1	*kl*	−*jl*	*il*	−*l*
***l* (*e4*)**	−*il*	−*jl*	−*kl*	−1	*i*	*j*	*k*
***il* (*e5*)**	*l*	−*kl*	*jl*	−*i*	−1	−*k*	*j*
***jl* (*e6*)**	*kl*	*l*	−*il*	−*j*	*k*	−1	−*i*
***kl* (*e7*)**	−*jl*	*il*	*l*	−*k*	−*j*	*i*	−1

Других чисел нет!

Тождество квадратов | *ab* | = | *a* | | *b* |

Теория чисел

Одного	модуль произведения двух *действительных чисел* равен произведению модулей сомножителей
Двух (Тождество Брахмагупты)	модуль произведения двух *комплексных чисел* равен произведению модулей сомножителей
Четырёх (Тождество Эйлера)	модульмодуль произведения двух кватернионовмодуль произведения двух кватернионов равен произведению модулей сомножителей
Восьми	модуль произведения двух *октонионов* равен произведению модулей сомножителей
Шестнадцати (и более)	Не существует ни для 16 (седенионы), ни для любого другого числа квадратов, кроме 1, 2, 4 и 8

Других чисел нет!

Тождество квадратов

Тождество Эйлера о четырёх квадратах — математическая теорема о том, что

произведение сумм четырёх квадратов является суммой четырёх квадратов.

$$
\begin{aligned}
&(a_1^2 + a_2^2 + a_3^2 + a_4^2)(b_1^2 + b_2^2 + b_3^2 + b_4^2) \\
&= (a_1 b_1 - a_2 b_2 - a_3 b_3 - a_4 b_4)^2 + (a_1 b_2 + a_2 b_1 + a_3 b_4 - a_4 b_3)^2 \\
&+ (a_1 b_3 - a_2 b_4 + a_3 b_1 + a_4 b_2)^2 + (a_1 b_4 + a_2 b_3 - a_3 b_2 + a_4 b_1)^2
\end{aligned}
$$

Других чисел нет!

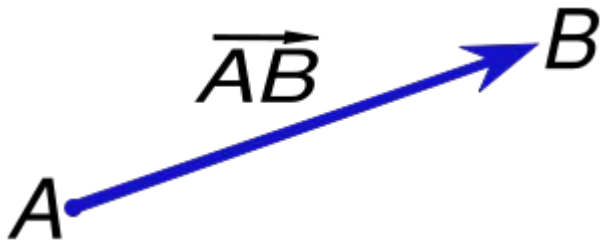
Тождество квадратов

Тождество восьми квадратов — математическая теорема о том, что

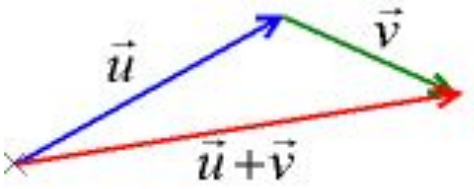
произведение сумм восьми квадратов является суммой восьми квадратов.

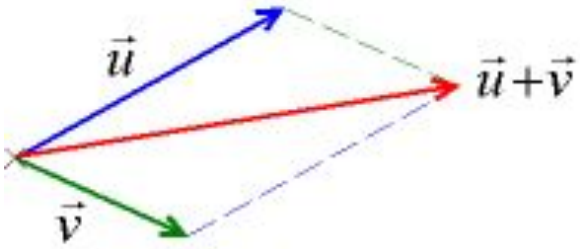
$$
\begin{aligned}
&(a_1^2+a_2^2+a_3^2+a_4^2+a_5^2+a_6^2+a_7^2+a_8^2)(b_1^2+b_2^2+b_3^2+b_4^2+b_5^2+b_6^2+b_7^2+b_8^2) = \\
&\quad (a_1b_1 - a_2b_2 - a_3b_3 - a_4b_4 - a_5b_5 - a_6b_6 - a_7b_7 - a_8b_8)^2 + \\
&\quad (a_2b_1 + a_1b_2 + a_4b_3 - a_3b_4 + a_6b_5 - a_5b_6 - a_8b_7 + a_7b_8)^2 + \\
&\quad (a_3b_1 - a_4b_2 + a_1b_3 + a_2b_4 + a_7b_5 + a_8b_6 - a_5b_7 - a_6b_8)^2 + \\
&\quad (a_4b_1 + a_3b_2 - a_2b_3 + a_1b_4 + a_8b_5 - a_7b_6 + a_6b_7 - a_5b_8)^2 + \\
&\quad (a_5b_1 - a_6b_2 - a_7b_3 - a_8b_4 + a_1b_5 + a_2b_6 + a_3b_7 + a_4b_8)^2 + \\
&\quad (a_6b_1 + a_5b_2 - a_8b_3 + a_7b_4 - a_2b_5 + a_1b_6 - a_4b_7 + a_3b_8)^2 + \\
&\quad (a_7b_1 + a_8b_2 + a_5b_3 - a_6b_4 - a_3b_5 + a_4b_6 + a_1b_7 - a_2b_8)^2 + \\
&\quad (a_8b_1 - a_7b_2 + a_6b_3 + a_5b_4 - a_4b_5 - a_3b_6 + a_2b_7 + a_1b_8)^2
\end{aligned}
$$

А другие пространства есть… Векторы и "Векторы"



Вектор $\overrightarrow{AB}$





Два вектора *u*, *v* и вектор их суммы

Векторное пространство

$(a+b)\mathbf{x} = a\mathbf{x} + b\mathbf{x}$

$a(\mathbf{x}+\mathbf{y}) = a\mathbf{x} + a\mathbf{y}$

$(a*b)\mathbf{x} = a(b\mathbf{x})$

$1\mathbf{x} = \mathbf{x}$

Определение: "Вектор" это такой математический объект, для которого определены операции сложения векторов и умножения вектора на число.

Примечание: Как и "чисел", "векторов" в математике множество. Например, функции также образуют векторное пространство.

А другие пространства есть… Матрицы

Матрица — математический объект, записываемый в виде прямоугольной таблицы чисел, которая представляет собой совокупность строк — математический объект, записываемый в виде прямоугольной таблицы чисел, которая представляет собой совокупность строк и столбцов, на пересечении которых находятся её элементы. Количество строк и столбцов матрицы задают размер матрицы.

$$\begin{pmatrix} a_{11} & \cdots & a_{1j} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{i1} & \cdots & a_{ij} & \cdots & a_{in} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mj} & \cdots & a_{mn} \end{pmatrix}, \quad \begin{bmatrix} a_{11} & \cdots & a_{1j} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{i1} & \cdots & a_{ij} & \cdots & a_{in} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mj} & \cdots & a_{mn} \end{bmatrix}, \quad \begin{Vmatrix} a_{11} & \cdots & a_{1j} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{i1} & \cdots & a_{ij} & \cdots & a_{in} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mj} & \cdots & a_{mn} \end{Vmatrix}$$

Сложение матриц A + B есть операция нахождения матрицы *C*,

$$c_{ij} = a_{ij} + b_{ij}$$

Умножение матриц (обозначение: *AB*) — вычисление матрицы *C*,

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

Задачи и матрицы. Системы линейных уравнений

Систему из *m* уравнений с *n* неизвестными

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m \end{cases}$$

можно представить в матричном виде

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}; \quad X = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}; \quad B = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}$$

и тогда всю систему можно записать так:

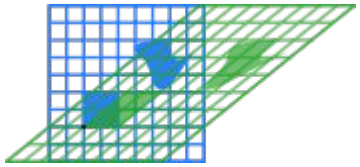
$$AX = B.$$

Задачи и матрицы. Линейные преобразования пространства

В частном случае, когда рассматриваются линейные преобразования плоскости:

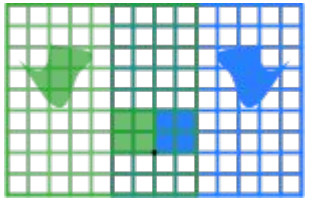
Горизонтальный скос (m=1.25)

$$\begin{bmatrix} 1 & 1.25 \\ 0 & 1 \end{bmatrix}$$



Горизонтальное отражение

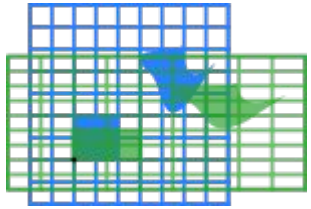
$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$



$$x' = a_{11}x + a_{12}y$$
$$y' = a_{21}x + a_{22}y$$

Сжатие (r=3/2)

$$\begin{bmatrix} 3/2 & 0 \\ 0 & 2/3 \end{bmatrix}$$

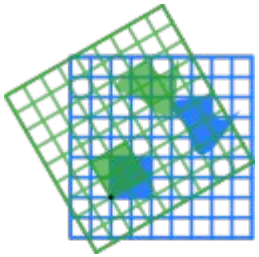


Масштабирование (3/2)

$$\begin{bmatrix} 3/2 & 0 \\ 0 & 3/2 \end{bmatrix}$$

Поворот ($\pi/6^R$ = 30°)

$$\begin{bmatrix} \cos(\pi/6^R) & -\sin(\pi/6^R) \\ \sin(\pi/6^R) & \cos(\pi/6^R) \end{bmatrix}$$



Представление чисел в компьютерных системах

- Целочисленные Целочисленные: со знаком, то есть могут принимать как положительные, так и отрицательные значения; и без знака, то есть могут принимать только неотрицательные значения.
- Вещественные Вещественные: с запятой Вещественные: с запятой (то есть хранятся знак и цифры целой и дробной частей) и с плавающей запятой (то есть число приводится к виду $m*b^e$, где m — мантисса, где m — мантисса, b — основание показательной функции, где m — мантисса, b — основание показательной функции, e — показатель степени (порядок, где m — мантисса, b — основание показательной функции, e — показатель степени (порядок) (в англоязычной литературе экспонента), причём в *нормальной форме* 0<=m<b, а в *нормализованной форме* 1<=m<b, e — целое число и хранятся знак и числа m и e).
- Числа произвольной точности, обращение с которыми происходит посредством длинной арифметики Числа

Подробнее см.

ВИКИПЕДИЯ
Свободная энциклопедия

Представление чисел и других данных в компьютерных системах

Типы данных	
Неинтерпретируемые	Бит · Ниббл · Байт · Трит · Трайт · Слово
Числовые	Целый · Фиксированная запятая · **С плавающей запятой** · Рациональный · Комплексный · Длинный · Интервальный
Текстовые	Символьный · Строковый
Указатель	Адрес · Ссылка
Композитные	Алгебраический тип данных (обобщённый) · Массив · Ассоциативный массив · Класс · Список · Объект · Option type · Product · Структура · Множество · Объединение (tagged)
Другие	Логический · Низший тип · Коллекция · Перечисляемый тип · Исключение · First-class function · Opaque data type · Recursive data type · Семафор · Поток · Высший тип · Type class · Unit type · Void
Связанные темы	Абстрактный тип данных · Структура данных · Интерфейс · Kind (type theory) · Примитивный тип · Subtyping · Шаблон · Конструктор типа · Parametric polymorphism

Контрольный вопрос по предыдущей лекции. Действительно ли каждый бит натурального числа, записанного в компьютере в двоичной системе, имеет информативность 1 бит?

Подсказка: Все дело в разрядности и диапазоне значений.

Подробнее см. ВикипедиЯ Свободная энциклопедия

Натуральные двоичные числа

Преобразование десятичных чисел в двоичные

19 /2 = 9 с остатком 1
9 /2 = 4 с остатком 1
4 /2 = 2 без остатка 0
2 /2 = 1 без остатка 0
1 /2 = 0 с остатком 1

В результате получаем число 19 в двоичной записи: 10011.

Преобразование двоичных чисел в десятичные

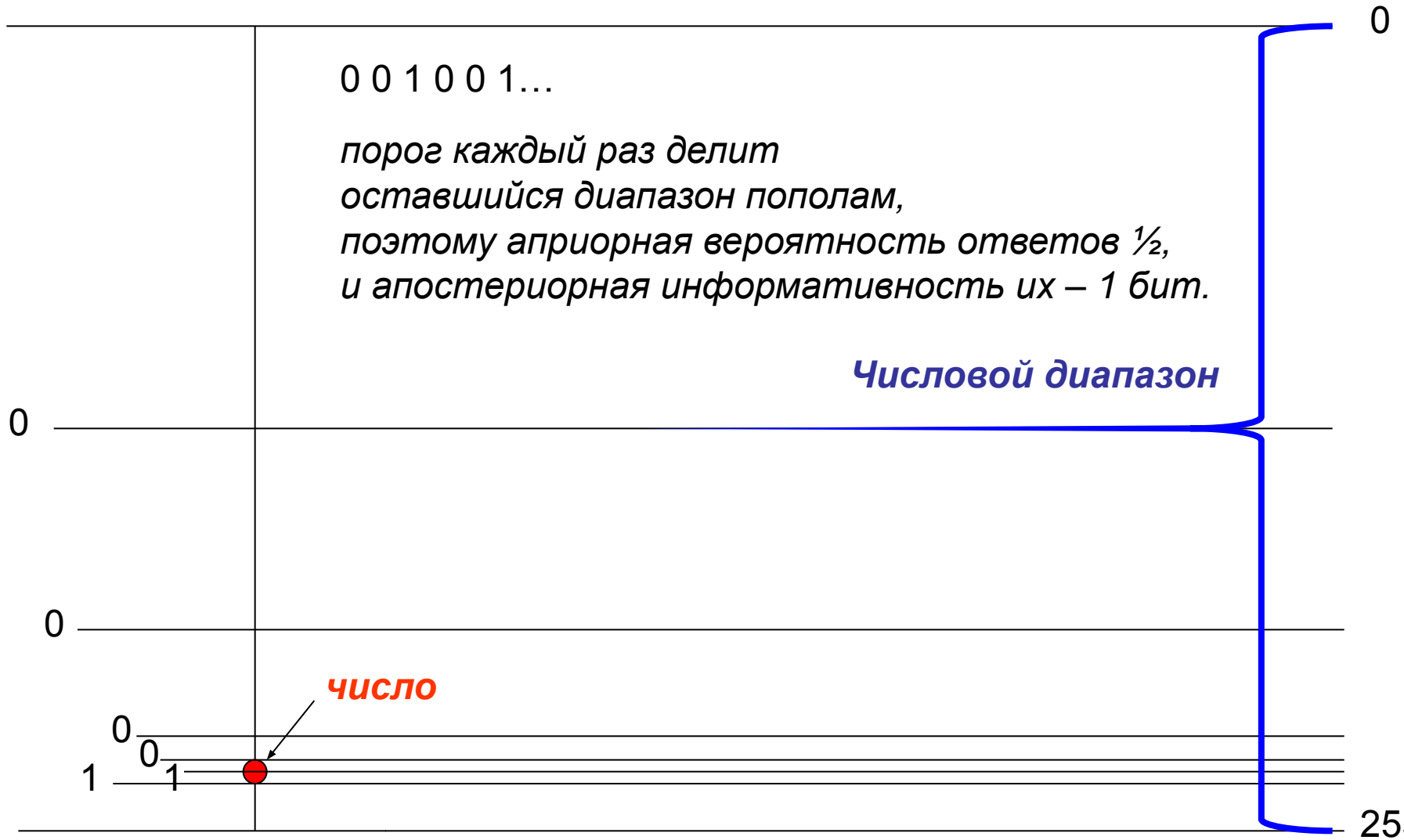
Пусть дано двоичное число 110001. Для перевода в десятичное запишите его справа налево как сумму по разрядам следующим образом:

$$1\times2^0+0\times2^1+0\times2^2+0\times2^3+1\times2^4+1\times2^5 =$$

$$= 1\times1+0\times2+0\times4+0\times8+1\times16+1\times32 = 49$$

Информативность натуральных двоичных чисел

Вопрос: Число больше порога? Да – 1, Нет – 0.



Представление целых чисел в ЭВМ

Дополнительный код (англ. *two's complement*, иногда *twos-complement*) — наиболее распространённый способ представления отрицательных целых чисел в компьютерах) — наиболее распространённый способ представления отрицательных целых чисел в компьютерах. Он позволяет заменить операцию вычитания на операцию сложения и сделать операции сложения и вычитания одинаковыми для знаковых и беззнаковых чисел. Дополнительный код отрицательного числа можно получить инвертированием модуля двоичного числа (первое дополнение) и прибавлением к инверсии единицы (второе дополнение). Либо вычитанием числа из нуля.

Двоичное 8-ми разрядное число *со знаком* в дополнительном коде может представлять любое целое в диапазоне от −128 до +127. .

Десятичное представление	**Код двоичного представления (8 бит)**	
	прямой	**дополнительный**
127	01111111	01111111
1	00000001	00000001
0	00000000	00000000
-0	10000000	00000000
-1	10000001	11111111
-2	10000010	11111110
-3	10000011	11111101

Представление дробных чисел в ЭВМ

вещественного числа в памяти ЭВМ — формат представления рационального или вещественного числа в памяти ЭВМ в виде целого числа. При этом само число x и его целочисленное представление x' связаны формулой $x = x' \cdot z$, где z — цена (вес) младшего разряда.

Простейший пример арифметики с фиксированной запятой — перевод рублейПростейший пример арифметики с фиксированной запятой — перевод рублей в копейки. В таком случае, чтобы запомнить сумму 12 рублей 34 копейки, мы записываем в ячейку памяти число 1234.

Недостаток фиксированной запятой — очень узкий диапазон чисел, с угрозой переполненияНедостаток фиксированной запятой — очень узкий диапазон чисел, с угрозой переполнения на одном конце диапазона и потерей точностиНедостаток фиксированной запятой — очень узкий диапазон чисел, с угрозой переполнения на одном конце диапазона и потерей точности вычислений на другом. Эта проблема и привела к изобретению плавающей запятойНедостаток фиксированной запятой — очень узкий диапазон чисел, с угрозой переполнения на одном конце диапазона и потерей точности вычислений на другом. Эта проблема и привела к изобретению плавающей запятой. Например: если нужна точность в 3 значащих цифры, 4-байтовая фиксированная запятая даёт диапазон в 6 порядков (то есть, разница приблизительно 106 между самым большим и самым маленьким числом), 4-байтовое число одинарной точности — в 70 порядков.

Представление действительных чисел

Плавающая запятая — форма представления действительных чисел, в которой число хранится в форме мантиссы — форма представления действительных чисел, в которой число хранится в форме мантиссы и показателя степени — форма представления действительных чисел, в которой число хранится в форме мантиссы и показателя степени. При этом число с плавающей запятой имеет фиксированную относительную точность и изменяющуюся абсолютную. Например, число 1,528535047 ×10−25 в большинстве языков программирования высокого уровня записывается как 1.528535047E-25.

Точность	**Одинарная**	**Двойная**	**Расширенная**
Размер (байты)	4	8	10
Число десятичных знаков	7	15	19
Наименьшее значение (>0), denorm	$1{,}4\times10^{-45}$	$5{,}0\times10^{-324}$	$1{,}9\times10^{-4951}$
Наименьшее значение (>0), normal	$1{,}2\times10^{-38}$	$2{,}3\times10^{-308}$	$3{,}4\times10^{-4932}$
Наибольшее значение	$3{,}4\times10^{+38}$	$1{,}7\times10^{+308}$	$1{,}1\times10^{+4932}$
Поля	S-E-F	S-E-F	S-E-I-F
Размеры полей	1-8-23	1-11-52	1-15-1-63

- S — знак, E — показатель степени, I — целая часть, F — дробная часть
- Так же, как и для целых, знаковый бит — старший.

Типы данных

- Простые.
  - Перечислимый тип. Может хранить только те значения, которые прямо указаны в его описании.
  - Числовые данные.
  - Символьный типСимвольный тип. Хранит один символСимвольный тип. Хранит один символ. Могут использоваться различные кодировки.
  - Логический типЛогический тип. Имеет два значения: истинаЛогический тип. Имеет два значения: истина и ложьЛогический тип. Имеет два значения: истина и ложь, при троичной логикеЛогический тип. Имеет два значения: истина и ложь, при троичной логике может иметь и третье значение — «не определено» (или «неизвестно»). Могут применяться логические операцииЛогический тип. Имеет два значения: истина и ложь, при троичной логике может иметь и третье значение — «не определено» (или «неизвестно»). Могут применяться логические операции. Используется в операторах ветвленияЛогический тип. Имеет два значения: истина и ложь, при троичной логике может иметь и третье значение — «не определено» (или «неизвестно»). Могут применяться логические операции. Используется в операторах ветвления и циклахЛогический тип. Имеет два значения: истина и

Типы данных

- Составные (сложные).
  - МассивМассив. Является индексированным набором элементов одного типа. Одномерный массивМассив. Является индексированным набором элементов одного типа. Одномерный массив — векторМассив. Является индексированным набором элементов одного типа. Одномерный массив — вектор, двумерный массивМассив. Является индексированным набором элементов одного типа. Одномерный массив — вектор, двумерный массив — матрица.
    - Строковый типСтроковый тип. Хранит строкуСтроковый тип. Хранит строку символов. Аналогом сложения в строковой алгебре является конкатенацияСтроковый тип. Хранит строку символов. Аналогом сложения в строковой алгебре является конкатенация (прибавление одной строки в конец другой строки). В языках, близких к бинарному представлению данных, чаще рассматривается как массив символов, в языках более высокой абстракции зачастую выделяется в качестве простого.
  - Запись (структура). Набор различных элементов (полей записи), хранимый как единое целое. Возможен доступ к отдельным полям записи. Например, struct в C или record в Pascal.
  - Класс.

Подробнее см. ВикипедиЯ Свободная энциклопедия

Типы данных

Структурные типы данных

- Регулярные массивы
- Структуры (записи, кортежи)
- Динамические списки и массивы переменной длинны
- Ациклические графы (деревья)
- Сетевые графы

Специализированные типы данных

- Таблицы и базы данных
- Текст (символы и символьные строки)
- Аудиоданные
- Векторная и растровая графика

Подробнее см. ВикипедиЯ Свободная энциклопедия

Типы данных. Массив

Индексный массив — именованный **набор однотипных переменных**, расположенных в памяти непосредственно друг за другом, доступ к которым осуществляется по индексу, расположенных в памяти непосредственно друг за другом, доступ к которым осуществляется по индексу (в отличие от списка).
Индекс массива — целое число, либо значение типа, приводимого к целому, указывающее на конкретный элемент массива.

Примеры:

```
int Array[10]; // Одномерный массив целых чисел размера 10
               // Нумерация элементов от 0 до 9
double Array[12][15]; // Двумерный массив вещественных чисел двойной точности
                      // размера 12 на 15. Нумерация строк от 0 то 11, столбцов от 0 до 14
```

Достоинства массива как структуры данных

- лёгкость вычисления адреса элемента по его индексу (поскольку элементы массива располагаются один за другим)
- одинаковое время доступа ко всем элементам
- экономия памяти - элементы состоят только из информационного поля

Недостатки

- невозможность удаления или добавления элемента без сдвига других
- при работе с массивом в стиле C (с указателями*) и при отсутствии дополнительных средств контроля — угроза выхода за границы массива и повреждения данных

***Указатель** Хранит адрес Хранит адрес в памяти компьютера Хранит адрес в памяти компьютера, указывающий на какую-либо информацию, как правило — указатель Хранит адрес в памяти компьютера, указывающий на какую-либо

Типы данных. Структура

Структура — конструкция большинства языков программирования, позволяющая содержать в себе **фиксированный набор переменных различных типов**.
В языках семейства Pascal структуры традиционно называют **записями** (англ. *record*).

Пример объявления структуры на C:

```
struct str_name {
                    int member_1;
                    float member_2;
                    char member_3[256];
                   };
```

Пример объявления структуры на Pascal:

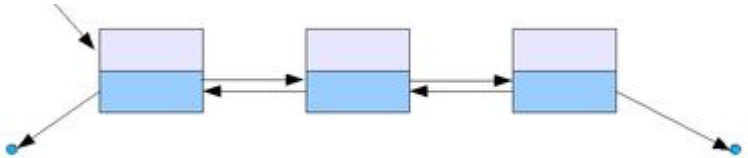
```
type str_name = record
begin
        member_1 : integer;
        member_2 : extended;
        member_3 : string;
end;
```

Типы данных. Список

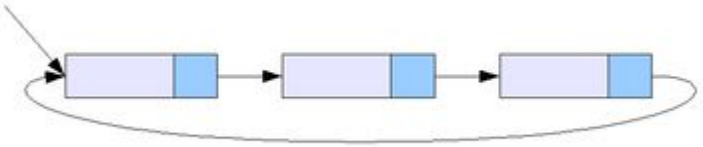
Связный список — структура данных, состоящая из узлов — структура данных, состоящая из узлов, каждый из которых содержит как собственно данные — структура данных, состоящая из узлов, каждый из которых содержит как собственно данные, так и одну или две ссылки («связки») на следующий и/или предыдущий узел списка.

Односвязный список

Двусвязный список



Кольцевой связный список



Достоинства списка как структуры данных

- лёгкость добавления и удаления элементов
- размер ограничен только объёмом памяти и разрядностью указателей

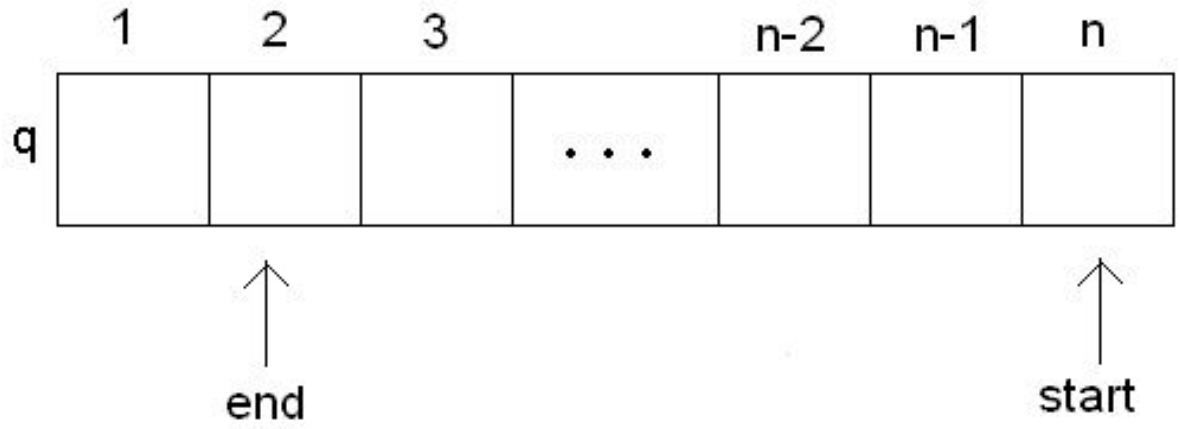
Недостатки

- сложность определения адреса элемента по его индексу (номеру) в списке
- на поля-указатели (указатели на следующий и предыдущий элемент) расходуется дополнительная память
- работа со списком медленнее, чем с массивами, так как к любому элементу списка можно обратиться, только пройдя все предшествующие ему элементы
- элементы списка могут быть расположены в памяти разреженно, что окажет негативный эффект на кэширование процессора
- над связными списками гораздо труднее (хотя и в принципе возможно) производить параллельные векторные операции, такие как вычисление суммы

Типы данных. Динамические списки типа "стек" и "очередь"

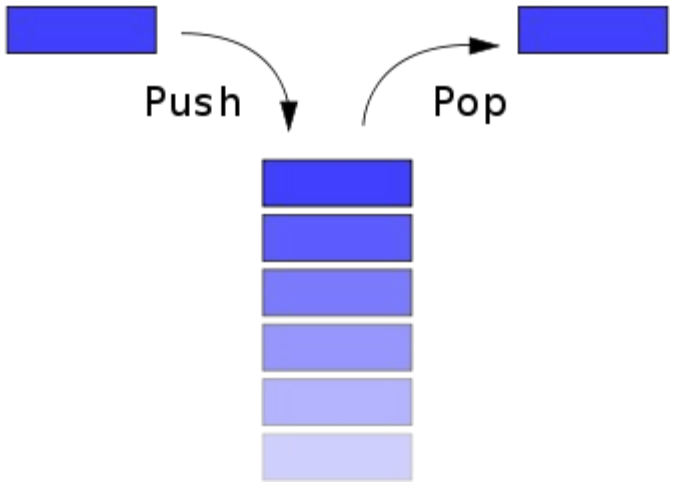
Очередь — структура данных — структура данных с дисциплиной доступа к элементам «первый пришёл — первый вышел» (FIFO, First In — First Out). Добавление элемента (принято обозначать словом enqueue — поставить в очередь) возможно лишь в конец очереди, выборка — только из начала очереди (что принято называть словом dequeue — убрать из очереди), при этом выбранный элемент из очереди удаляется.

Пример – очередь сообщений Windows.



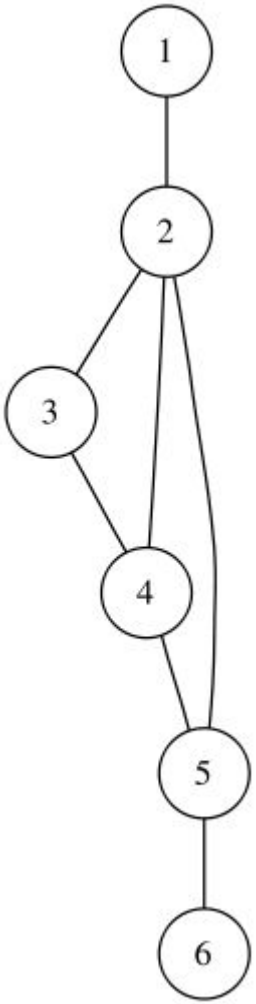
Типы данных. Динамические списки типа "стек" и "очередь"

Стек (англ. *stack* — стопка) — структура данных, в которой доступ к элементам организован по принципу *LIFO* (англ. *last in — first out*, «последним пришёл — первым вышел»). Чаще всего принцип работы стека сравнивают со стопкой тарелок: чтобы взять вторую сверху, нужно снять верхнюю.
Добавление элемента, называемое также проталкиванием (*push*), возможно только в вершину стека (добавленный элемент становится первым сверху).
Удаление элемента, называемое также выталкиванием (*pop*), тоже возможно только из вершины стека, при этом второй сверху элемент становится верхним.
Пример – стек вызова подпрограмм (рекурсивый вызов функций).



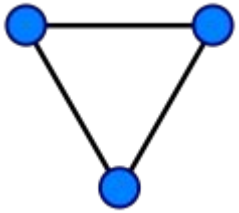
Типы данных. Графы (модели)

Граф — это совокупность непустого множества вершин и множества пар вершин (связей между вершинами). Объекты представляются как **вершины**, или **узлы** графа, а связи — как **дуги**, или **рёбра**. Для разных областей применения виды графов могут различаться направленностью, ограничениями на количество связей и дополнительными данными о вершинах или рёбрах.
Важно помнить: *Большинство структур, представляющих практический интерес в математике и информатике, могут быть представлены графами (включая массивы, записи и списки). В некотором смысле вся информатика может быть описана в терминах получения, преобразования и анализа графов.*
Модель: *совокупность элементов и отношений между ними (т.е. граф).*
Блок-схема *алгоритма – граф, и т.д.*



Неориентированный граф с шестью вершинами и семью рёбрами

Типы данных. Графы (модели)



Неориентированный граф G — это упорядоченная пара $G := (V,E)$, для которой выполнены следующие условия: V — это непустое множество **вершин**, или **узлов**, E — это множество пар (в случае неориентированного графа — неупорядоченных) вершин, называемых **рёбрами**.

Вершины и рёбра графа называются также **элементами** графа, число вершин в графе $|V|$ — **порядком**, число рёбер $|E|$ — **размером** графа.

Вершины u и v называются **концевыми** вершинами (или просто **концами**) ребра $e = \{u,v\}$. Ребро, в свою очередь, **соединяет** эти вершины. Две концевые вершины одного и того же ребра называются **соседними**.

Два ребра называются **смежными**, если они имеют общую концевую вершину.Два ребра называются **кратными**, если множества их концевых вершин совпадают. Ребро называется **петлёй**, если его концы совпадают, то есть $e = \{v,v\}$. **Степенью** $\deg V$ вершины V называют количество инцидентных ей рёбер(при этом петли считают дважды). Вершина называется **изолированной**, если она не является концом ни для одного ребра; **висячей** (или **листом**), если она является концом ровно одного ребра.

Типы данных. Графы (модели)

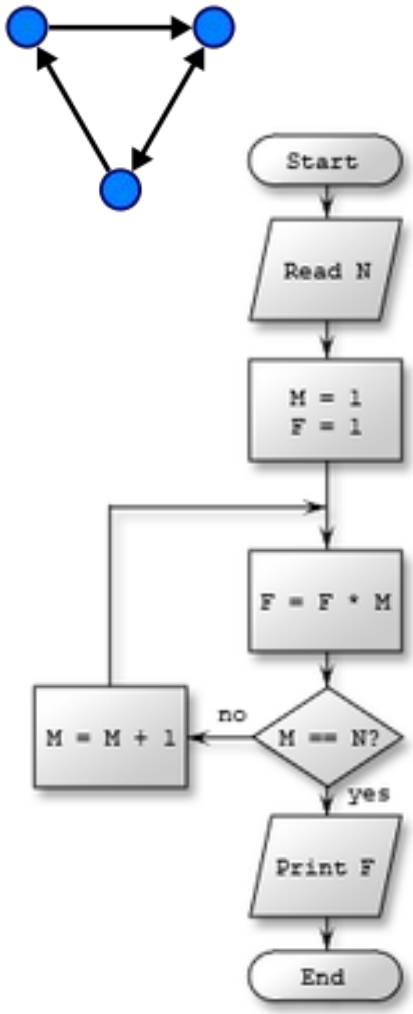
Ориентированный граф (сокращённо **орграф**) *G* — это упорядоченная пара *G*: = (*V*,*A*), для которой выполнены следующие условия: *V* — это непустое множество **вершин** или **узлов**, *A* — это множество (упорядоченных) пар различных вершин, называемых **дугами** или **ориентированными рёбрами**.

Дуга — это упорядоченная пара вершин (*v*,*w*), где вершину *v* называют началом, а *w* — концом дуги. Можно сказать, что дуга $v \to w$ ведёт от вершины *v* к вершине *w*.

Путём (или **цепью**) в графе называют конечную последовательность вершин, в которой каждая вершина (кроме последней) соединена со следующей в последовательности вершин ребром.

Ориентированным путём в орграфе называют конечную последовательность вершин v_i , для которой все пары (v_i,v_i + 1) являются (ориентированными) рёбрами.

Циклом называют путь, в котором первая и последняя вершины совпадают. При этом **длиной** пути (или цикла) называют число составляющих его *рёбер*.



Пример блок-схемы алгоритма вычисления факториала числа *N*

Типы данных. Графы (модели)

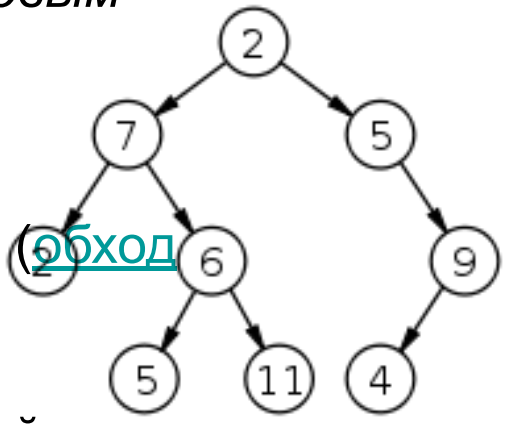
Граф называется:

- **связным**, если для любых вершин *u*,*v* есть путь из *u* в *v*.
- **сильно связным** или **ориентированно связным**, если он ориентированный, и из любой вершины в любую другую имеется ориентированный путь.
- **деревом**, если он связный и не содержит *простых* циклов.
- **полным**, если любые его две (различные, если не допускаются петли) вершины соединены ребром.
- **двудольным**, если его вершины можно разбить на два непересекающихся подмножества *V*1 и *V*2 так, что всякое ребро соединяет вершину из *V*1 с вершиной из *V*2.
- **k-дольным**, если его вершины можно разбить на *k* непересекающихся подмножества *V*1, *V*2, …, *Vk* так, что не будет рёбер, соединяющих вершины одного и того же подмножества.
- **полным двудольным**, если каждая вершина одного подмножества соединена ребром с каждой вершиной другого подмножества.
- **планарным**, если граф можно изобразить диаграммой на плоскости без пересечений рёбер.
- **взвешенным**, если каждому ребру графа поставлено в соответствие некоторое число, называемое весом ребра.

Типы данных. Графы: деревья

Дерево — является связанным графом, не содержащим циклы. На дереве:

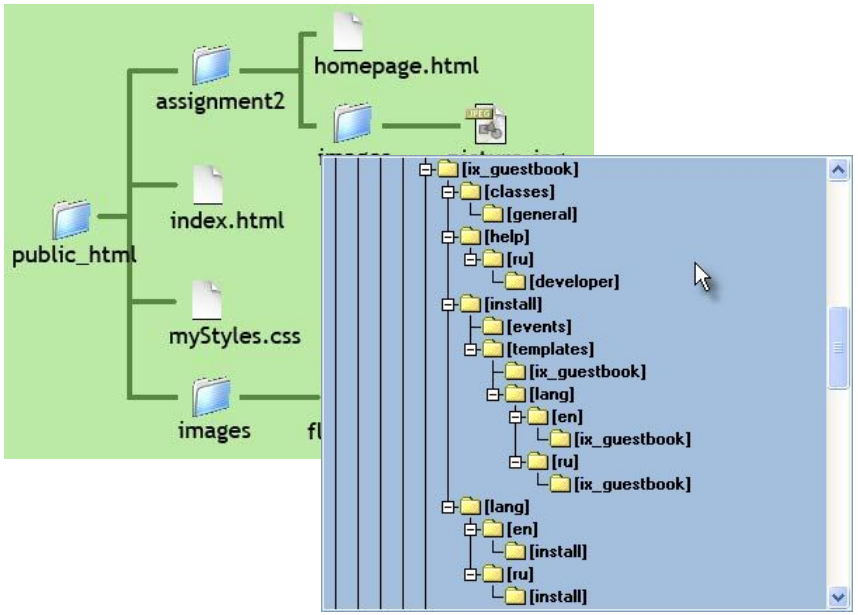
- *Корневой узел* — самый верхний узел дерева.
- *Корень* — одна из вершин, по желанию наблюдателя.
- *лист*, *листовой* или *терминальный узел* — узел, не имеющий дочерних элементов.
- *Внутренний узел* — любой *узел* дерева, имеющий *потомков*, и таким образом не являющийся *листовым узлом*.

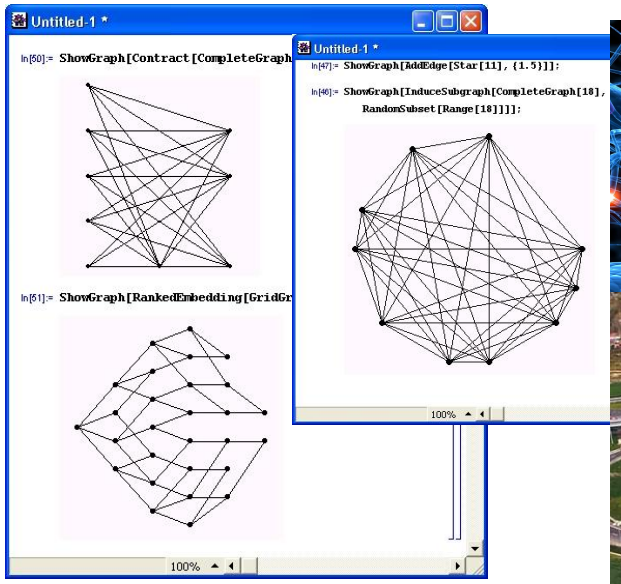


Основные применения

- управление иерархией данных;
- упрощение поискаупрощение поиска информации (обход дерева);
- управление сортированными списками данных;
- синтактический разбор арифметических выражений (англ. *parsing*), оптимизация программ;
- в качестве технологии компоновкив качестве технологии компоновки цифровых картинок для получения различных визуальных эффектов;
- В задачах принятия многоэтапного решения.

Типы данных. Графы: Деревья, Сети





Приложение 3. Санкт-Петербург

Подробнее см.

ВИКИПЕДИЯ
Свободная энциклопедия

Типы данных. Таблицы. Базы данных

Базы данных	
Концепции	Модель данных • Реляционная (модель • алгебра • нормальная форма • ссылочная целостность • БД • СУБД) • Иерархическая (модель • БД) • Сетевая (модель • СУБД) • Объектно-ориентированная (БД • СУБД) • Транзакция • Журнализация • Секционирование
Объекты	Отношение (таблица) • Представление • Хранимая процедура • Триггер • Курсор • Индекс
Ключи	Потенциальный • Первичный • Внешний • Естественный • Суррогатный (искусственный) • Суперключ
SQL	SELECT • INSERT • UPDATE • MERGE • DELETE • TRUNCATE • JOIN • UNION • INTERSECT • EXCEPT • CREATE • ALTER • DROP • GRANT • COMMIT • ROLLBACK
СУБД	IMS • DB2 • Informix • Oracle Database • Microsoft SQL Server • Adaptive Server Enterprise • Teradata Database • Firebird • PostgreSQL • MySQL • SQLite • Microsoft Access • Visual FoxPro • ЛИНТЕР • CouchDB • MongoDB
Компоненты	Язык запросов • Оптимизатор запросов • План выполнения запроса • ODBC • ADO • ADO.NET • JDBC

Подробнее см.

ВИКИПЕДИЯ
Свободная энциклопедия

Кортежи. Таблицы. Реляционные базы данных. SQL.
Конецкий М.В., Спиридонов В.В., Шумова Е.О. Базы данных. Основы SQL реляционных баз данных: Учебное пособие. - СПб.: СЗТУ, 2005. - 160 с.

Типы данных. Кодирование данных

Структурные типы данных

- Регулярные массивы
- Динамические списки и массивы переменной длинны
- Ациклические графы (деревья)
- Сетевые графы

Специализированные типы данных

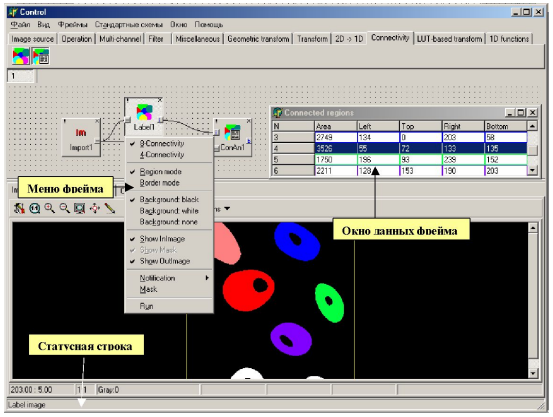
- Таблицы и базы данных
- Текст (символы и символьные строки)
- Аудиоданные
- Векторная и растровая графика

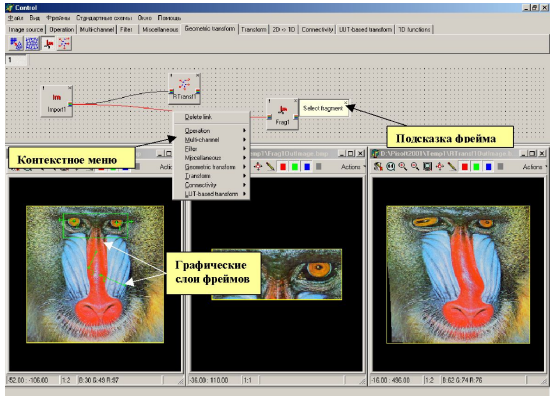
Кодирование данных - *преобразования, касающиеся не содержания, а формы представления данных:*

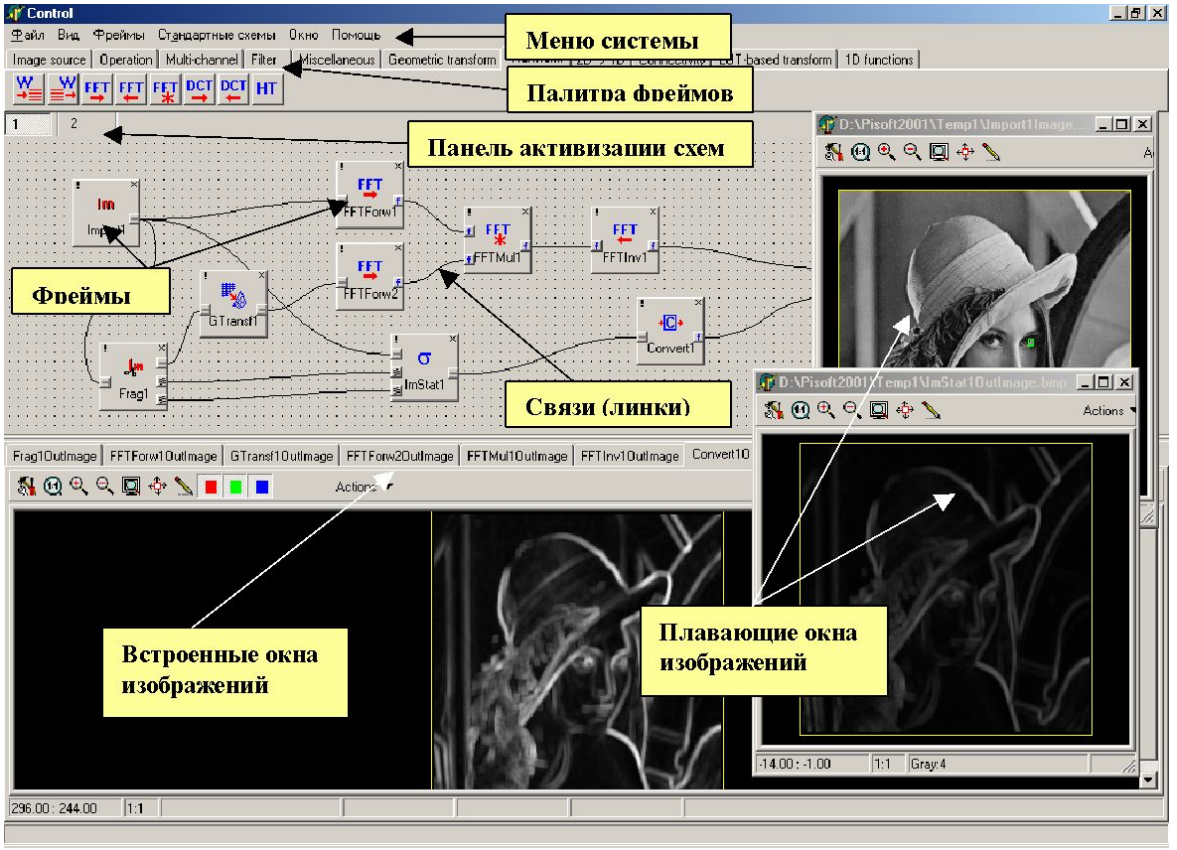
- Сжатие (хранение и передача данных)
- Помехоустойчивое кодирование (передача данных)
- Криптография (защита данных)

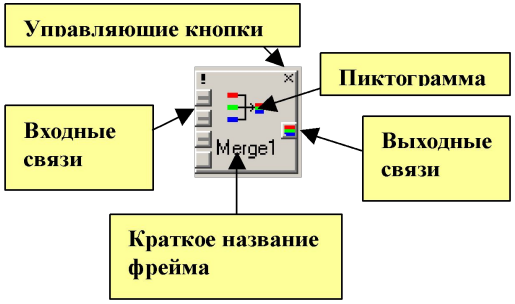
Подробнее см. ВИКИПЕДИЯ Свободная энциклопедия

Типы данных. Цифровое изображение









Лабораторная система обработки и анализа изображений **Pisoft Image Framework**

Рассмотрим с ее помощью типы данных, связанные с изображением

Компьютерное зрение как типовая область компьютерной обработки информации

- Специализированное зрение без "встроенных" алгоритмов.
- Требования к системам и алгоритмам технического зрения: *надежность, точность, быстродействие*.
- Основные задачи анализа изображений.
- Компьютерное зрение = "геометрия сегодня"

Компьютерное зрение как обобщение школьной геометрии

Геометрия Евклида	**Компьютерное зрение**
1. Задачи на построение.	Обнаружение объектов. Извлечение новых пространственных данных из исходных.
2. Задачи на доказательство.	Распознавание и идентификация объектов по их изображениям.
3. Циркуль и линейка	Компьютерная архитектура и язык программирования.
4. Построение за конечное число шагов	Ограничения по памяти и быстродействию.

Компьютерное зрение как школьная геометрия++

Отличия КЗ от классической геометрии:

1. Увеличение числа точек на порядки.

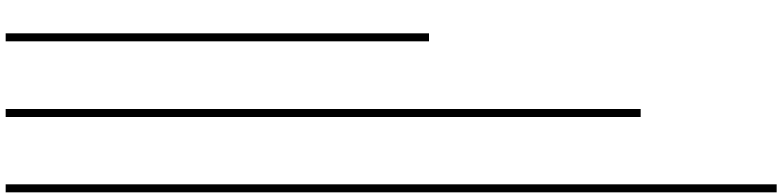
2. Появление у точек атрибутов интенсивности, цвета и других характеристик.

3. Рассмотрение не только правильных фигур, но и фигур любых других форм.

4. Учет всякого рода неидеальностей, погрешностей, шумов, искажений, дискретностей, неполной наблюдаемости и т.п. реальных факторов, которые, впрочем, также имеют математическое выражение.

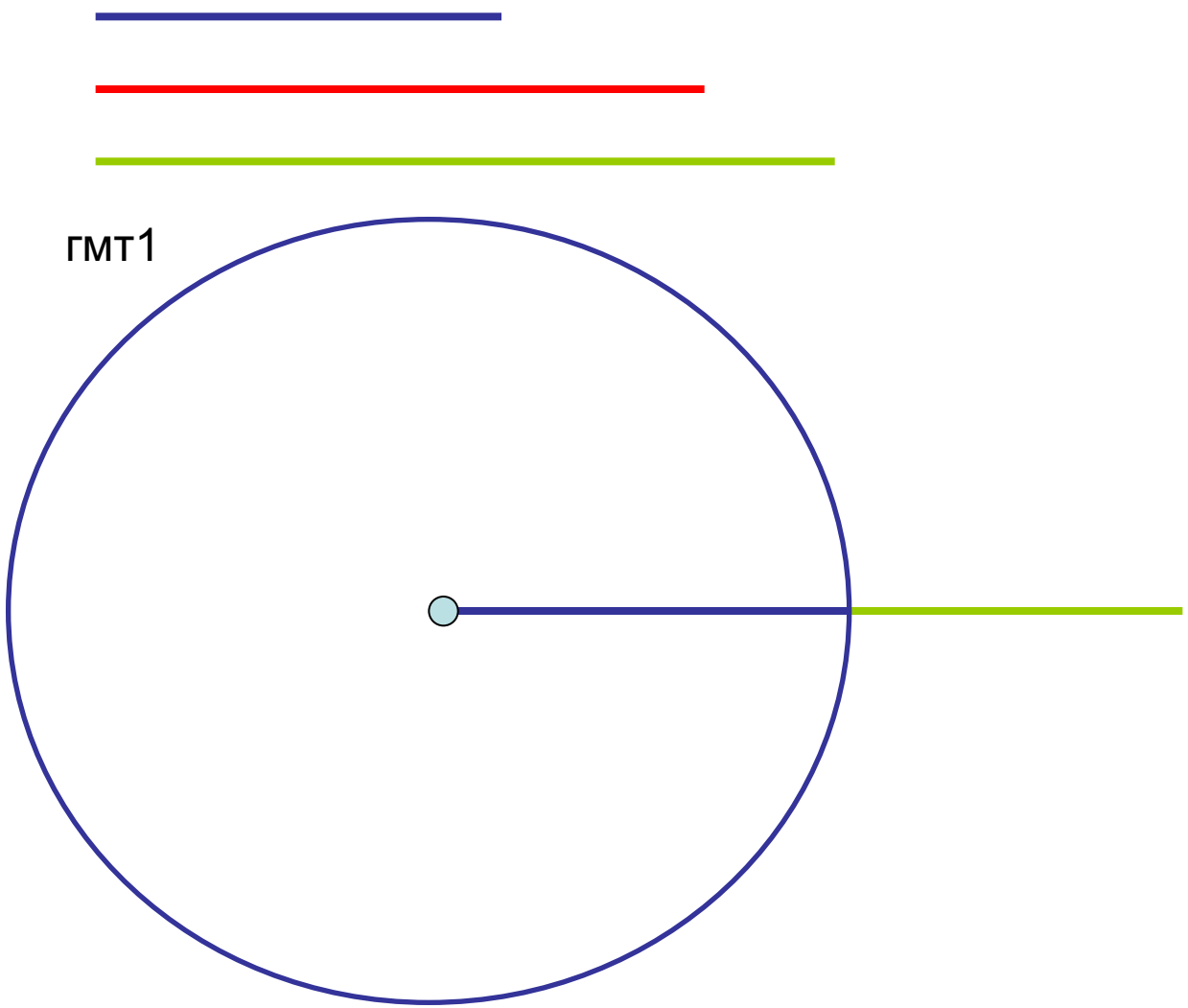
Метод общих геометрических мест

Задача 1: Построение треугольника по 3 заданным отрезкам.



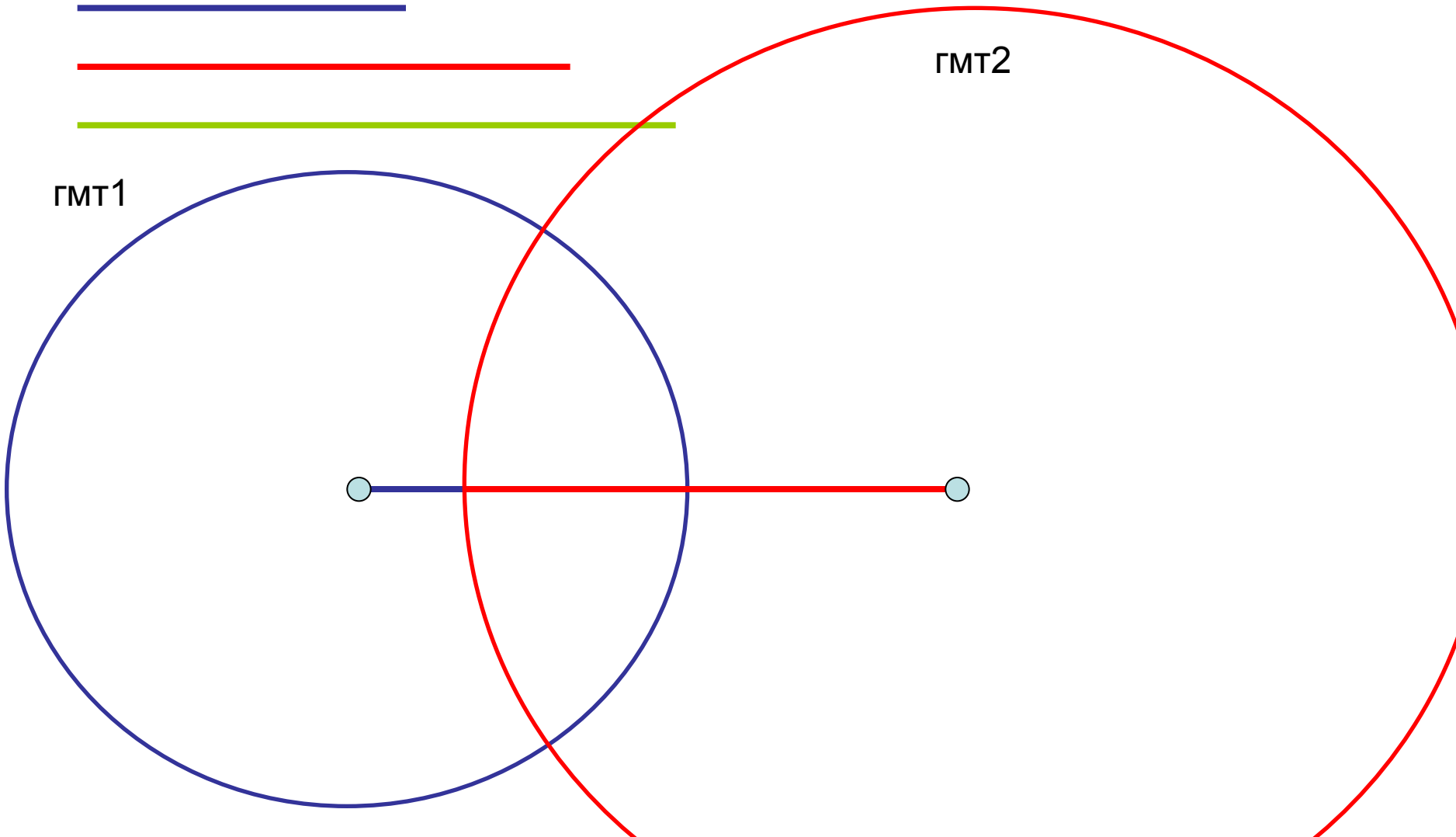
Метод общих геометрических мест

Задача 1: Построение треугольника по 3 заданным отрезкам.



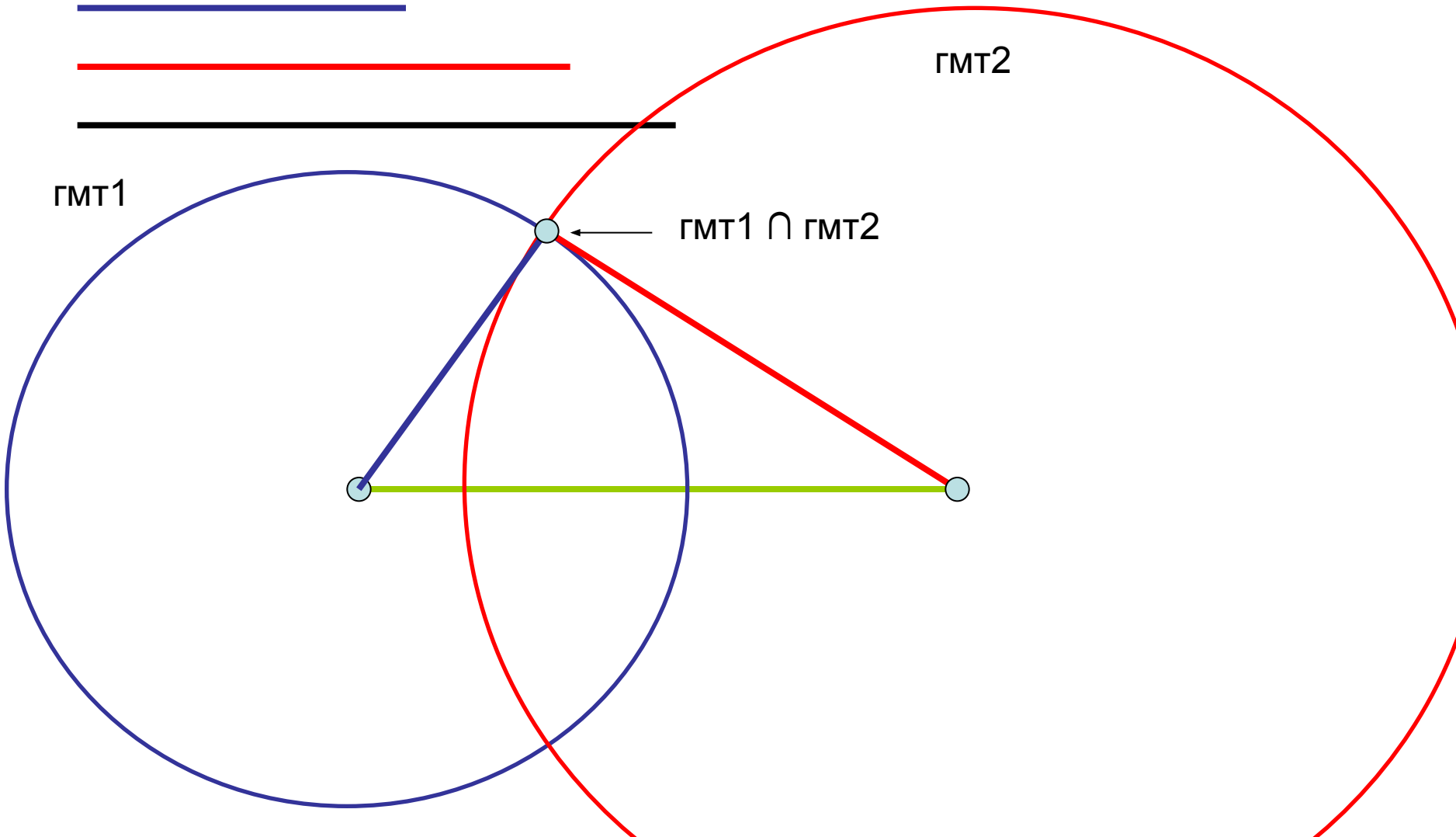
Метод общих геометрических мест

Задача 1: Построение треугольника по 3 заданным отрезкам.



Метод общих геометрических мест

Задача 1: Построение треугольника по 3 заданным отрезкам.

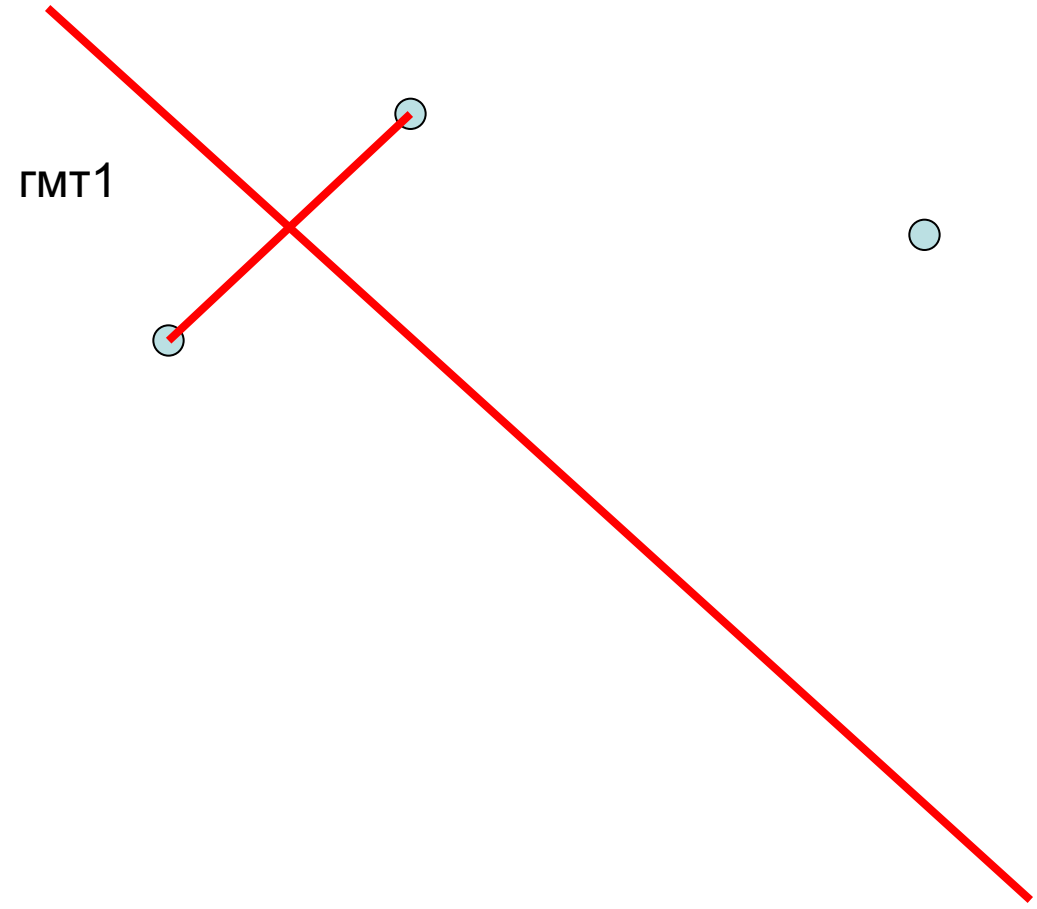


Метод общих геометрических мест

Задача 2: Построение окружности по 3 заданным точкам.

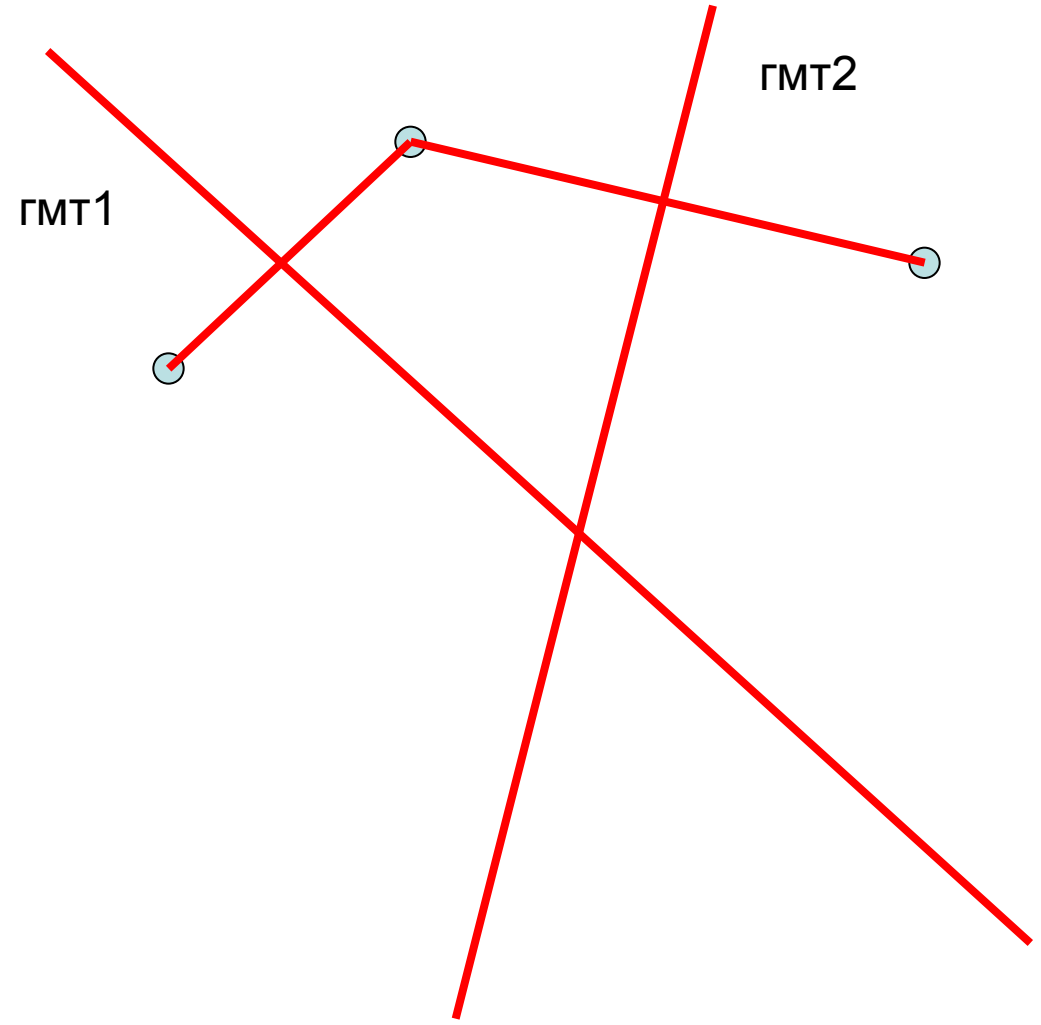
Метод общих геометрических мест

Задача 2: Построение окружности по 3 заданным точкам.



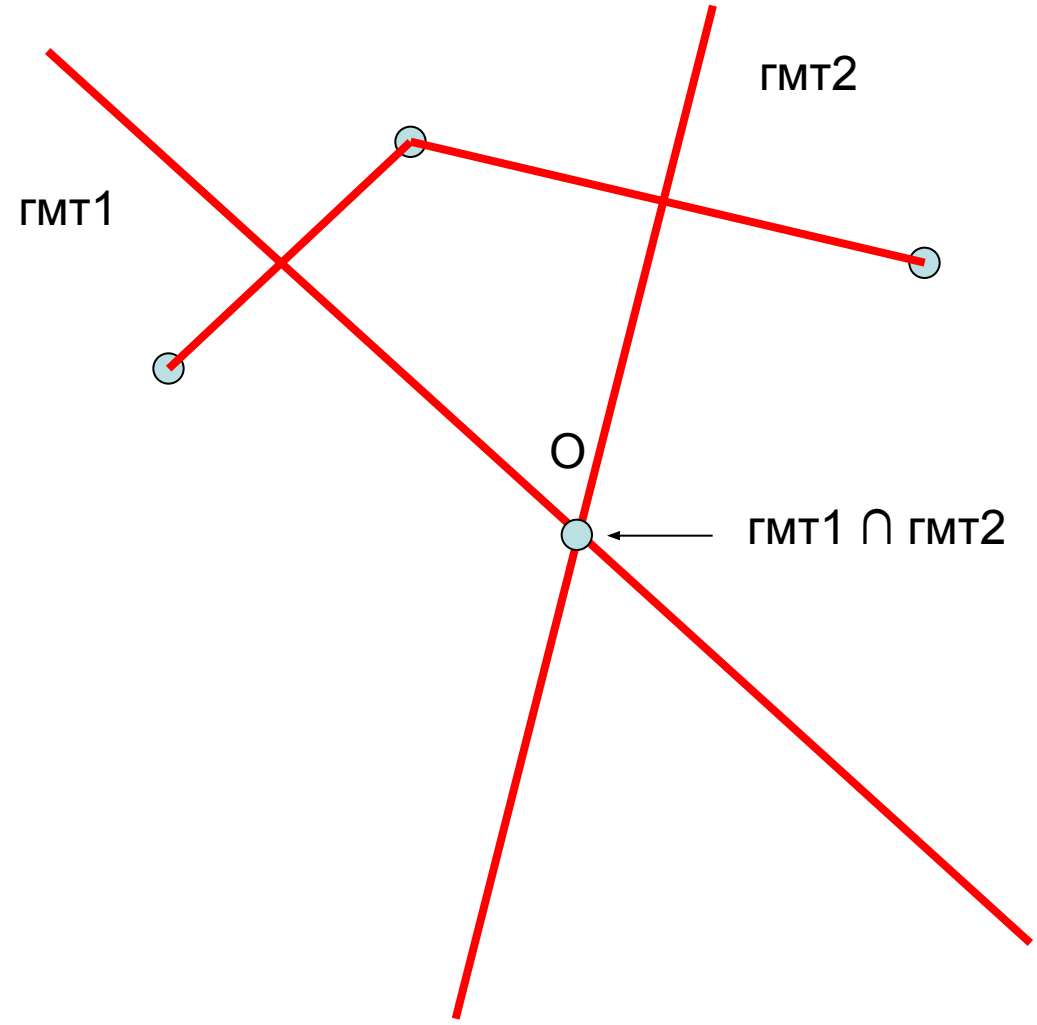
Метод общих геометрических мест

Задача 2: Построение окружности по 3 заданным точкам.



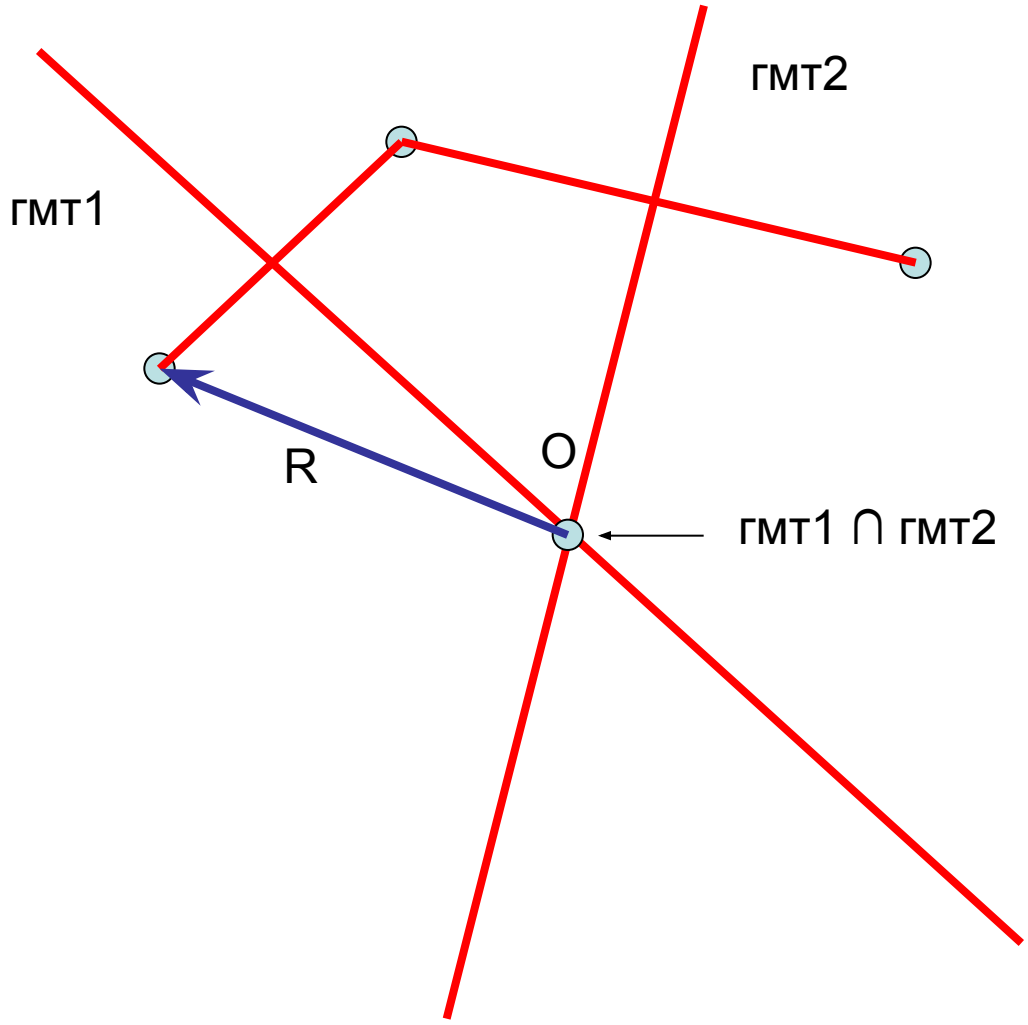
Метод общих геометрических мест

Задача 2: Построение окружности по 3 заданным точкам.



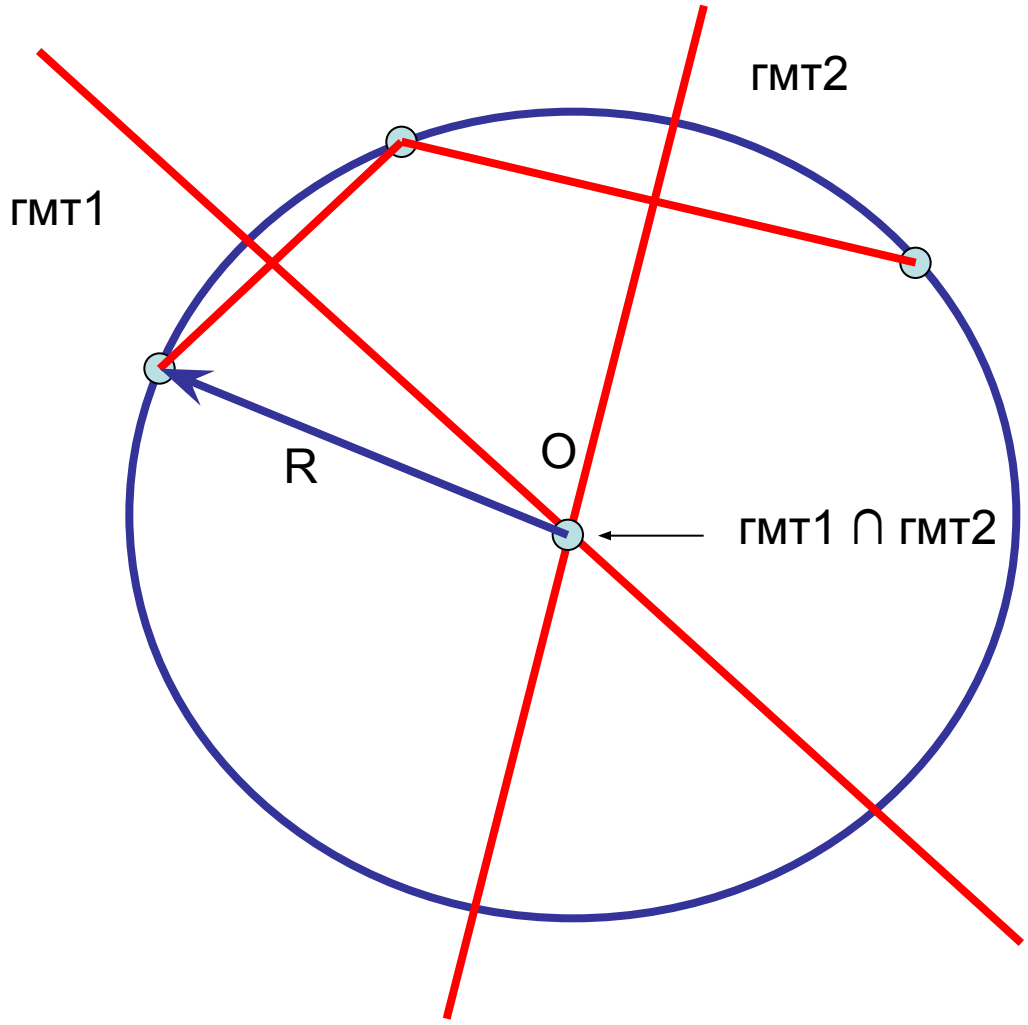
Метод общих геометрических мест

Задача 2: Построение окружности по 3 заданным точкам.



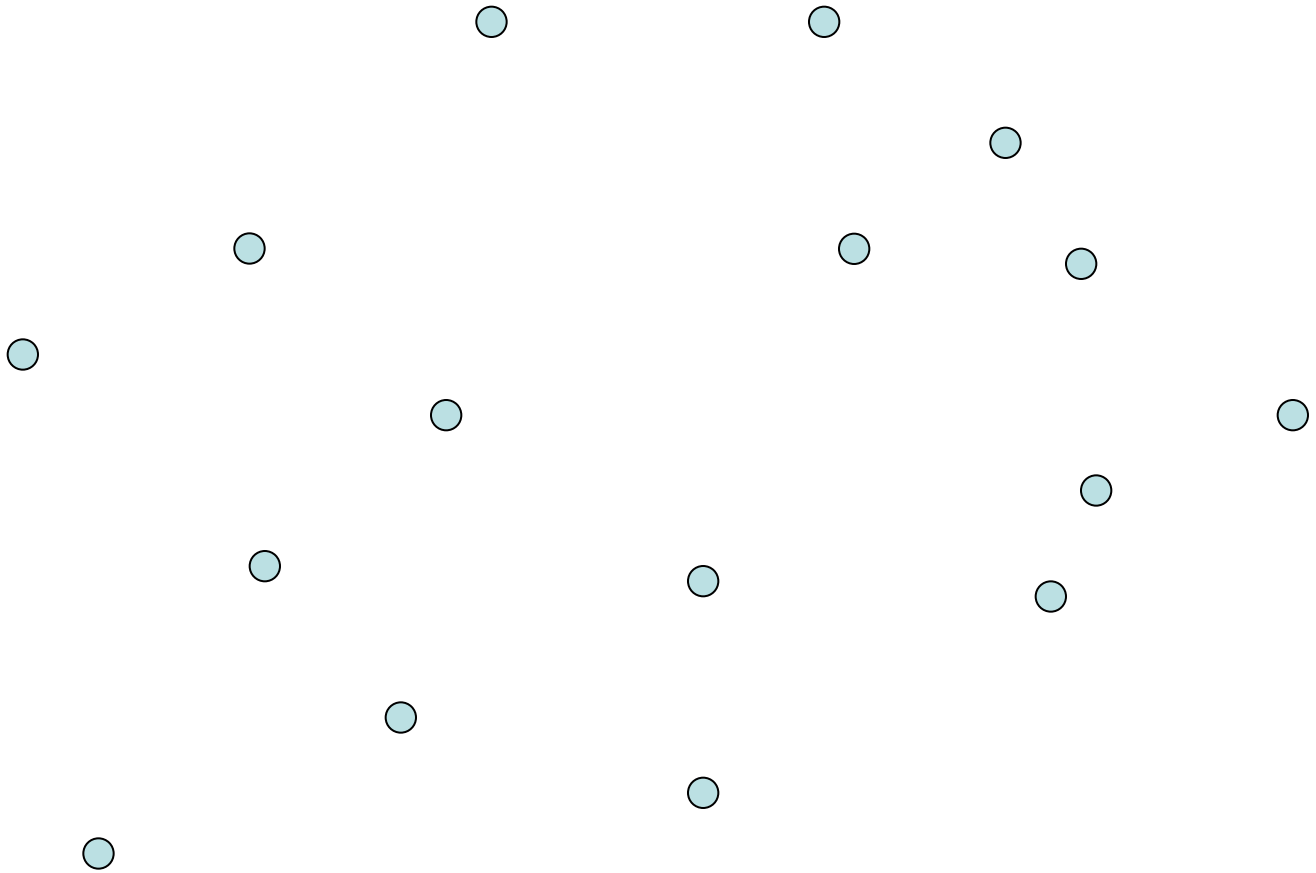
Метод общих геометрических мест

Задача 2: Построение окружности по 3 заданным точкам.



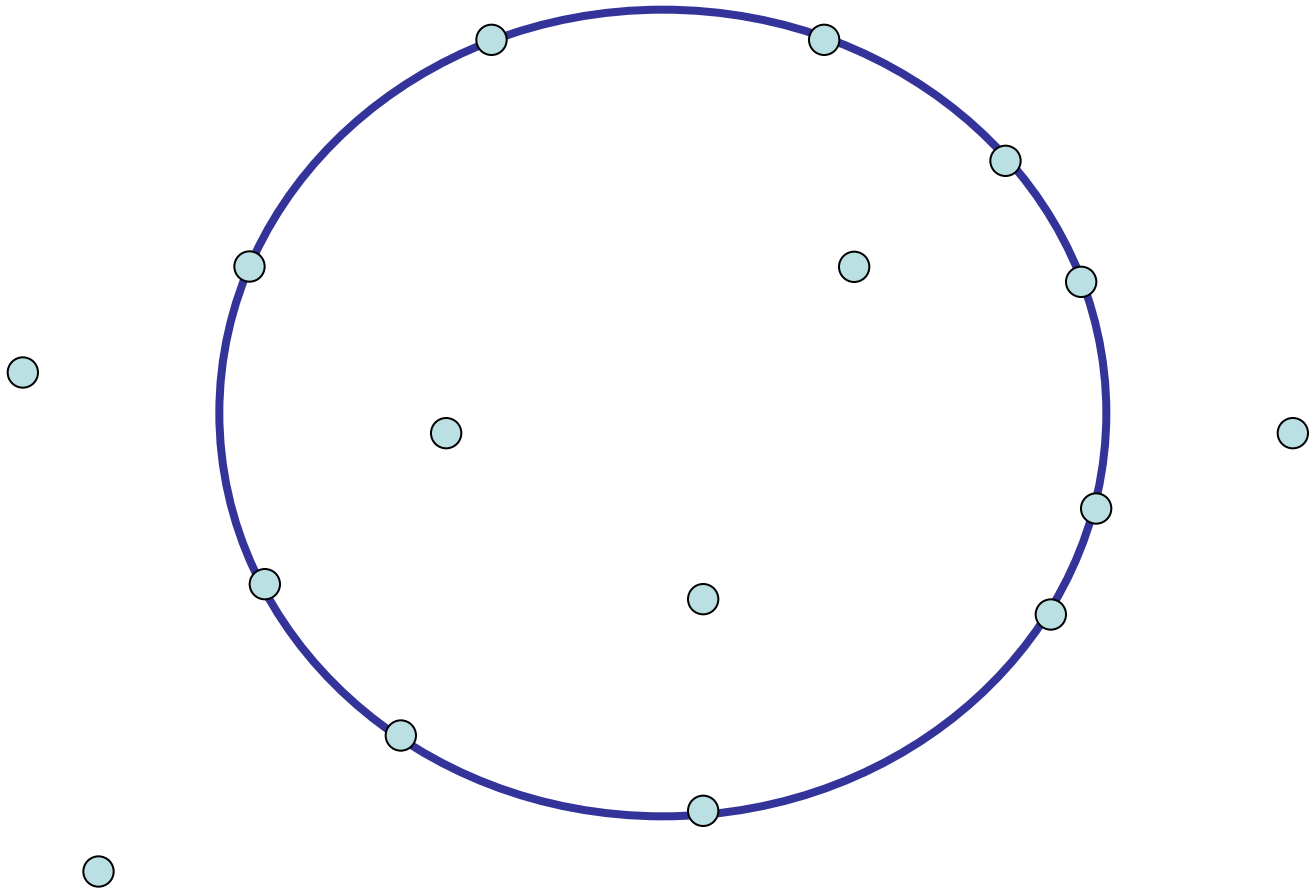
Метод общих геометрических мест

Задача 3: Построение окружности по N заданным точкам



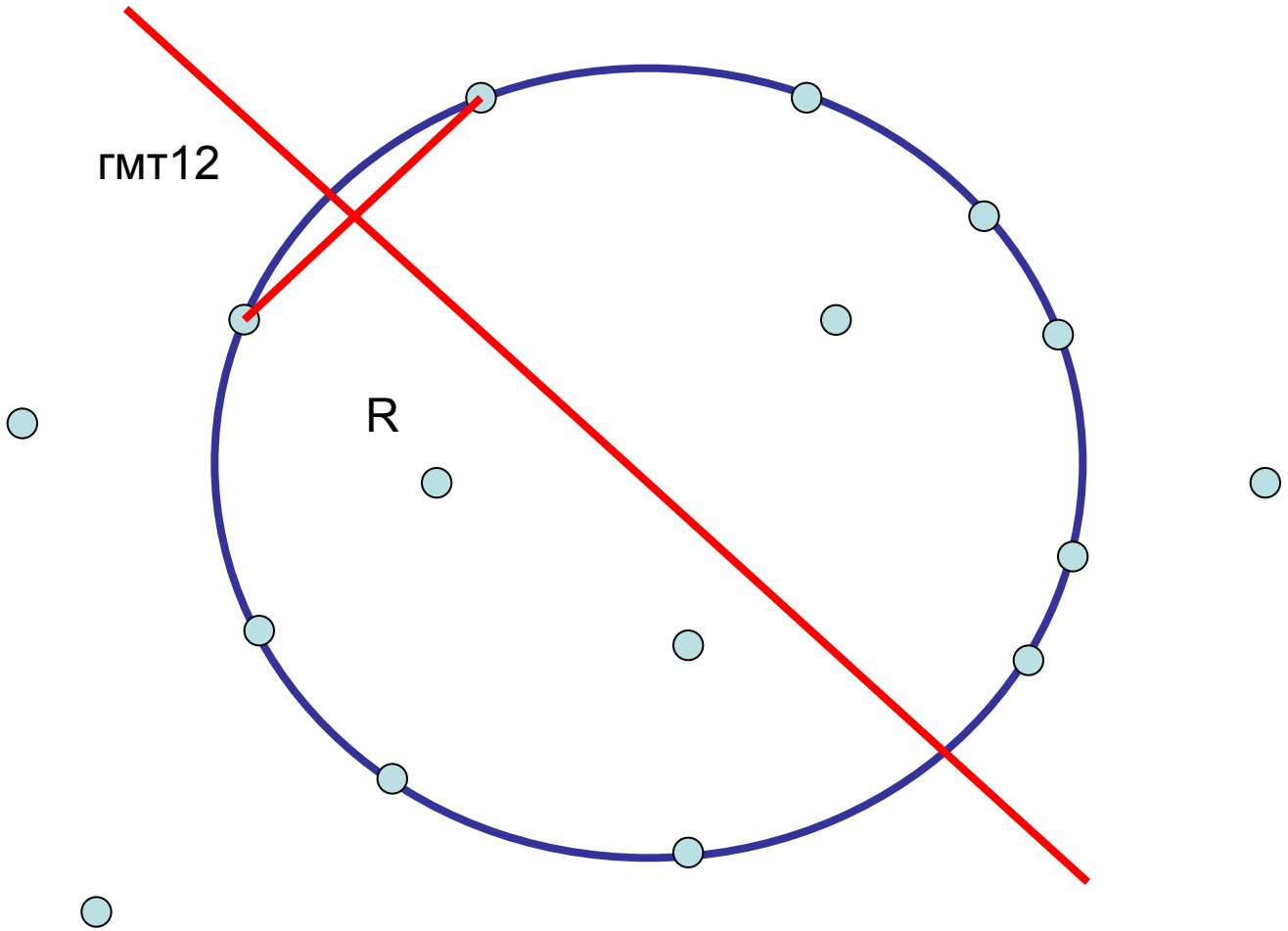
Метод общих геометрических мест

Задача 3: Построение окружности по N заданным точкам



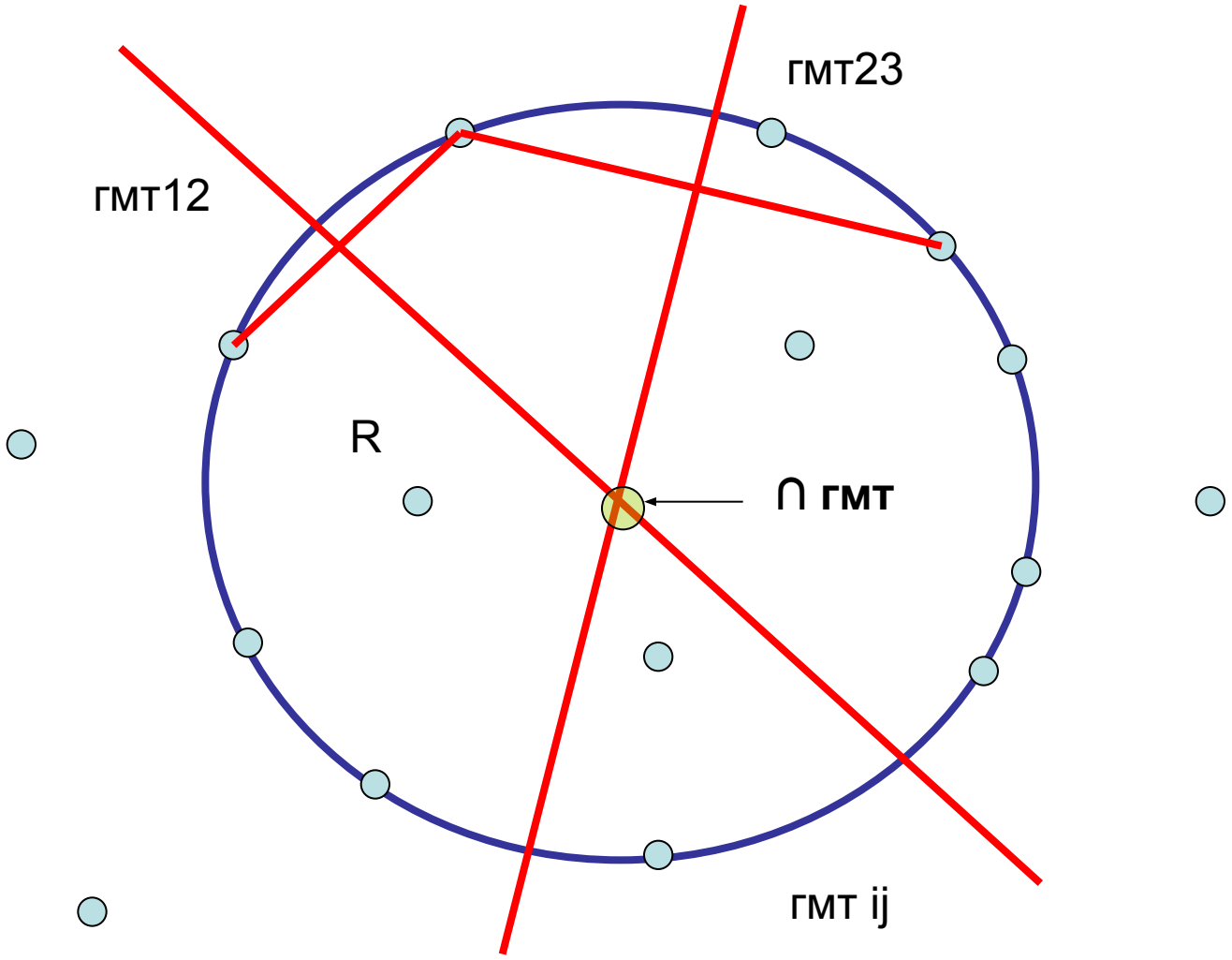
Метод общих геометрических мест

Задача 3: Построение окружности по N заданным точкам



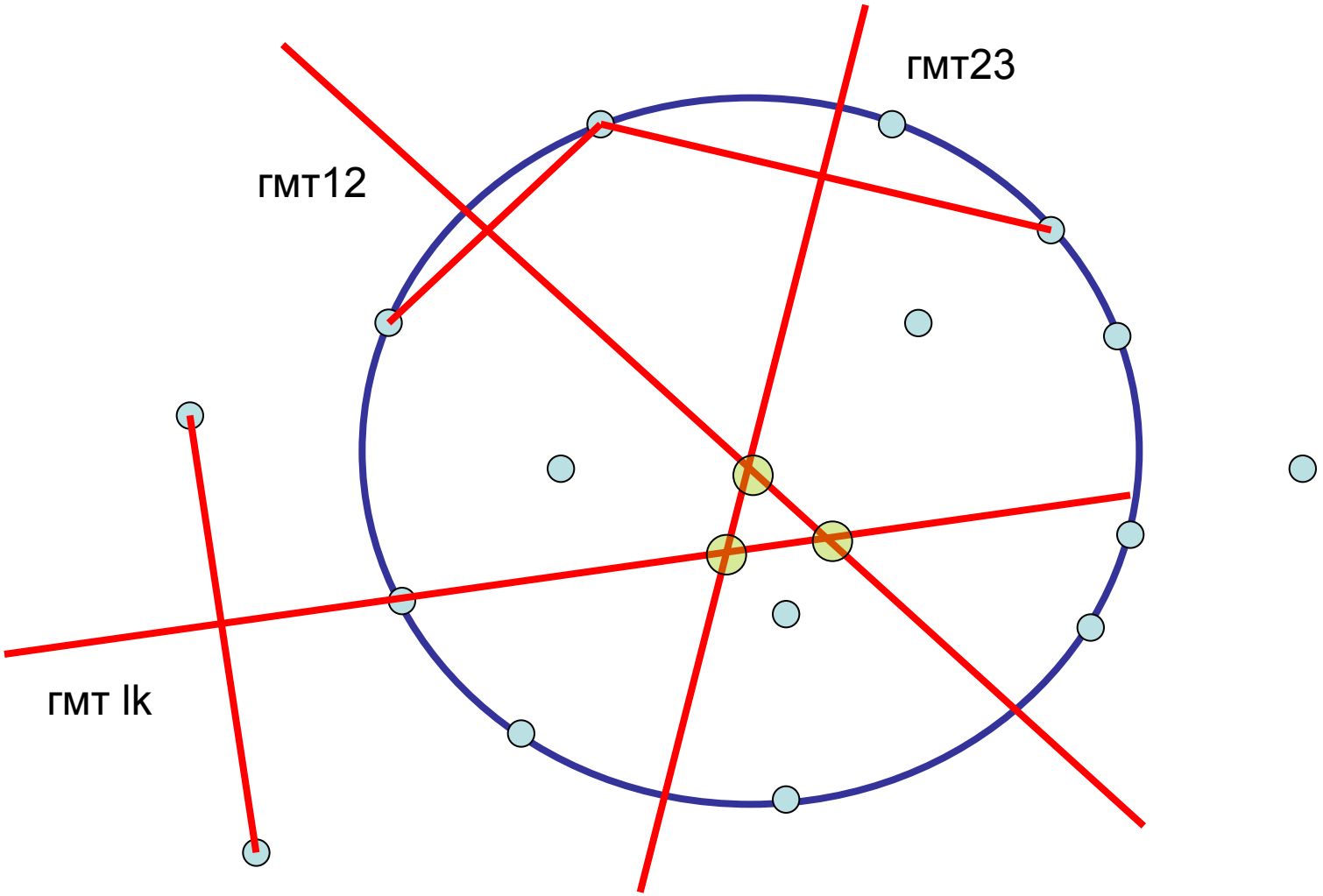
Метод общих геометрических мест

Задача 3: Построение окружности по N заданным точкам



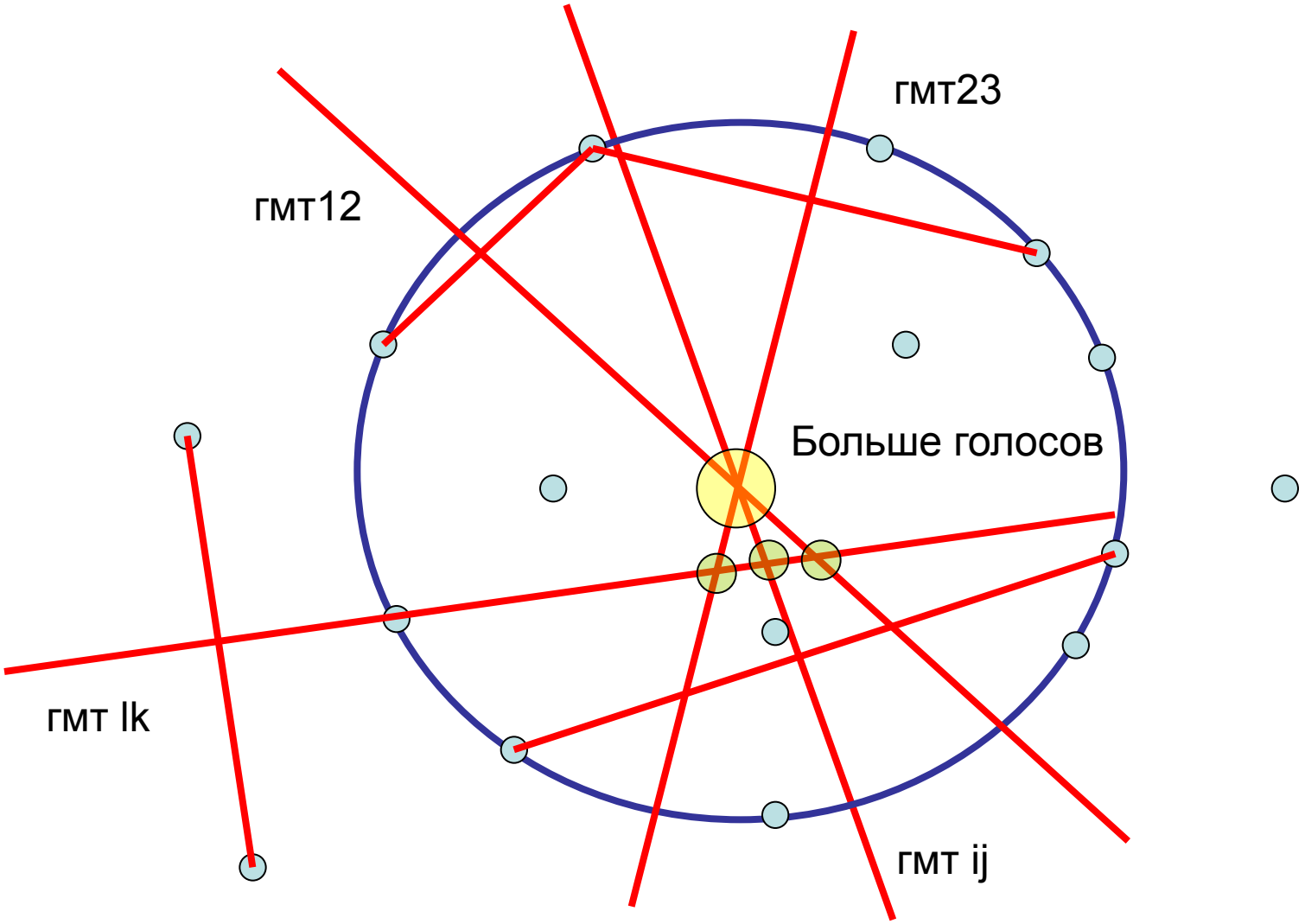
Метод общих геометрических мест

Задача 3: Построение окружности по N заданным точкам



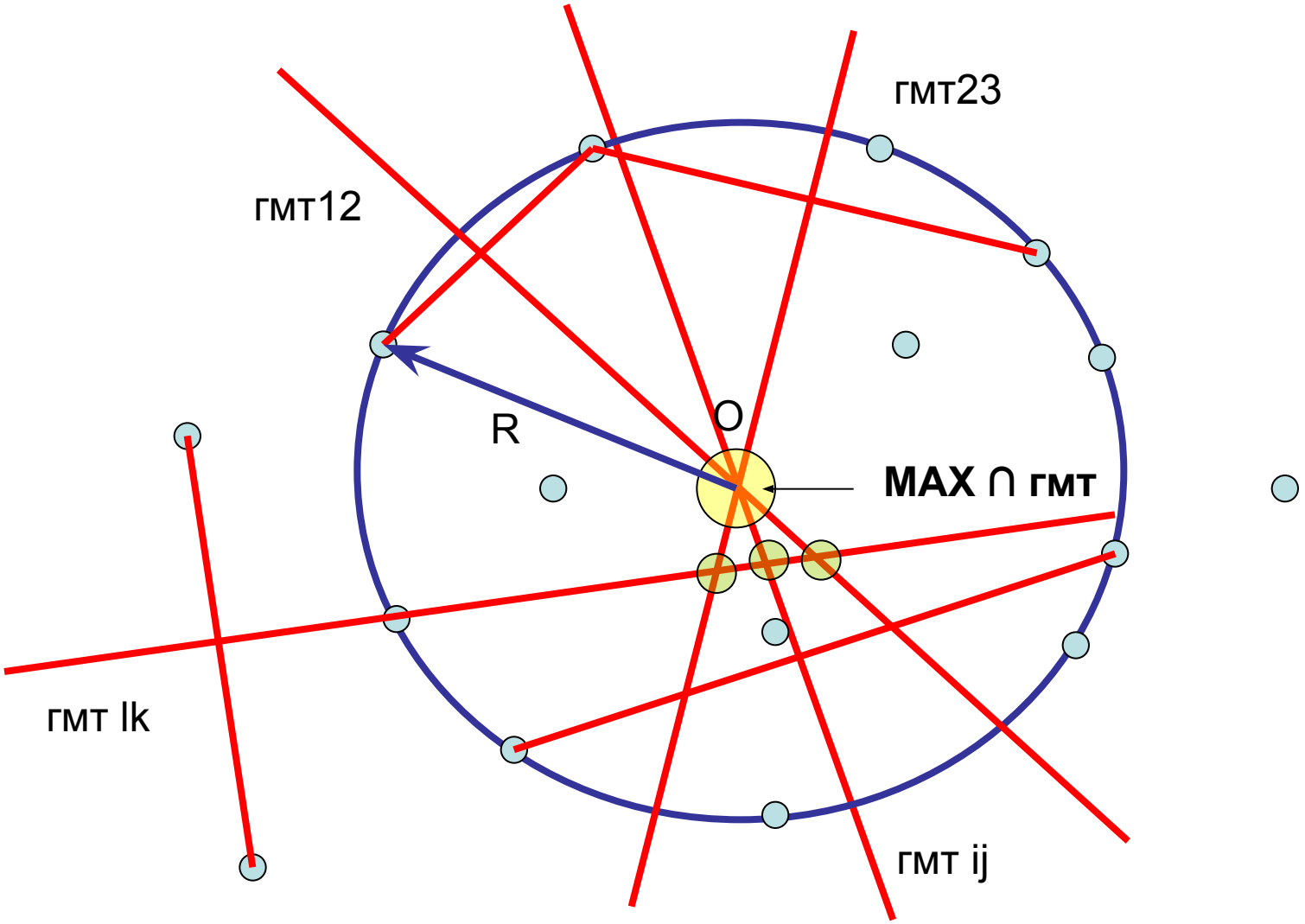
Метод общих геометрических мест

Задача 3: Построение окружности по N заданным точкам



Метод общих геометрических мест

Задача 3: Построение окружности по N заданным точкам



Области применения и типовые примеры практических систем компьютерного и машинного зрения.

Примеры приложений

Задачи и методы компьютерной обработки информации

Содержание учебного курса

Возможные темы курсовых

Программа учебного курса