Понятие об алгоритме. Свойства алгоритмов. Способы записи алгоритмов.

вперед

Оглавление:

Основные этапы решения задач на компьютере:

Этот процесс можно представить в виде нескольких последовательных этапов:

1. Постановка задачи,
2. Математическое или информационное моделирование,
3. Алгоритмизация задачи,
4. Программирование,
5. Ввод программы и исходных данных в ЭВМ,
6. Тестирование и отладка программ,
7. Исполнение отлаженной программы и анализ результатов.

Понятие алгоритма. Свойства алгоритма.

Алгоритм это точное предписание,которое определяет процесс, ведущий от исходных данных к требуемому конечному результату. Алгоритм должен быть понятен *исполнителю*(компьютеру, роботу и пр.)

Любой применимый алгоритм обладает следующими **основными свойствами:**

- Результативностью
- Определенностью
- Массовостью.

Результативность означает возможность получения результата после выполнения конечного количества операций. **Определенность** состоит в совпадении получаемых результатов независимо от пользователя и применяемых технических средств. **Массовость** заключается в возможности применения алгоритма к целому классу однотипных задач.

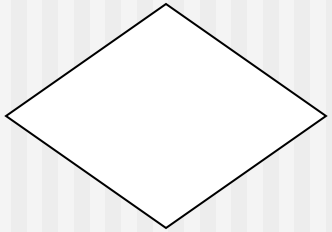
Форма представления алгоритма

Форма представления алгоритма может быть разной: словесно-формульная,структурная – блок-схема,граф-схема и др.

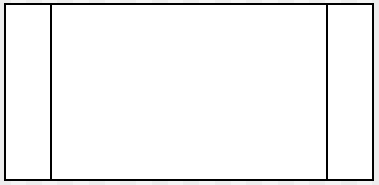
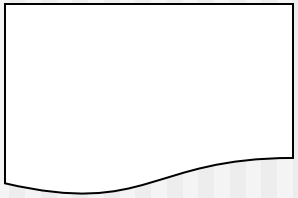
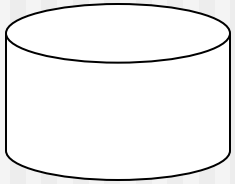
Блок-схемный алгоритм изображается геометрическими фигурами (блоками), связанными по управлению линиями (направлениями потока)со стрелками. В блоках записывается последовательность действий.

Блок-схема дает более наглядное представление алгоритма: каждая операция представлена отдельной геометрической фигурой. Кроме того графическое изображение наглядно показывает разветвление путей решения задачи в зависимости от различных условий, повторение отдельных этапов процесса и другие детали.

Условные обозначения блоков схем алгоритма

Наименование	Обозначение	Функции
Процесс		Выполнение операций или группы операций, в результате которых изменяется значение, форма представления или расположение данных.
Ввод-вывод		Преобразование данных в форму, пригодную для обработки (ввод) или отображения результатов обработки (вывод).
Решение		Выбор направления выполнения алгоритма в зависимости от некоторых переменных условий.

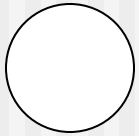
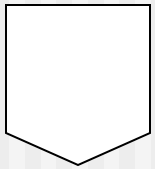
Условные обозначения блоков схем алгоритма (2)

Наименование	Обозначение	Функции
Предопределенный процесс		Использование ранее созданных и отдельно написанных программ (подпрограмм).
Документ		Вывод данных на бумажный носитель
Магнитный диск		Ввод-вывод данных, носителем которых служит магнитный диск.

назад

вперед

Условные обозначения блоков схем алгоритма (3)

Наименование	Обозначение	Функции
Пуск-останов		Начало, конец, прерывание процесса обработки данных.
Соединитель		Указание связи между прерванными линиями, соединяющими блоки.
Межстраничный соединитель		Ввод-вывод данных, носителем которых служит магнитный диск.
Комментарий		Связь между элементом схемы и пояснением.

назад

вперед

Пример 1.

Решение квадратного уравнения

Исходные данные:

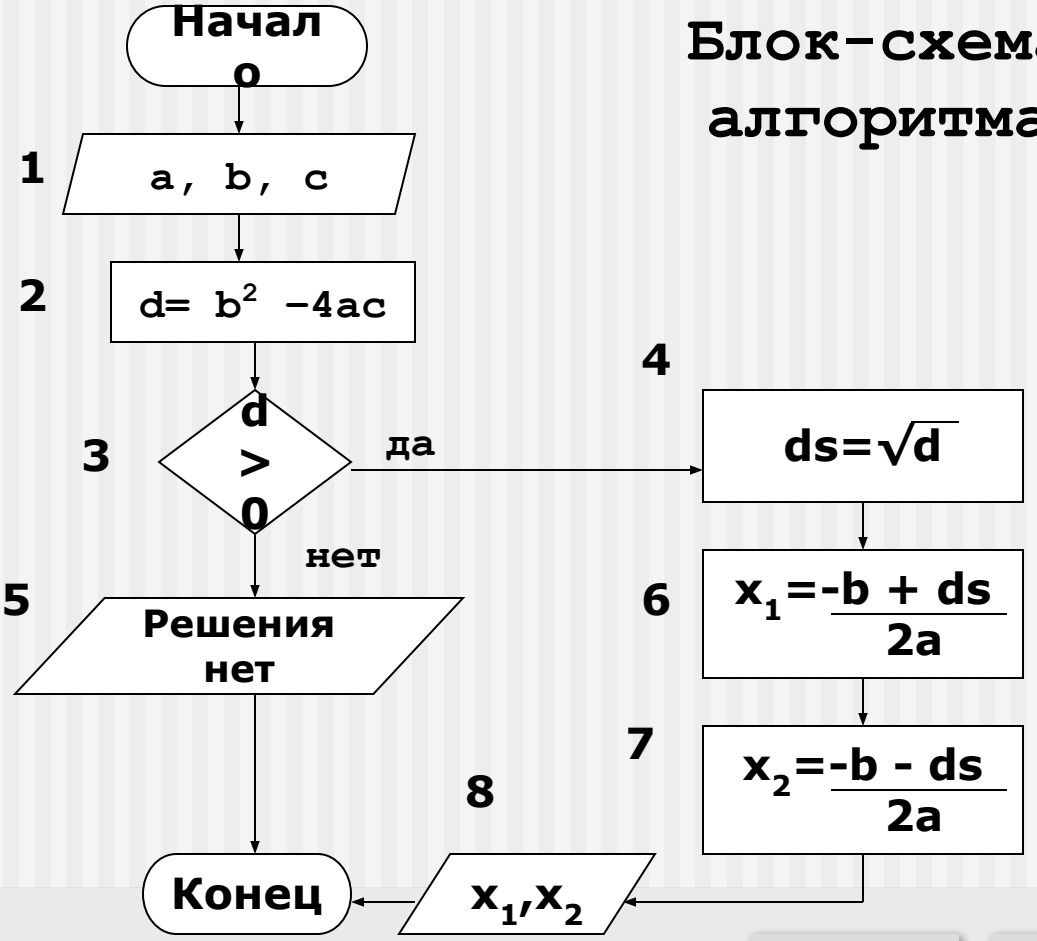
Значения переменных $a>0, b>0, c>0$

Задача:

Вычислить действительные корни уравнения

Блок 1- исходные данные, **блок 2**–вычисляется дискриминанта, **блок 3** – проверка значения d. Если оно меньше 0 , то **«Решения нет»**, если больше 0, то вычисляются квадратный корень, значения корней, **блок 8** – результат выводится на экран

Блок-схема алгоритма



назад вперед

Структурные схемы алгоритмов

Одним из свойств алгоритма является **дискретность** – возможность расчленения процесса вычислений, предписанных алгоритмом, на отдельные этапы, возможность деления участков программы с определенной структурой. Можно выделить и наглядно представить графически три простейшие структуры:

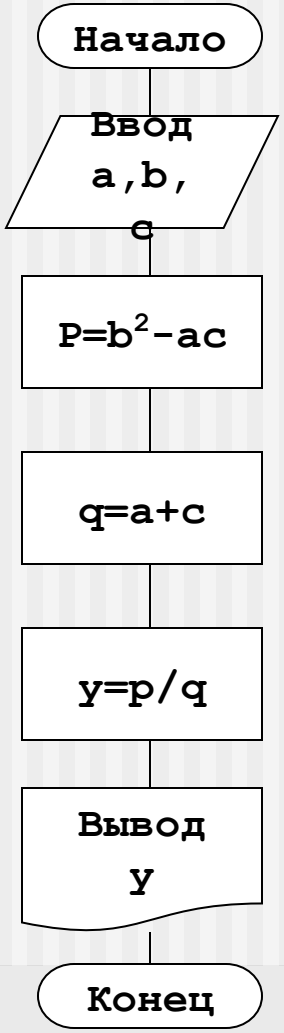
- Последовательность двух или более операций,
- Выбор направления,
- Повторение.

Любой вычислительный процесс может быть представлен как комбинация этих элементарных алгоритмических структур. Соответственно, вычислительные процессы, выполняемые на ЭВМ по заданной программе, можно разделить на три основных вида:

- Линейные
- Ветвящиеся
- Циклические

Линейные алгоритмы

Линейным принято называть процесс, в котором операции выполняются последовательно, в порядке их записи. Каждая операция является самостоятельной, независимой от каких-либо условий.



назад

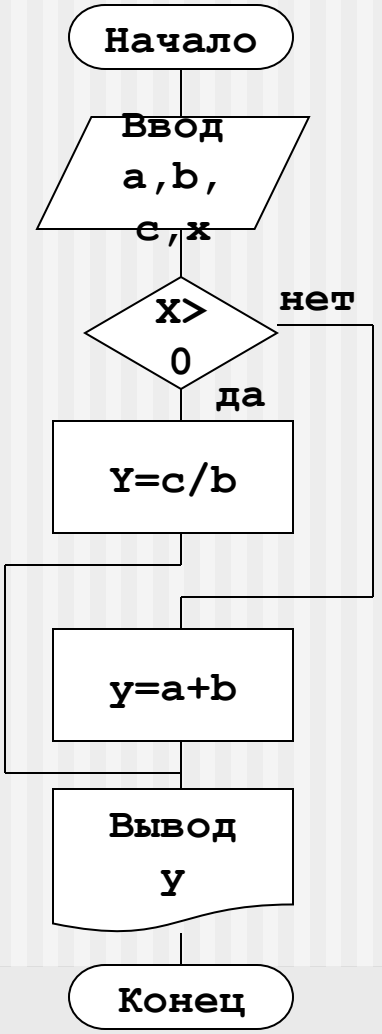
вперед

Ветвящиеся алгоритмы

Ветвящимися называется процесс, если для его реализации предусмотрено несколько направлений (ветвей). Каждое отдельное направление является отдельной ветвью вычислений. Направление ветвления выбирается логической проверкой, в результате которой возможно два ответа: «да» - условие выполнено, «нет» - условие не выполнено.

Пример ветвящегося алгоритма, определяющего процесс

$$y = \begin{cases} a+b, & \text{если } x \le 0, \\ c/b, & \text{если } x > 0 \end{cases}$$



назад

вперед

Циклические алгоритмы

Циклическими называются программы, содержащие циклы.

Цикл – это многократно повторяемый участок программы.

В организации цикла можно выделить следующие этапы:

1. Подготовка цикла,
2. Выполнение вычислений цикла (тело цикла),
3. Модификация параметров,
4. Проверка условий окончания цикла.

Цикл называется **детерминированным**, если число повторений тела цикла заранее известно или определено.

Цикл называется **итерационным**, если число повторений заранее неизвестно, а зависит от значений параметров, участвующих в вычислениях.

Циклические алгоритмы

Пример нахождения суммы 10-ти чисел

Контрольные вопросы:

- Назовите этапы подготовки и решения задач на ЭВМ.
- Что такое алгоритм и какими свойствами он обладает?
- Укажите формы представления алгоритмов.
- Опишите структурные схемы алгоритмов.
- Составьте и запишите алгоритмы (в тетради):

1. Даны две простые дроби. Составить алгоритм получения дроби, являющейся результатом их деления.
2. Написать алгоритм вычисления по формуле:

$y=(1-x^2+2,5x^3+x^4)^2$, учитывая следующие ограничения:

Пользоваться можно только операциями сложения, вычитания и умножения; каждое выражение может содержать только одну арифметическую операцию.

Внимание! Выражение – запись, определяющая последовательность действий над величинами. Выражение может содержать константы, переменные, знаки операций, функции.

Литература:

- В.Б.Попов, Turbo-Pascal для школьников, стр.15-19,
- А.Д.Хомоненко, Основы современных компьютерных технологий, стр.36-46,
- Задачник-практикум под ред. И.Семакина, стр.204-208,
- Ю.Шафрин, Информационные технологии, стр.43-52.

возврат

Решение задач включает следующие ***основные этапы***, часть из которых осуществляется без участия ЭВМ.

1. ***Постановка задач:***
   - сбор информации о задаче;
   - формулировка условия задачи;
   - определение конечных целей;
   - описание данных.
2. ***Построение математической модели.***
3. ***Построение алгоритма:***
   - выбор формы записи алгоритма (блок-схема, табличная и др.);
   - запись алгоритма.
4. ***Программирование:***
   - выбор языка программирования;
   - выбор способа представления данных;
   - запись алгоритма на выбранном языке;
   - выбор тестов и методов тестирования.
5. ***Тестирование:***
   - проверка работоспособности программы.

6. Отладка:

- анализ результатов тестирования;
- устранение ошибок, совершенствование программы.

7. Сопровождение программы:

- доработка программы для решения конкретных задач;
- составление документации к использованию.

Тестирование устанавливает факт наличия ошибки.
Отладка выясняет её причину.

Программа-отладчик обеспечивает следующие возможности:

- пошаговое исполнение программы с остановкой после каждого оператора;
- просмотр в процессе отладки текущего значения любой переменной или нахождение значения любого выражения;
- установку в программе *контрольных точек*, в которых программа временно прекращает своё выполнение, так что можно оценить промежуточные результаты.

! При *отладке* важно помнить:

- лучше использовать простые тестовые данные;
- ошибки разделять и устранять поочерёдно, не вносить в программу сразу несколько изменений;
- не следует считать причиной ошибок транслятор .

Тест = совокупность данных для программы + точное описание предполагаемых результатов

Два *этапа* процесса тестирования:

- проверка в нормальных условиях;
- проверка в экстремальных условиях.

Характерные ошибки программирования:

Вид ошибки	*Пример*
Неправильная постановка задачи	Правильное решение неверно сформулированной задачи
Неверный алгоритм	Выбор алгоритма, приводящего к неточному или не эффективному решению задач

Вид ошибки	*Пример*
Логические ошибки	Неполный учет ситуаций, которые могут возникнуть
Семантические ошибки	Непонимание работы оператора
Синтаксические ошибки	Нарушение правил, установленных в данном языке программирования
Ошибки при выполнении операций	Несоответствие операции типу данных, слишком большие числа, деление на ноль, извлечение квадратного корня из отрицательного числа
Ошибки в данных	Неудачное определение возможного диапазона изменения данных
Опечатки	Перепутаны близкие по написанию символы, например, цифра 1 и буква I
Ошибки ввода/вывода	Неправильное задание типов данных

! ***Отсутствие сообщений редактора данной среды программирования о синтаксических ошибках является необходимым , но недостаточным условием, чтобы считать программу правильной.***

Примеры **синтаксических** ошибок:

- неправильная запись формата оператора;
- несогласованность скобок;
- пропуск разделителей.

Примеры **логических** ошибок:

- неверное указание ветви алгоритма после проверки некоторого условия;
- неполный учет возможных условий;
- пропуск в программе одного или более блоков алгоритма.

Примеры **ошибок в циклах**:

- неверное условие выполнения или окончания цикла;
- неверное задание счётчика цикла;
- бесконечный цикл.

Примеры **ошибок ввода/вывода** и **ошибок при работе с данными**:

- неправильное задание типа данных;
- организация считывания меньшего или большего объема данных, чем требуется.

Примеры **ошибок в использовании переменных**:

- повторное использование имени переменной для обозначения другой величины;
- ошибочное использование одной переменной вместо другой.

Примеры **ошибок при работе с массивами**:

- неправильно задан размер массива;
- неправильно задан тип массива.

Примеры **ошибок в арифметических операциях**:

- неверное указание типа переменной (например, целого вместо
- вещественного);
- неверное определение порядка действий;

- деление на нуль;
- извлечение квадратного корня из отрицательного числа;

Эти ошибки обнаруживаются с помощью тестирования.

!

Программа, предназначенная для длительной эксплуатации, должна иметь соответствующую документацию и инструкцию по ее использованию (т.н. Help).

Выбор методов решения задач

Чтобы решить задачу необходима точная постановка задачи и правильный выбор метода решения задачи. Постановка задачи сводится, как правило, к математической форме описания условий задачи по схеме:

Задача (словесное описание).
Дано (перечисление исходного).
Требуется (перечисление требуемого).
Связь (зависимость между исходным и требуемым).
При (условия допустимости исходного).

Выбор метода решения должен обеспечить получение требуемых результатов для любых допустимых исходных данных.

Методы решения задач:

- рекуррентный;
- рекурсивный;
- приближённые методы.

Рекуррентный метод

Метод, использующий рекуррентные соотношения, называется **рекуррентным**.

Формулы, в которых очередной член последовательности выражается через один или несколько предыдущих членов, называются **рекуррентными соотношениями.**

В программировании часто используют метод рекуррентного суммирования для определения суммы конечной последовательности чисел.

Применим ранее описанную схему:

Задача (найти сумму конечной последовательности заданных чисел).
Дано (x_1, $x_2 \ldots x_n$ – последовательность n чисел).
Требуется (найти S – сумму этих чисел).
Связь ($S = x_1 + x_2 + \ldots + x_n$).
При (любых значениях чисел последовательности).

Применив рекуррентный метод, получим соотношения:

$S_0 = 0$

$S_k = S_{k-1} + x_k \qquad k = 1,2,\ldots n$

$S = S_n$

Примеры рекуррентных соотношений:

- нахождение членов арифметических и геометрических прогрессий;
- нахождение членов ряда Фибоначчи.

Итальянский математик XVI в. Фибоначчи построил математический ряд 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55,…, описывающий биологический процесс (процесс размножения кроликов). Попробуйте записать рекуррентное соотношение для этого ряда.

. . .

Легко заметить закономерность: член ряда равен сумме двух предыдущих членов. Если в таком ряду взять отношение последующего члена к предыдущему и наоборот, получим числа **1,618** и **0,618.** Эти числа были открыты «пифагорейцами» и получили название **«золотых»**.

Они являются корнями квадратного уравнения, которое можно вывести из следующей пропорции:

$$\frac{A+B}{A} = \frac{A}{B}$$

A и B – соответственно две части одного отрезка.

Числа эти не зря названы «золотыми». «Пифагорейцы» же заметили, что музыкальный звукоряд построен по закону соотношений частот, равных «золотому» числу. И везде, где человек ощущает гармонию: в звуках, живописи, размерах – всюду присутствуют «золотые» числа (числа Фибоначчи).

К изумлению Кеплера, создававшего свою модель солнечной системы и уже знавшего основные размеры планет, периоды их обращения и взаимные расстояния, эти числа оказались числами Фибоначчи. Оказалось, что весь ближний космос организован по законам музыкальной гармонии. (Всё это Кеплер изложил в своей книге «Музыка сфер».)

.

Даже сам человек создан по закону «золотого сечения». Так, дельта-ритмы мозга при реакции на внешний раздражитель – это затухающие апериодические колебания, соседние периоды которых относятся по закону «золотого сечения».

Если рост человека принять за весь отрезок, а пупок – точка деления, то от него вниз – часть А, а вверх часть – В. Для мужчин это соотношение в среднем 1,7, а для женщин 1,5, т.е. мужчины в среднем стройнее (не из интуитивного ли стремления соответствовать гармонии «золотого сечения» появились туфли на каблуках?).

В ряду Фибоначчи **чем больше порядковый номер членов ряда, тем точнее выполняется «золотое соотношение»**. Проверьте это.

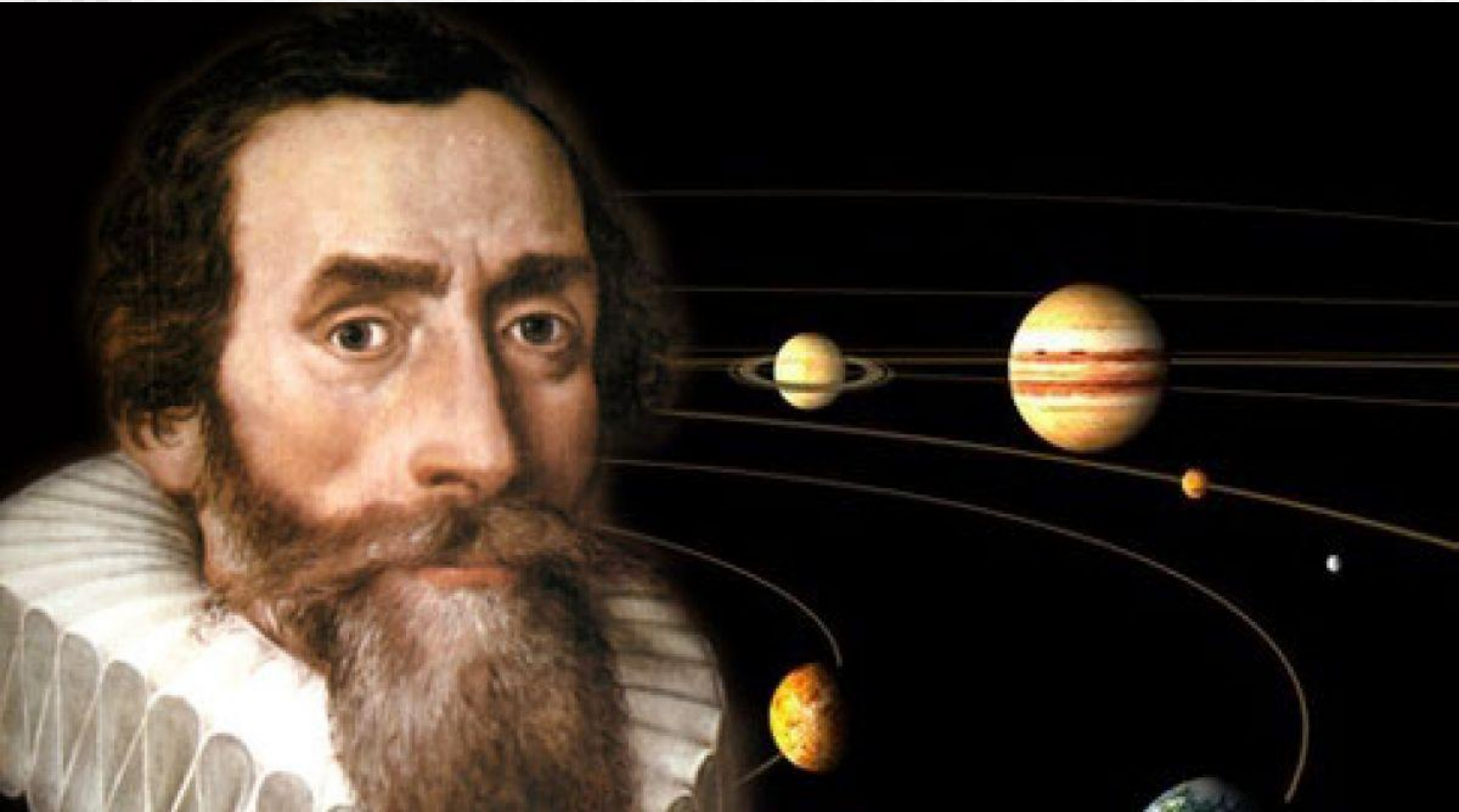
Великий итальянский ученый Фебоначчи

Астрономия до ЕвдоксаАстрономия до Евдокса не знала сферАстрономия до Евдокса не знала сфер. ПлатонАстрономия до Евдокса не знала сфер. Платон говорит о «кругах», Аристотель о «небесных светилах», или «звёздах». В латинской науке поздней античности и средневековья то же понятие передаётся чаще всего как «небесная гармония», «небесная

В небесной «гармонии» **8** разновысотных звуков**:** звёздное небо **(**высший тон**),** Сатурн**,** Юпитер**,** Марс**,** Меркурий**,** Венера**,** Солнце**,** Луна **(**низший тон**).**

В то время как средневековые философы использовали понятие «музыка сфер» лишь метафорически, Кеплер рассчитал математические соотношения в движении планет и увязал их с музыкальными интервалами, установив семь основных гармонических интервалов (консонансов): октаву (2/1), большую сексту (5/3), малую сексту (8/5), ...

Иоганн Кеплер



Иоганн Кеплер

«Геометрия имеет два великих сокровища; Одно — это теорема Пифагора; Другое — разделение линии на экстремальное и среднее соотношение.

Первое можно сравнить с мерой золота; Второе мы можем назвать драгоценным камнем».

Рекурсивный метод

Рекурсивным называется метод вычисления функций через самих себя.

Иногда значение функции для некоторых аргументов можно выразить через значение этой же функции от других аргументов.

Пример 1:

$$\sin(x+\pi) = -\sin(x)$$ для всех вещественных x.

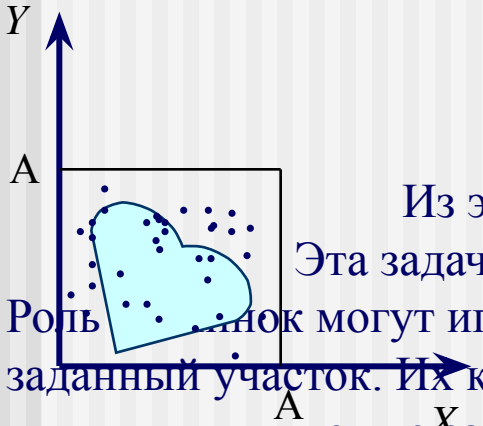
Пример 2:

$$n! = \begin{cases} 1, & \text{если } n = 1; \\ (n-1)! \times n, & \text{если } n > 1. \end{cases}$$

Метод Монте-Карло (приближённый метод)

Метод используется для вычисления площади фигуры, ограниченной произвольным контуром.

Пусть есть фигура на плоскости, и пусть она расположена внутри квадрата, сторона которого, A, известна. Площадь фигуры можно вычислить, засыпав квадрат слоем песка. Число песчинок внутри квадрата будет пропорционально его площади, а число песчинок внутри фигуры – пропорционально её площади. Из этой пропорции можно определить площадь фигуры.



Эта задача легко моделируется и программируется. Роль песчинок могут играть точки выводимые произвольно на заданный участок. Их координаты можно задавать с помощью генератора случайных чисел.