Основы алгоритмизации

История

Название **"алгоритм"** произошло от латинской формы имени величайшего среднеазиатского математика **Мухаммеда ибн Муса ал-Хорезми** (Alhorithmi), жившего в 783—850 гг.

В своей книге "Об индийском счете" он изложил правила записи натуральных чисел с помощью арабских цифр и правила действий над ними "столбиком", знакомые теперь каждому школьнику. В XII веке эта книга была переведена на латынь и получила широкое распространение в Европе.

Алгоритм

- **Алгоритм — заранее заданное понятное и точное предписание возможному исполнителю совершить определенную последовательность действий для получения решения задачи за конечное число шагов.**

Исполнитель алгоритма

- **Исполнитель алгоритма** — это некоторая абстрактная или реальная (техническая, биологическая или биотехническая) система, способная выполнить действия, предписываемые алгоритмом.

- Исполнителя характеризуют:
  - среда
  - элементарные действия
  - система команд
  - отказы

Свойства алгоритма

- **детерминированность** (определенность) – при заданных исходных данных обеспечивается однозначность искомого результата;
- **дискретность** – возможность разбиения алгоритма на отдельные этапы, выполнение которых не вызывает сомнений.
- **результативность** – реализуемый вычислительный процесс выполняется за конечное число этапов с выдачей осмысленного результата;
- **массовость** – пригодность для задач данного типа при исходных данных, принадлежащих заданному подмножеству;

Типы алгоритмов

- - алгоритмы линейной структуры;
- - алгоритмы разветвленной структуры;
- - алгоритмы циклической структуры.

- Алгоритмы решения практических задач обычно имеют комбинированную структуру, то есть включают в себя все три типа

Формы записи

- словесный (записи на естественном языке);
- структурно-стилизованный (записи на алгоритмическом языке и псевдокод);
- графический (изображение схем и графических символов);
- программный (тексты на языках программирования).

Словесная форма

- **Словесный способ** описания алгоритма представляет собой описание последовательных пронумерованных этапов обработки данных и задается в произвольном изложении на естественном языке.

- Достоинства:
  - Простота
- Недостатки
  - Многословность
  - Отсутствие строгой формализации
  - Неоднозначность

Примеры

Алгоритм сложения двух чисел (a и b)

1. Спросить, чему равно число a.
2. Спросить, чему равно число b.
3. Сложить a и b, результат присвоить c.
4. Сообщить результат c.

Пример

Под *неформализованным алгоритмом* будем понимать алгоритм, изложенный в виде набора действий, описанных на русском языке.

Задача:
Положим, Вы пришли в библиотеку и Вам нужно найти литературу по экономической теории. Вы рассчитываете получить необходимую литературу, пользуясь реестром книг библиотеки и помощью библиотекаря. Требуется описать алгоритм.

1. Открыть справочник терминов
2. Выбрать книгу из реестра
3. Разобрать название книги
4. Если название неизвестно, обратиться к библиотекарю
5. Если название неизвестно, пропустить книгу
6. Если книга связана с темой, занести её в список литературы
7. Если просмотрены не все книги, вернуться к пункту 2
8. Попросить у библиотекаря выбранные книги

Пример

Задача: Записать алгоритм нахождения наибольшего общего делителя (НОД) двух натуральных чисел (алгоритм Эвклида).

Алгоритм может быть следующим:

1. задать два числа;
2. если числа равны, то взять любое из них в качестве ответа и остановиться, в противном случае продолжить выполнение алгоритма;
3. определить большее из чисел;
4. заменить большее из чисел разностью большего и меньшего из чисел;
5. повторить алгоритм с шага 2.

Псевдокод

- **Структурно-стилизованный способ** описания алгоритма основан на записи алгоритмов в формализованном представлении предписаний, задаваемых путем использования ограниченного набора типовых синтаксических конструкций, называемых часто **псевдокодами**.

- **Псевдокод** представляет собой систему обозначений и правил, предназначенную для единообразной записи алгоритмов.

- Достоинства
  - Близость к языкам программирования
- Недостатки
  - Сложность освоения

Основные служебные слова

алг (алгоритм)	сим (символьный)	дано	для	да
арг (аргумент)	лит (литерный)	надо	от	нет
рез (результат)	лог (логический)	если	до	при
нач (начало)	таб(таблица)	то	знач	выбор
кон (конец)	нц (начало цикла)	иначе	и	ввод
цел (целый)	кц (конец цикла)	все	или	вывод
вещ (вещественный)	длин (длина)	пока	не	утв

Общий вид алгоритма

```
алг название алгоритма (аргументы и результаты)
  дано условия применимости алгоритма
  надо цель выполнения алгоритма
нач описание промежуточных величин
|   последовательность команд (тело алгоритма)
кон
```

Примеры предложений **алг**:

- **алг** Объем и площадь цилиндра (**арг вещ** R, H, **рез вещ** V, S)
- **алг** Корни КвУр (**арг вещ** a, b, c, **рез вещ** x1, x2, **рез лит** t)
- **алг** Исключить элемент (**арг цел** N, **арг рез вещ таб** A[1:N])
- **алг** Диагональ (**арг цел** N, **арг цел таб** A[1:N, 1:N], **рез лит** Otvet)

Дано и Надо

```
1. алг Замена (арг лит Str1, Str2, арг рез лит Text)
2.     дано | длины подстрок Str1 и Str2 совпадают
3.     надо | всюду в строке Text подстрока Str1 заменена на Str2
4.
5. алг Число максимумов (арг цел N, арг вещ таб A[1:N], рез цел K)
6.     дано | N>0
7.     надо | K — число максимальных элементов в таблице A
8.
9. алг Сопротивление (арг вещ R1, R2, арг цел N, рез вещ R)
10.      дано | N>5, R1>0, R2>0
11.      надо | R — сопротивление схемы
```

Команды школьного языка

- Команда присваивания
- Команды ввода и вывода.
  - - ввод имена переменных
  - - вывод имена переменных, выражения, тексты.
- Команды если и выбор
- Команды для и пока

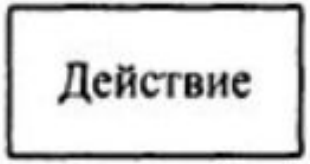
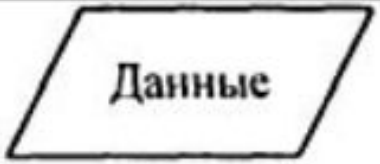
Пример программы на АЯ

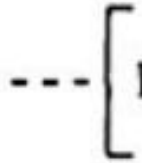
```
алг Сумма квадратов (арг цел n, рез цел S)
    дано | n > 0
    надо | S = 1*1 + 2*2 + 3*3 + ... + n*n
нач цел i
    ввод n; S:=0
    нц для i от 1 до n
      S:=S+i*i
    кц
    вывод "S = ", S
кон
```

Графический способ

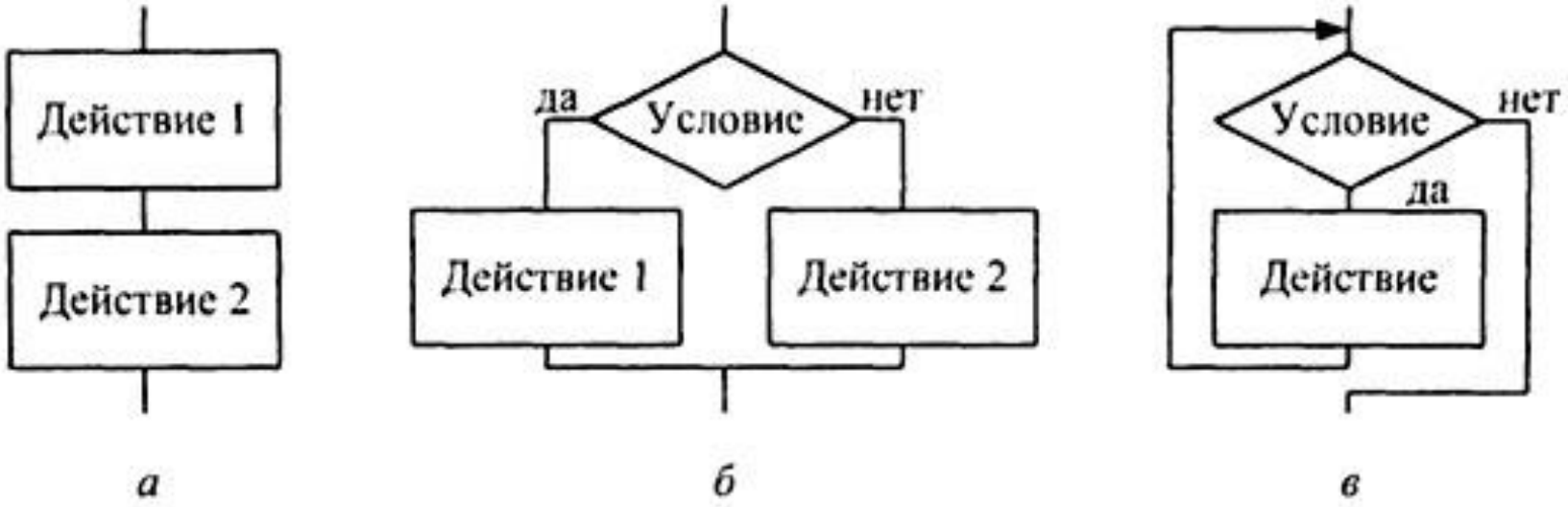
- **Графический способ** описания алгоритма предполагает, что для описания структуры алгоритма используется совокупность графических изображений (блоков), соединяемых линиями передачи управления. Такое изображение называется *методом* ***блок-схем***.

- ***Блок-схема*** алгоритма – это графическое представление хода решения задачи.

Название блока	Обозначение	Назначение блока
Терминатор		Начало, завершение программы или подпрограммы
Процесс		Обработка данных (вычисления, пересылки и т. п.)
Данные		Операции ввода-вывода
Решение		Ветвления, выбор, итерационные и поисковые циклы
Подготовка		Счетные циклы

Граница цикла		Любые циклы
Предопределенный процесс		Вызов процедур
Соединитель		Маркировка разрывов линий
Комментарий		Пояснения к операциям

Базовые алгоритмические структуры



а *б* *в*

- *следование* - обозначает последовательное выполнение
- *ветвление* - соответствует выбору одного из двух вариантов действий
- *цикл-пока* - определяет повторение действий, пока не будет нарушено условие, выполнение которого проверяется в начале цикла

Пример линейного алгоритма

Алгоритм вычисления значения выражения K=3b+6a

Ввод исх. данных: b, a

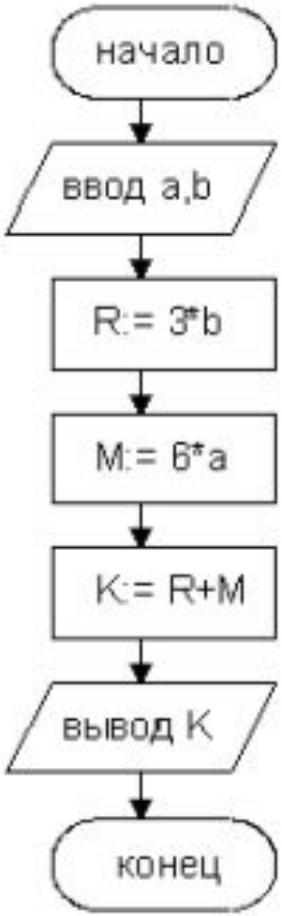
Вычисление R:= 3*b

Вычисление M:= 6*a

Вычисление K:= R+M

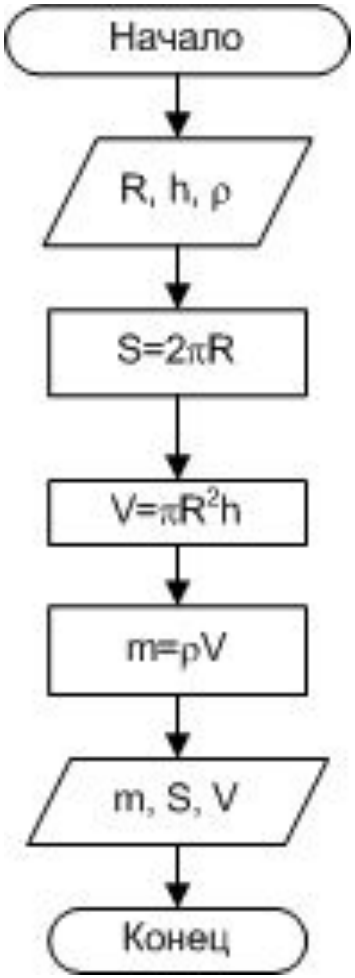
Вычисление K=3*b+6*a

Вывод результата: K



Пример линейного алгоритма

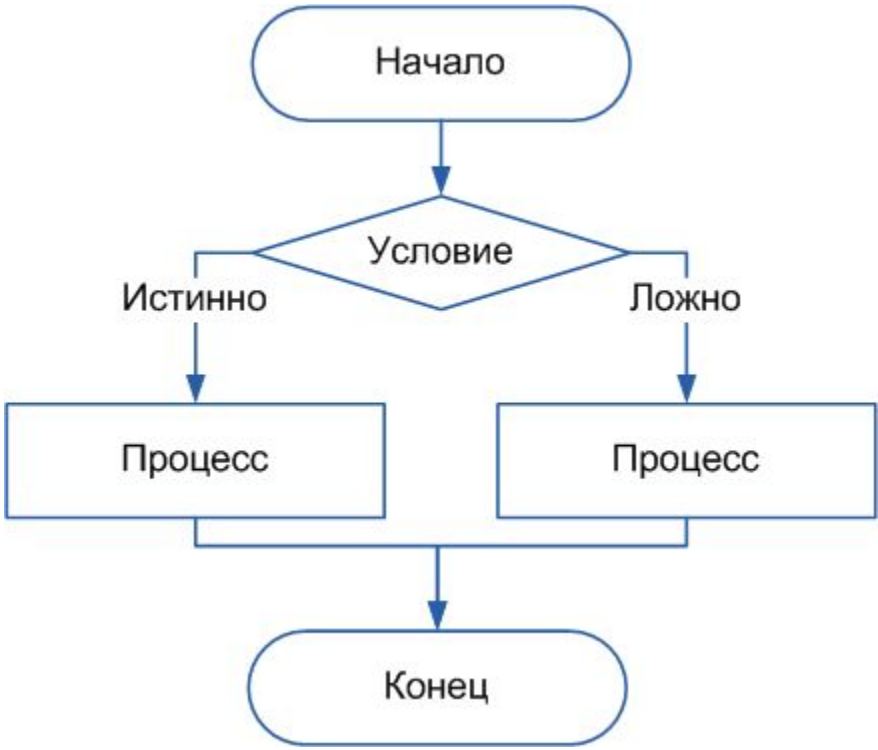
- Известны плотность и геометрические размеры цилиндрического слитка, полученного в металлургической лаборатории. Найти объем, массу и площадь основания слитка.
- ***Входные данные:*** **R** - радиус основания цилиндра, **h** - высота цилиндра, ρ- плотность материала слитка.
  Выходные данные: **m** - масса слитка, **V** - объем, **S** - площадь основания.



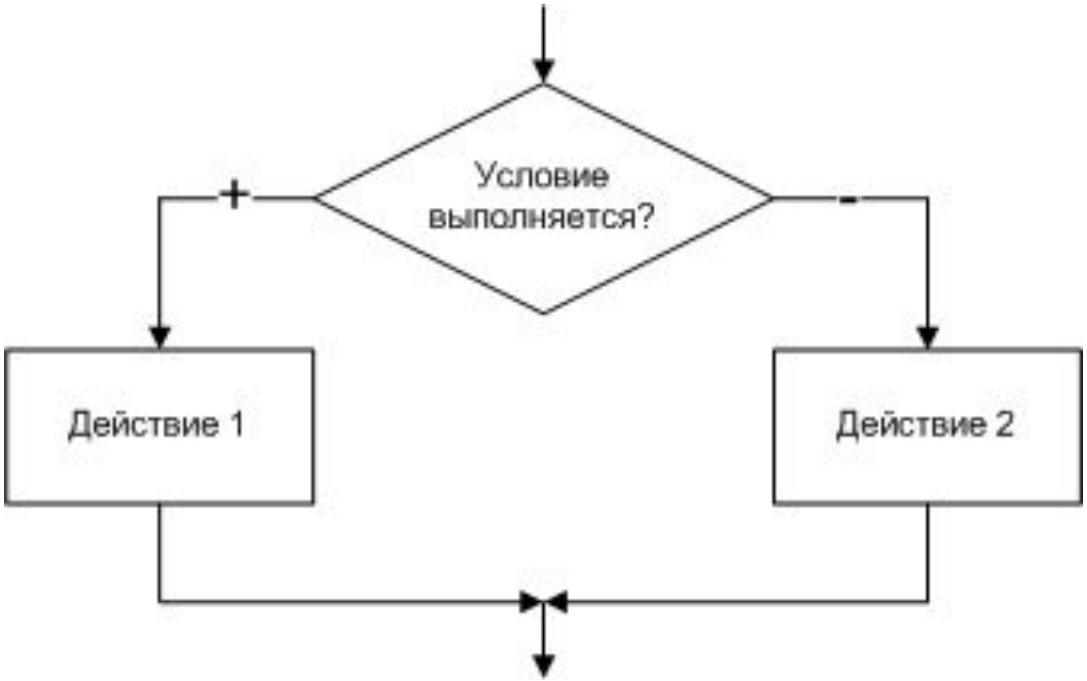
Ветвление

Ветвление

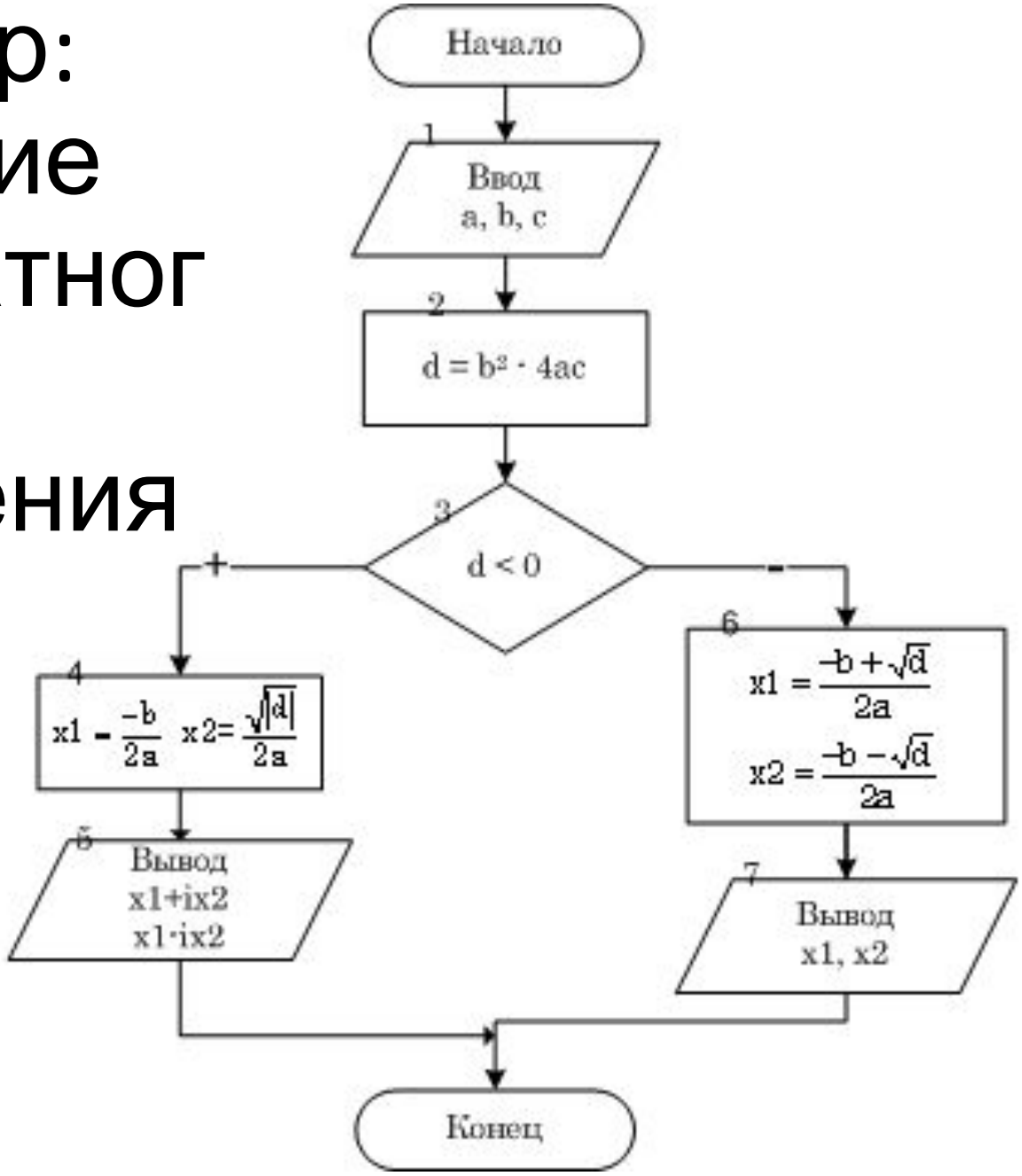
- Условие – логическое высказывание (см. дискретная математика)
- Пример условий: A= = 5, cost <100
- **Внимание:** ветви можно поменять местами, если изменить условие на противоположное
- Полные и неполные ветви



Полные и неполные ветви



Пример: решение квадратного уравнения



Циклы

Типы циклов

- *Счетные циклы (циклы с заданным количеством повторений)* – это циклические процессы, для которых количество повторений известно.
- *Итерационные циклы – это* циклические процессы, завершающиеся по достижении или нарушении некоторых условий.
- П*оисковые циклы* – это циклические процессы, из которых возможны два варианта выхода:
  - выход по завершению процесса;
  - досрочный выход по какому-либо дополнительному условию.

Цикл с предусловием

Американская нотация

Советская нотация

Внимание! Тело цикла может не выполняться, в принципе

Цикл с постусловием

Американская нотация

Советская нотация

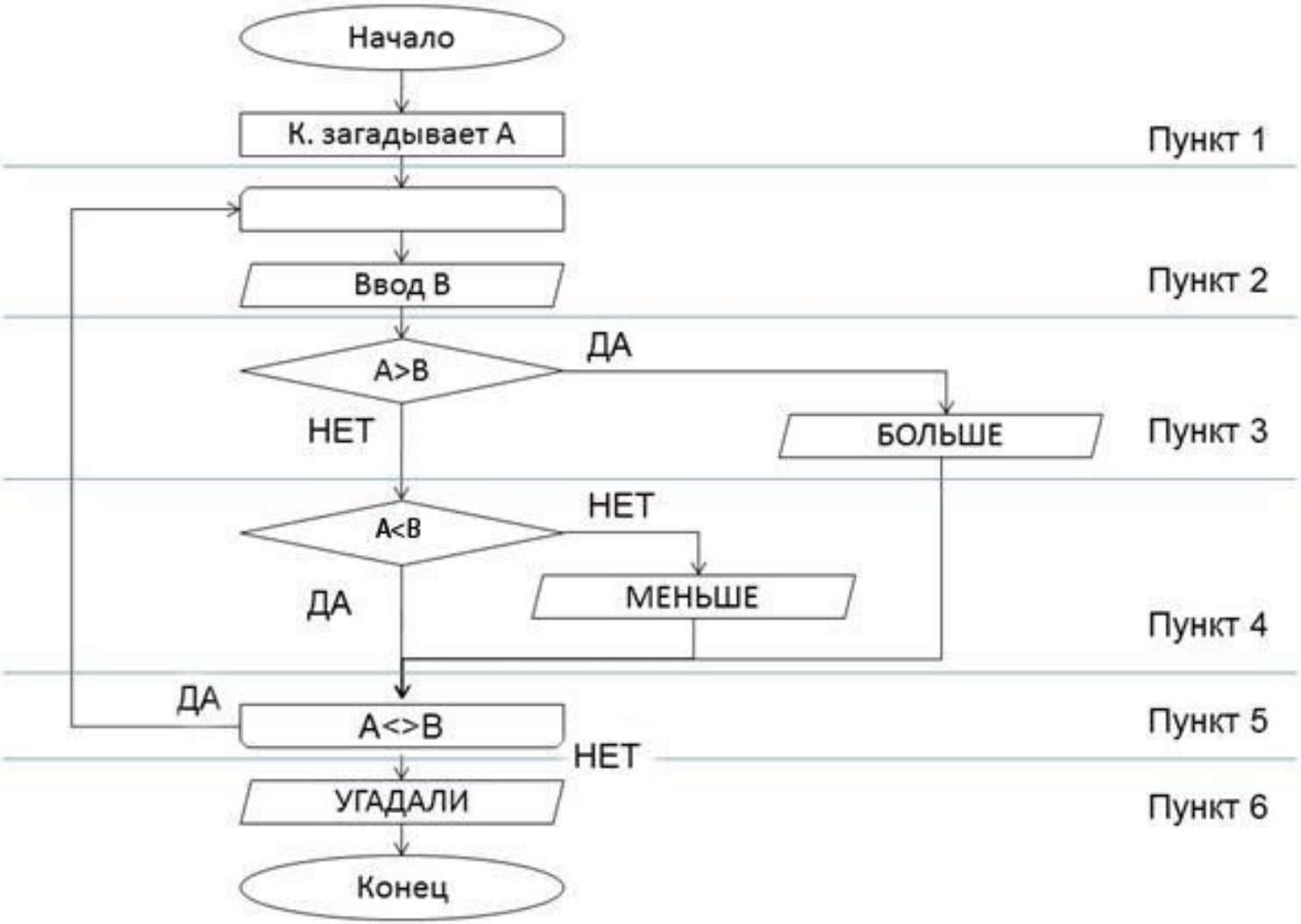
Внимание! Тело цикла выполнится минимум 1 раз

Задача: Угадай число

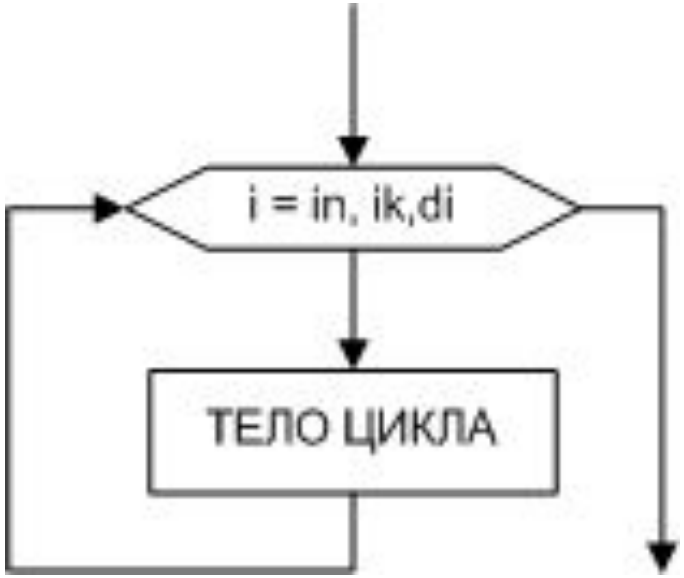
- Компьютер загадывает число от 0 до 10.
- Пользователь угадывает число и вводит его. Если введенное число меньше или больше загаданного, то компьютер выводит “загаданное число больше”, “загаданное число меньше” соответственно. Угадывание происходит до тех пор, пока пользователь точно не угадает число.
- Какой тип цикла использовать? Если в задаче, пользователь хотя бы 1 раз (обязательно) должен ввести число

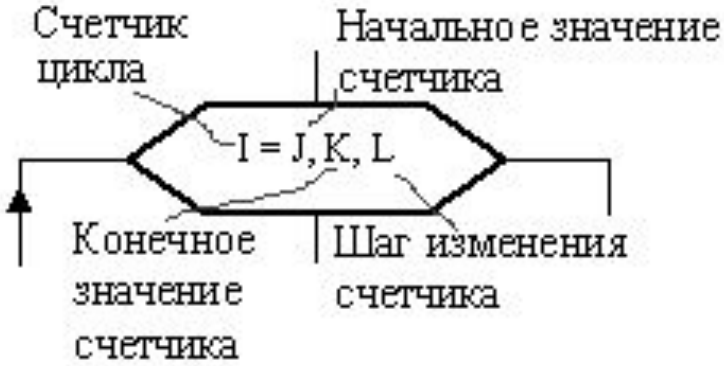
Неформализованное описание

1. Компьютер загадывает число A
2. Пользователь вводит число B (попытка угадать число)
3. Если загаданное число A больше числа B, то выводим “БОЛЬШЕ”
4. Если загаданное число A меньше числа B, то выводим “МЕНЬШЕ”
5. Если число не угадано, то переходим к п 2.
6. Если число угадано, то выводим “ВЫ УГАДАЛИ ЧИСЛО”


Начало
К. загадывает А
Пункт 1
Ввод В
Пункт 2
А>В
ДА
НЕТ
БОЛЬШЕ
Пункт 3
А<В
НЕТ
ДА
МЕНЬШЕ
Пункт 4
ДА
А<>В
Пункт 5
НЕТ
УГАДАЛИ
Пункт 6
Конец

Цикл и известным числом повторений



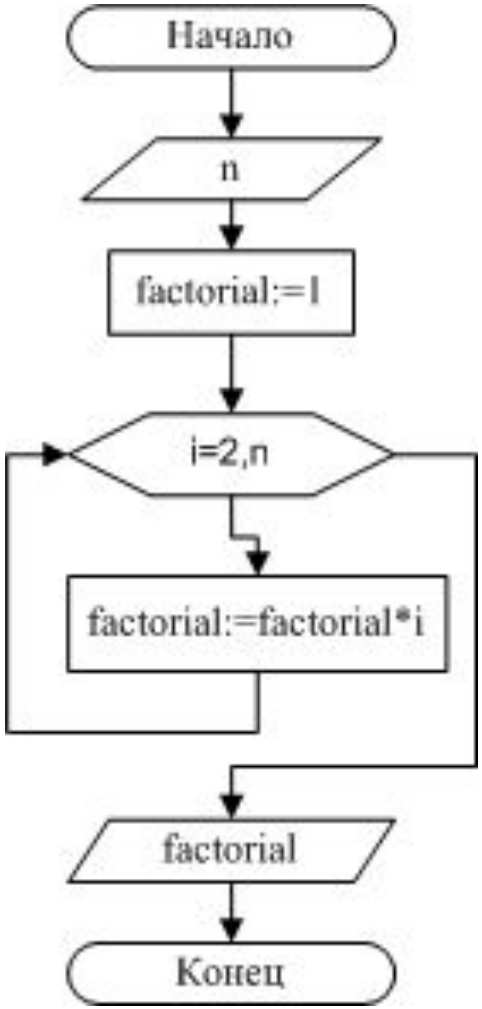


Задача: вычислить факториал

Входные данные: **N**-целое число, факториал которого необходимо вычислить.

Выходные данные: **factorial**- значение факториала числа **N**, произведение чисел от 1 до **N**, целое число.

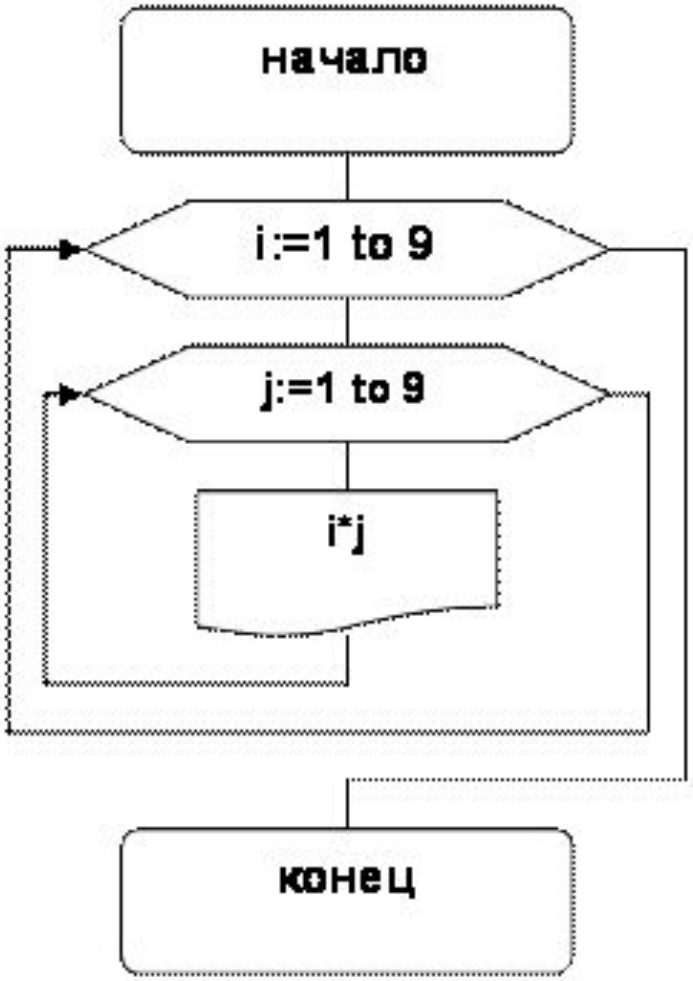
Промежуточные данные: **i**- целочисленная переменная, принимающая значения от 2 до **N** с шагом 1, параметр цикла.



Цикл с известным числом повторений (вариант 2)

Внимание: переменную **i** называют **параметром цикла**, так как это переменная, которая изменяется внутри цикла по определенному закону и влияет на его окончание.

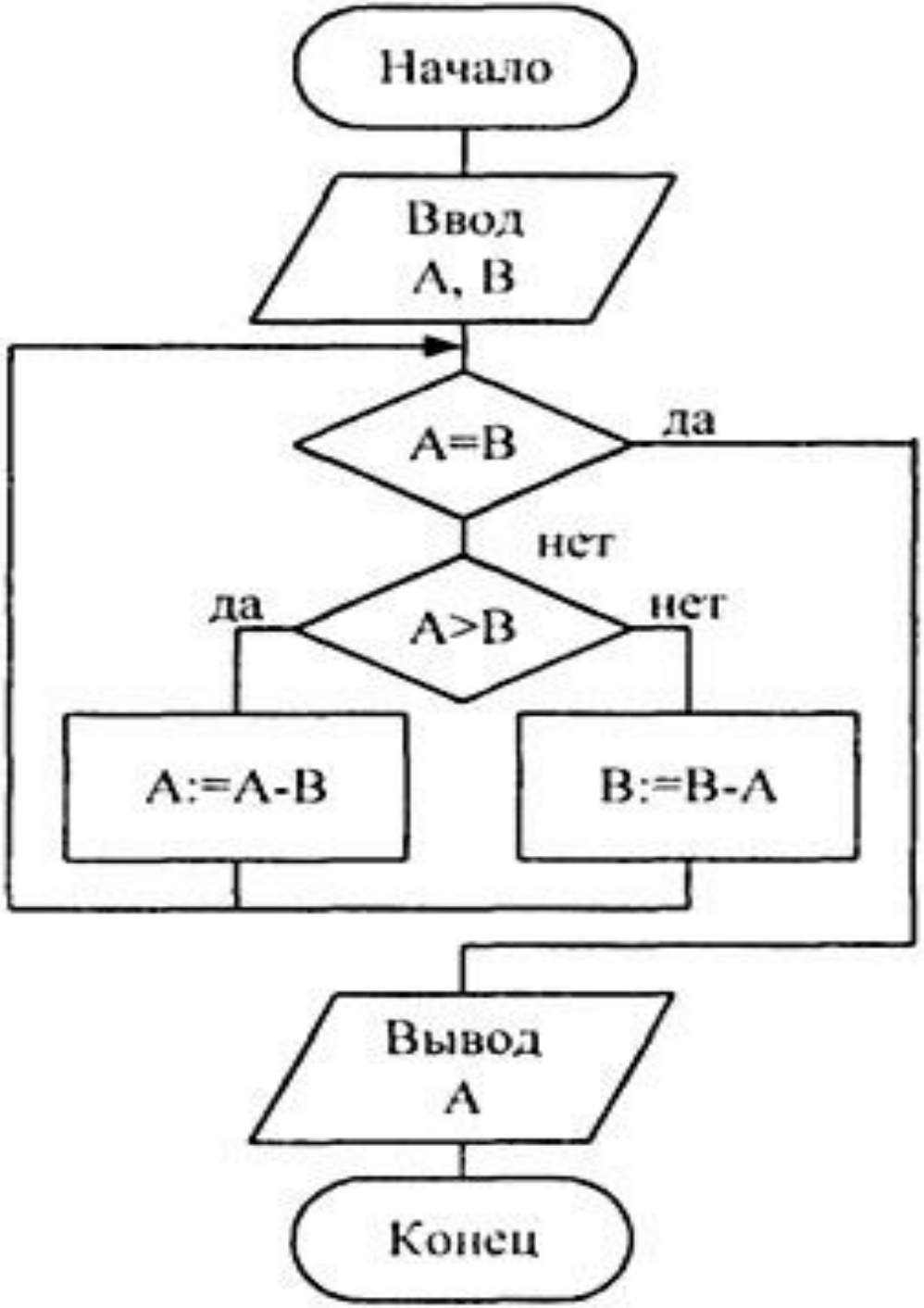
Вложенные циклы



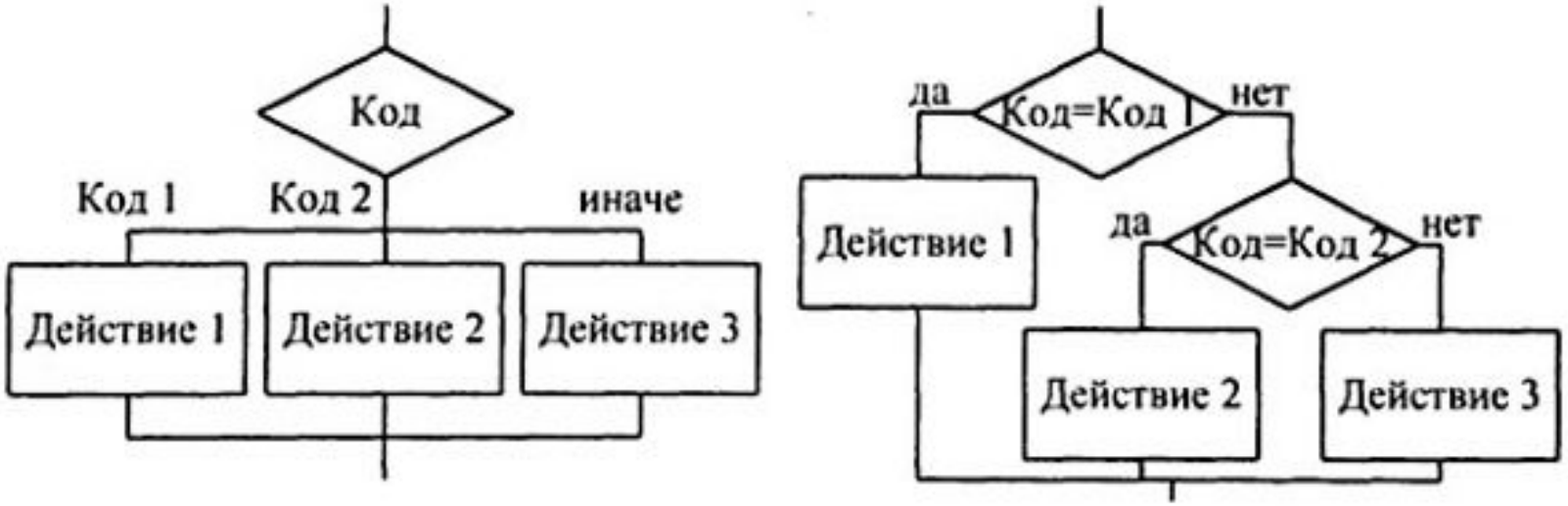
Пример комбинированного алгоритма

- Необходимо определить наибольший общий делитель двух натуральных чисел A и B.
- Для решения поставленной задачи используем алгоритм Евклида, который заключается в последовательной замене большего из чисел на разность большего и меньшего, пока числа не станут равны. Рассмотрим данный алгоритм на двух примерах.
- Пример (а): A=225, B=125. Применяя алгоритм Евклида, получаем для A и B наибольший общий делитель, равный 25.
- Пример (б): A=13, B=4. В этом случае наибольший общий делитель A и B равен 1.

Пример БСА комбинированного алгоритма

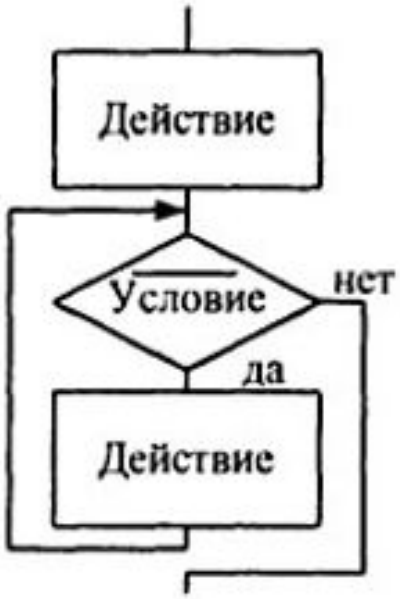


Дополнительные алгоритмические структуры



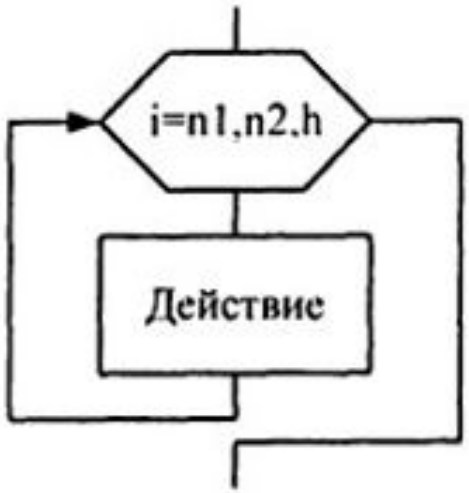
- *выбор* - выбор одного варианта из нескольких в зависимости от значения некоторой величины (рис. а, б);

Дополнительные алгоритмические структуры



- *цикл-до* - повторение некоторых действий до выполнения заданного условия, проверка которого осуществляется после выполнения действий в цикл;

Дополнительные алгоритмические структуры



- *цикл с заданным числом повторений (счетный цикл)* – повторение некоторых действий указанное число раз;

Контрольные вопросы

- Дайте определение алгоритма.
- Перечислите основные свойства алгоритмов и раскройте их сущность.
- Как подразделяются алгоритмы по типу реализуемого вычислительного процесса?
- Какие способы описания алгоритмов вам известны?
- Что понимается под графическим способом описания алгоритмов? В чем состоит преимущество данного способа перед словесным описанием алгоритма?
- Назовите базовые алгоритмические структуры и поясните их назначение.
- Каково назначение дополнительных алгоритмических структур? Каким образом они связаны с базовыми алгоритмическими структурами?