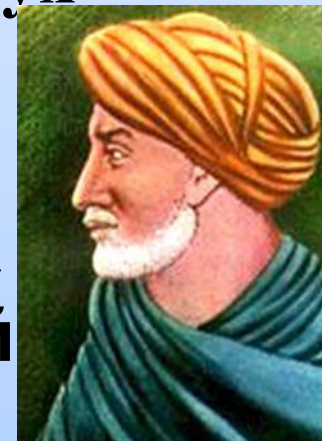




Исторический обзор

- III в. до н.э. - алгоритм нахождения НОД двух чисел (алгоритм Евклида).
- IX в. Муххамед бен Аль-Хорезми («аль-хорезми» - «алгоритм»), правила выполнения арифметических действий в десятичной с. с.
- XX в. – последовательность действий для решения различных задач



Теория алгоритмов - раздел математики, в котором изучаются теоретические возможности эффективных процедур вычисления (алгоритмов) и их приложения. Первые 2 вида математических понятий алгоритма:

- Описывается некоторый класс арифметических функций от конечного числа натуральных аргументов с натуральными значениями. Эти функции обладают некоторыми эффективными процедурами нахождения значения функции (если оно существует) по заданным значениям аргументов. Функции из этого класса называются частично рекурсивными (ч.р.ф.), а в случае, если ч.р.ф. всюду определены, их называют общерекурсивными.
- Описываются точные математические понятия машины и вычислимости на машине. Эти "теоретические вычислительные машины" называют машинами Тьюринга.

Различие между видами: в первом дается точное описание класса вычислимых арифметических функций, во втором - точное описание некоторого класса алгоритмических преобразований.

- 1931 г. Курта Гедель (теорема о неполноте символических логик): некоторые математические проблемы не могут быть решены алгоритмами из некоторого класса.
- 1936 г. А. Тьюринг, А. Черч и Э. Пост: машина Тьюринга, машина Поста и лямбда-исчисление Черча были эквивалентными формализмами алгоритма.
- В 1950-е годы существенный вклад в теорию алгоритмов внесли работы Колмогорова и Маркова.

Колмогоров: *Алгоритм* – это всякая система вычислений, выполняемых по строго определенным правилам, которая после какого-либо числа шагов заведомо приводит к решению поставленной задачи.

Марков: *Алгоритм* – это точное предписание, определяющее вычислительный процесс, идущий от варьируемых исходных данных к искомому результату.


Направления в теории алгоритмов (60-70 г.)

- **Классическая теория алгоритмов** (формулировка задач в терминах формальных языков, понятие задачи разрешения, введение сложностных классов, формулировка в 1965 году Эдмондсом проблемы $P=NP$, открытие класса NP -полных задач и его исследование);
- Теория асимптотического анализа алгоритмов (понятие сложности и трудоёмкости алгоритма, критерии оценки алгоритмов, методы получения асимптотических оценок, в частности для рекурсивных алгоритмов, асимптотический анализ трудоёмкости или времени выполнения): Кнут, Ахо, Хопкрофт, Ульман, Карп;
- Теория практического анализа вычислительных алгоритмов (получение явных функции трудоёмкости, интервальный анализ функций, практические критерии качества алгоритмов, методика выбора рациональных алгоритмов): фундаментальный труд Д. Кнута «Искусство программирования для ЭВМ».

Разделы современной теории алгоритмов



Цели и задачи теории алгоритмов

- 
- формализация понятия «алгоритм» и исследование формальных алгоритмических систем;
 - формальное доказательство алгоритмической неразрешимости ряда задач;*
 - классификация задач, определение и исследование сложностных классов;
 - асимптотический анализ сложности алгоритмов;
 - исследование и анализ рекурсивных алгоритмов;
 - получение явных функций трудоемкости в целях сравнительного анализа алгоритмов;
 - разработка критериев сравнительной оценки качества алгоритмов.

Требования к алгоритму

- алгоритм должен содержать *конечное* количество элементарно выполнимых предписаний, т.е. удовлетворять требованию конечности записи;
- алгоритм должен выполнять конечное количество шагов при решении задачи, т.е. удовлетворять требованию конечности действий;
- алгоритм должен быть единым для всех допустимых исходных данных, т.е. удовлетворять требованию универсальности;
- алгоритм должен приводить к правильному по отношению к поставленной задаче решению, т.е. удовлетворять требованию правильности.

Машина Тьюринга



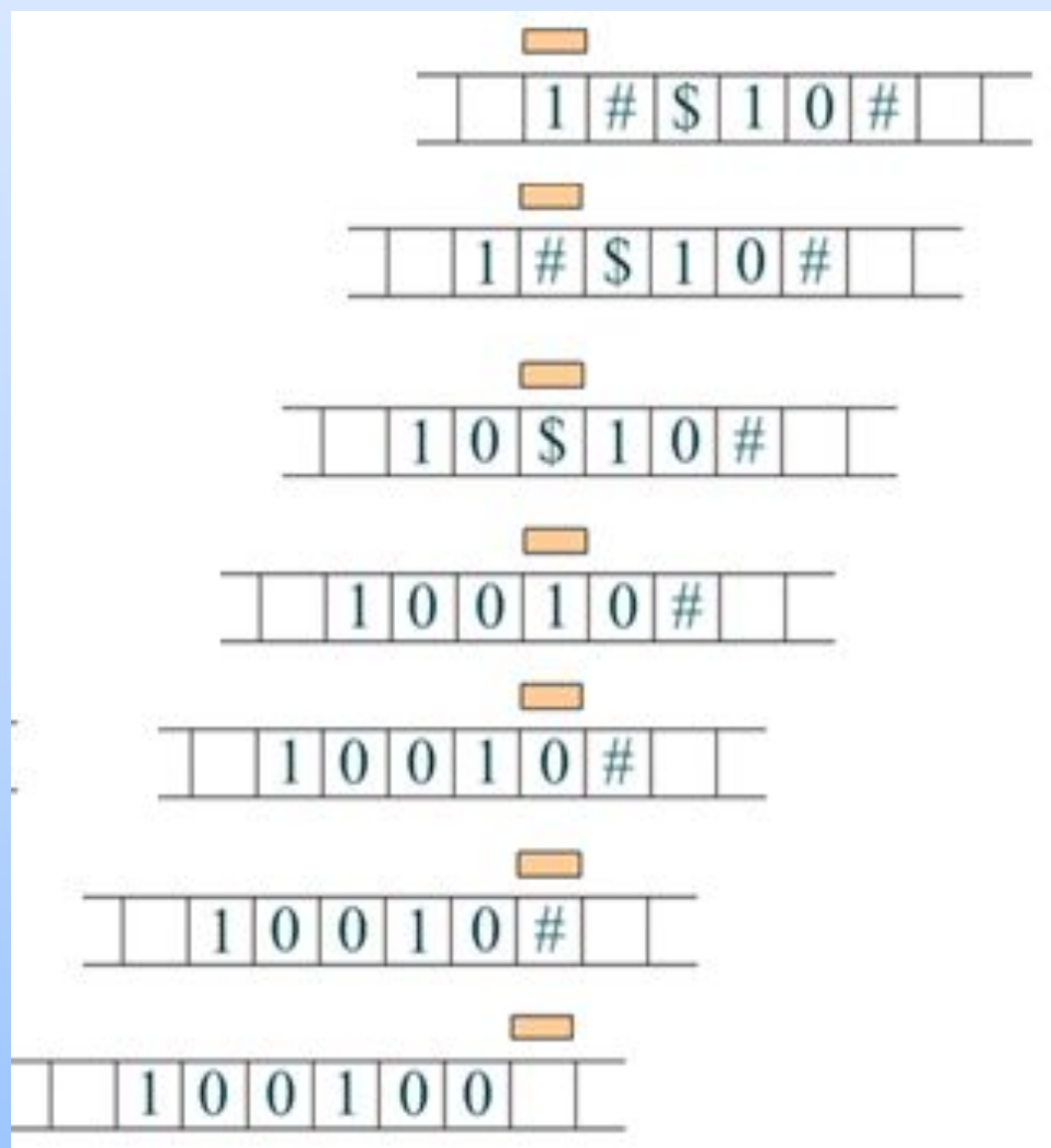
Бесконечная в обе стороны лента, разделенная на ячейки. Лента может протягиваться на одну клетку вправо или влево под управлением автомата (головки), которая является неподвижной.

Внешний алфавит. Конечное множество (например, A), элементы которого называются буквами (символами). Одна из букв этого алфавита (например, a_0) должна представлять собой пустой символ.

Внутренний алфавит. Конечное множество состояний головки (автомата). Одно из состояний (например, q_1) должно быть начальным (запускающим программу). Еще одно из состояний (q_0) должно быть конечным (завершающим программу) – состояние останова.

Таблица переходов. Описание поведения автомата (головки) в зависимости от состояния и считанного символа.

	#	S	0	1	ao
q1	0 →	0 →	→	→	q0



Задача. Требуется заменить все символы # и \$ на нули. В момент запуска головка находится над любой буквой слова.

	#	\$	0	1	a ₀
q ₁	←	←	←	←	→ q ₂
q ₂	0 →	0 →	→	→	q ₀

Даны два целых положительных числа в десятичной системе счисления. Сконструировать машину Тьюринга, которая будет находить разность этих чисел, если известно, что первое число больше второго, а между ними стоит знак минус. Каретка находится над левой крайней цифрой левого числа.

Применение результатов теории алгоритмов




Теоретический аспект: при исследовании некоторой задачи результаты теории алгоритмов позволяют ответить на вопрос: является ли эта задача в принципе алгоритмически разрешимой.

Для алгоритмически неразрешимых задач: возможно ли их сведение к задаче останова машины Тьюринга.

В случае алгоритмической разрешимости задачи возникает следующий важный теоретический вопрос: о принадлежности этой задачи к классу **NP-полных задач***. При утвердительном ответе на этот вопрос можно говорить о существенных временных затратах для получения точного решения для больших размерностей исходных данных.

Понятие алгоритма


- 
- Алгоритм** – это конечная последовательность указаний ...
- ... на языке понятном исполнителю, ...
 - ... задающая процесс решения задач определенного типа ...
 - ... и ведущая к получению результата, однозначно определяемого допустимыми исходными данными*.

**Результат выполнения алгоритма напрямую зависит от исходных данных. Т.е. один и тот же алгоритм при разных исходных данных даст разные результаты. С другой стороны, если одному и тому же алгоритму передать несколько раз одни и те же данные, он должен столько же раз выдать один и тот же результат*

Свойства алгоритма

- ✓ **Дискретность** (разделенность на части) и упорядоченность. Алгоритм должен состоять из отдельных действий, выполняемых последовательно друг за другом.
- ✓ **Детерминированность** (однозначная определенность). Многократное применение одного алгоритма к одному и тому же набору исходных данных всегда дает один и тот же результат.
- ✓ **Формальность.** Алгоритм не должен допускать неоднозначности толкования действий для исполнителя.
- ✓ **Результативность и конечность.** Работа алгоритма должна завершаться за определенное число шагов, при этом задача должна быть решена.
- ✓ **Массовость.** Определенный алгоритм должен быть применим ко всем однотипным задачам.

Способы записи алгоритмов



Словесное описание алгоритма – инструкция о выполнении действий в определенной последовательности с помощью слов и предложений естественного языка.

Формульно-словесный способ записи инструкция о действиях содержит формальные символы и выражения (формулы) в сочетании со словесными пояснениями.

Графическая запись или схема – это изображение алгоритма с помощью геометрических фигур, называемых блоками.

Псевдокод - система правил записи алгоритма с использованием набора определенных конструкций для описания управляющих действий. (*Ершовский язык*)

Языки программирования. К языкам программирования относятся языки команд или машинные языки и языки высокого уровня.

Схемы алгоритмов



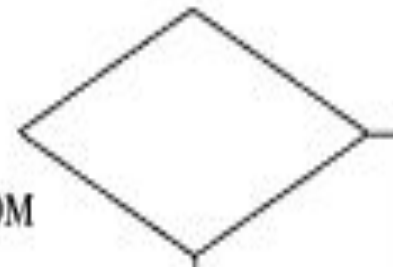
ПУСК/ОСТАНОВ –
начало или конец
выполнения действий



ПРОЦЕСС –
выполнение действия
или группы действий



ПРЕДОПРЕДЕЛЕННЫЙ
ПРОЦЕСС – действия
детализируются на другом
участке схемы



РЕШЕНИЕ – выбор
направления действия
в зависимости
от условия внутри
символа

Нет

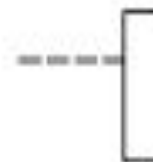
Да



ВВОД/ВЫВОД –
ввод данных
для обработки
или вывод результатов
обработки



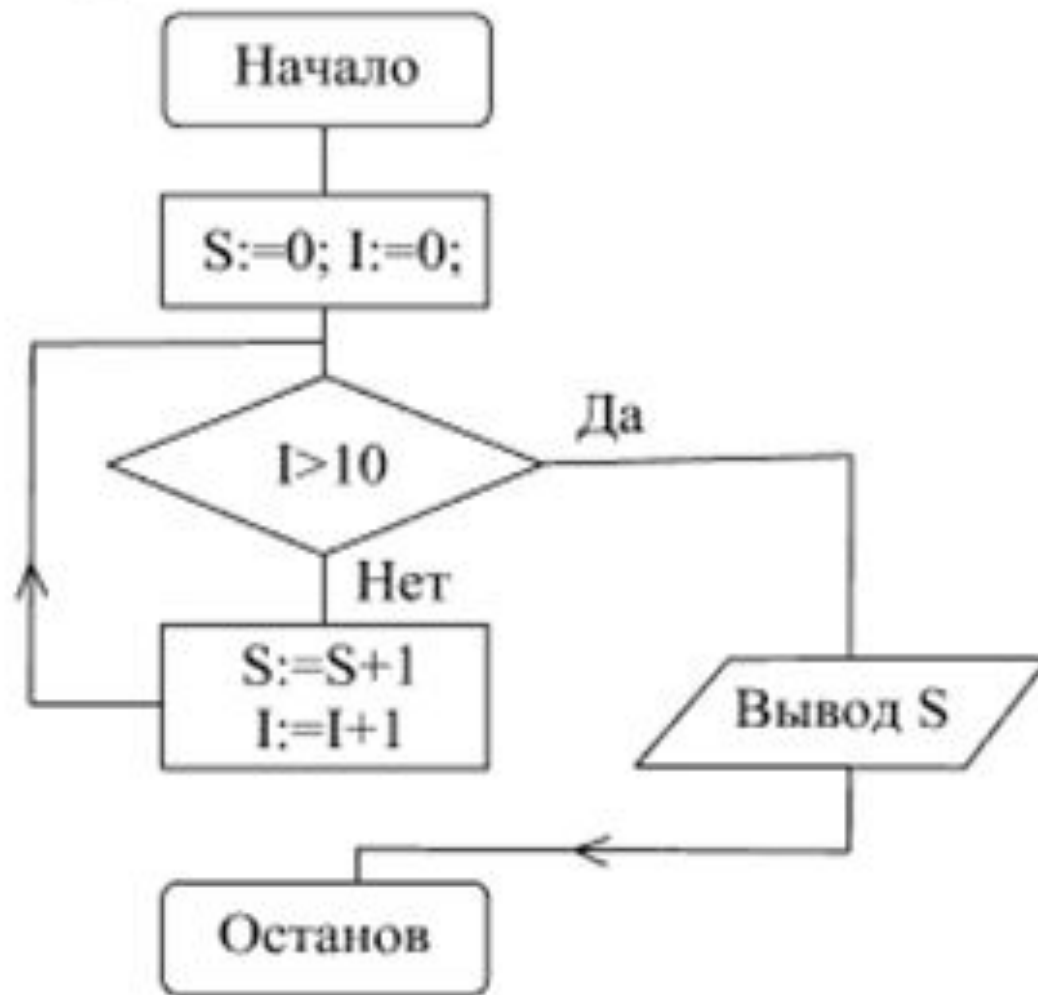
ЛИНИЯ ПОТОКА – связь
действий, отраженных
в схеме



КОММЕНТАРИЙ –
пояснения к символу

Основные виды алгоритмов

Циклический алгоритм



$$y = \begin{cases} \prod_{i=1}^n \sin(ix) & \text{при } x > b, \\ \sum_{i=1}^n \cos(ix) & \text{при } x \in [a, b], \\ e^x & \text{при } x < a. \end{cases}$$

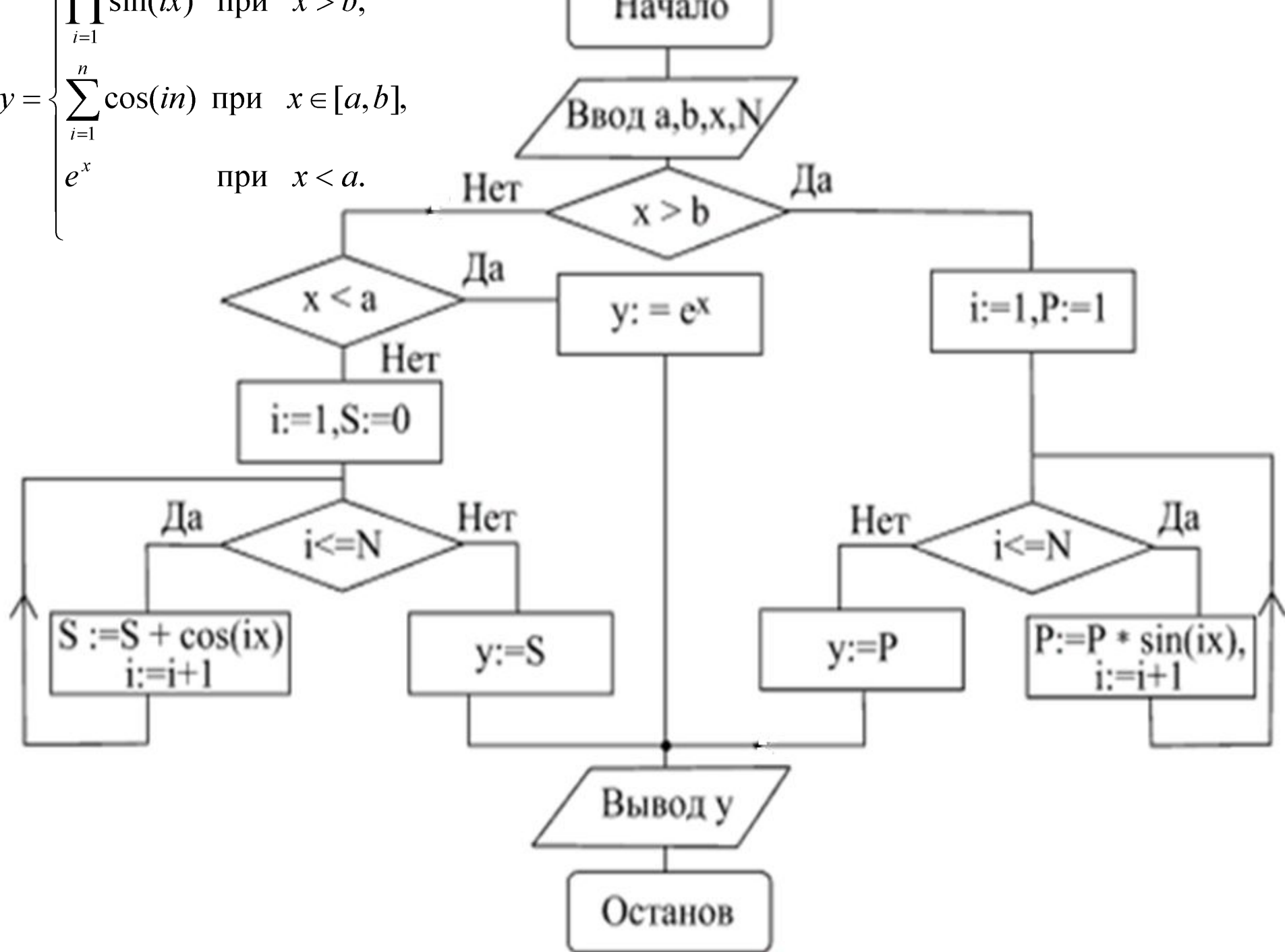
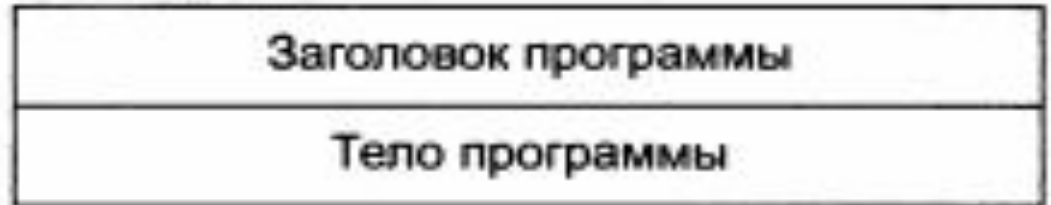
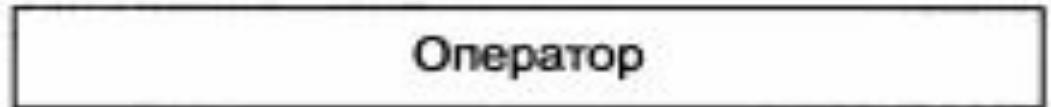


Диаграмма Нэсси–Шнейдермана

Программа



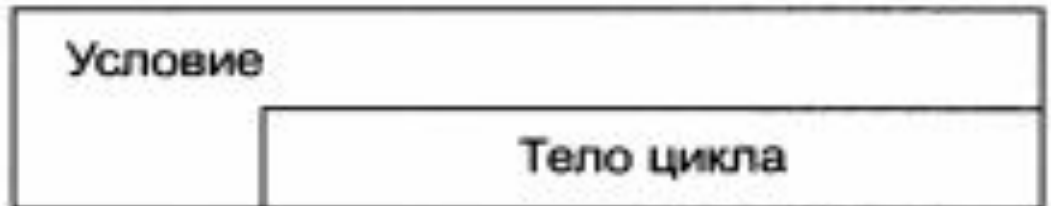
Простая конструкция



Условный оператор



Циклическая конструкция



Нахождение действительных корней квадратного уравнения

Ввод a, b, c

Да

$a = 0$

Нет

Да

$b = 0$

Нет

$d = b^2 - 4ac$

Да

$c = 0$

Нет

$x = -c / b$

Да

$d < 0$

Нет

Вывод "x —
любое решение"

Вывод "нет решений"

Вывод x

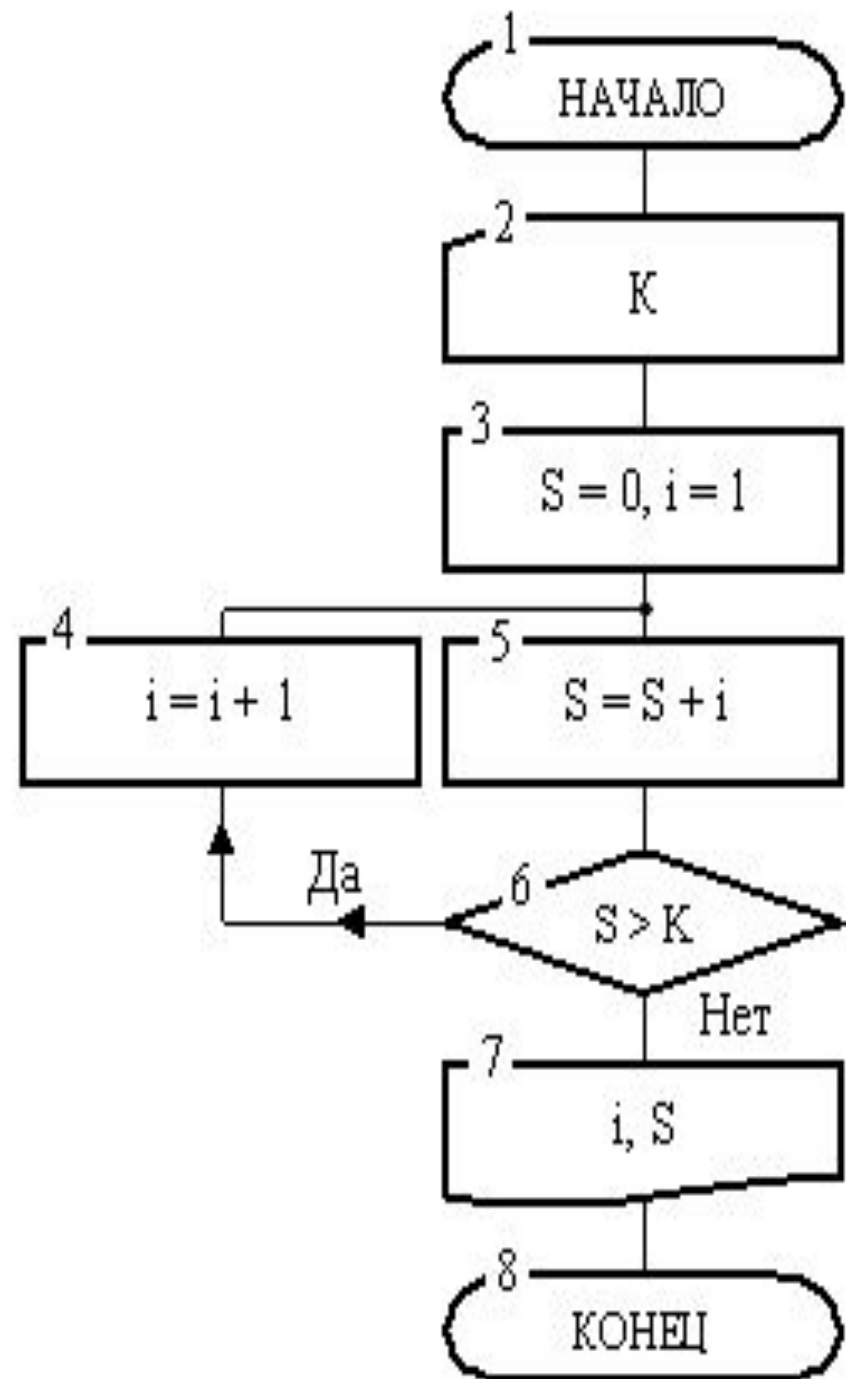
Вывод "нет
вещественных
корней"

$x_1 = (-b + d^{1/2}) / (2a)$
 $x_2 = (-b - d^{1/2}) / (2a)$

Вывод x_1, x_2

Конец

Пример 1. Рассмотрим пример алгоритма с циклом, имеющим наперед неизвестное количество проходов. Для этого решим следующую задачу. Указать наименьшее количество членов ряда натуральных чисел $1, 2, 3, \dots$, сумма которых больше числа K .



Пример 2. Рассмотрим задачу сортировки одномерного массива Z длины N . Отсортировать массив – значит расположить его элементы в порядке роста или убывания. Опишем метод сортировки массива в порядке роста. Сначала выполняется проход по массиву с целью определения в нем наименьшего элемента. Затем производится перестановка этого элемента с первым. Далее совершается второй проход по массиву, начиная со второго элемента. Найденный наименьший элемент переставляется со вторым и т. д. После $(N-1)$ -го прохода с выполнением названных операций массив окажется отсортированным.

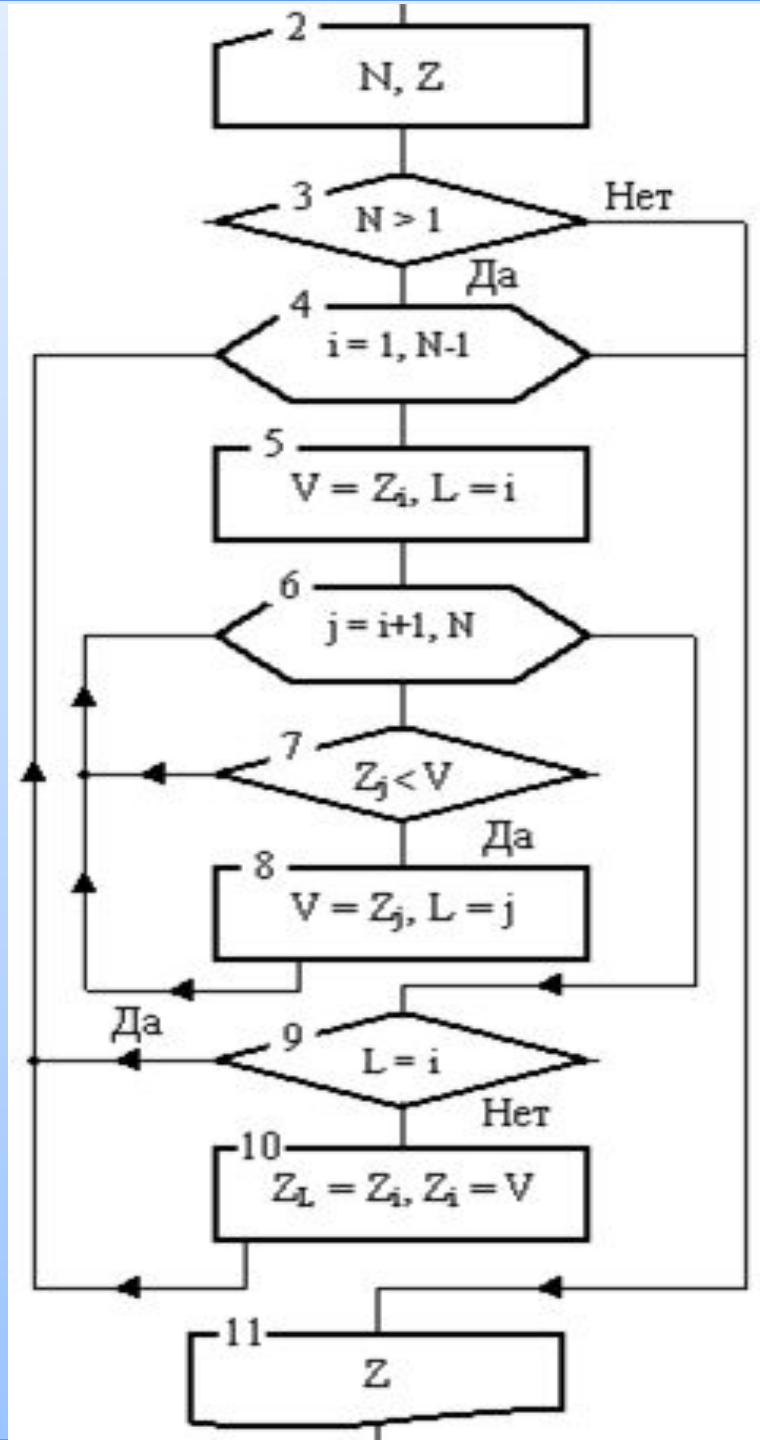
$n = 5$

$Z = \{1, 3, 5, 2, 4\}$

i	v	h	j	
1	1	1	2	$3 < 1$
			3	$5 < 1$
			4	$2 < 1$
		1	5	$4 < 1$
2	3	2	3	$5 < 3$
	2	4	4	$2 < 3$
			5	$4 < 2$
3	5 3	3 4	4	$3 < 5$
			5	$4 < 3$
4	5 4	4 5	5	$4 < 5$

~~1 2 3 4 5~~
~~1 2 3 4 5~~
~~1 2 3 4 5~~

вывод: 1 2 3 4 5



Пример 3. Дан двумерный квадратный массив Z , состоящий из N строк и N столбцов. Необходимо найти среднее арифметическое S его отрицательных элементов и заменить положительные элементы побочной диагонали массива средним арифметическим S .

