Алгоритмы. Свойства алгоритмов. Исполнители

Алгоритм

Алгоритм – это понятное и точное предписание исполнителю выполнить конечную последовательность команд, приводящую от исходных данных к искомому результату.

Свойства алгоритмов.

- **Понятность** – каждый шаг представлен в форме, понятной исполнителю (на его языке).
- **Дискретность** – разбиение на отдельные элементарные шаги.
- **Однозначность** - детерминированность, определённость формулировок, не допускающая разных толкований (исполнителю должно быть точно понятно, какой шаг выполнять дальше).

Свойства алгоритмов.

- **Результативность** – получение результата после конечного числа шагов, предусматривающее все возможные варианты (последовательность шагов не должна быть бесконечной).
- **Массовость** – возможность решать множество однотипных задач.

Способы записи алгоритмов

- словесный;
- табличный;
- графический;
- программа на алгоритмическом языке.

Способы записи алгоритмов

- Иногда алгоритмы записывают не на естественном, а формальном языке. Так, например, в решении шахматной задачи вместо фразы *“Конь, находящийся на поле d5, берет фигуру на поле f6 и объявляет шах”* пишут *“Kd5:f6+”.*
- Формализованная запись алгоритма понятна меньшему количеству исполнителей, но она обеспечивает краткость и недвусмысленность, облегчая тем самым задачу исполнения алгоритма.

Способы записи алгоритмов

- Графическая форма записи алгоритма более наглядна, чем словесная. Распространенным графическим способом представления алгоритмов являются блок-схемы.
- Блок-схема алгоритма состоит из блоков, соединенных линиями. Блоки различной формы изображают начало, конец и отдельные шаги алгоритма, а также условие выполнения шага.

Обозначение в блок-схемах

Начало- конец

Действие, операция

Принятие решения
(проверка условия)

Ввод-вывод данных

Исполнитель алгоритма

Исполнитель алгоритма - человек и/или автоматическое устройство:
- понимающий язык, на котором записан алгоритм; и
- способный выполнить этот алгоритм.

Кто может быть исполнителем алгоритма?

- Исполнителем алгоритма может быть не только человек, но и автоматическое устройство (реальное или воображаемое). В этом случае шаги алгоритма часто называют командами и вводят их в устройство в той форме, в которой оно сможет их обрабатывать.
- Языки алгоритмического управления устройствами являются формальными. Алгоритм, представленный на языке устройства, называется программой для этого устройства.

Разработка и исполнение

- Разрабатывает алгоритмы: человек,
- Исполняют алгоритмы: люди и устройства – компьютеры, роботы, станки, спутники, сложная бытовая техника, детские игрушки.
- Исполнитель решает задачу по заданному алгоритму, строго следуя по предписаниям (программе) не вникая и не рассуждая, почему он так делает.

Исполнителя характеризует:

- **Системой команд Исполнителя** называется совокупность всех команд, которые может выполнить Исполнитель.
- Совокупность всех действий, которые он может выполнить в ответ на эти команды, называется **системой допустимых действий Исполнителя**.
- **Среда** – это обстановка, в которой работает исполнитель
- **Элементарное действие** – действие, совершаемое исполнителем после вызова команды.
- **Отказы**. Возникают при вызове команды в недопустимом для данной команды состоянии среды.

Что такое псевдокод?

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

Псевдокод занимает промежуточное место между естественным и формальным языками. С одной стороны, он близок к обычному естественному языку, поэтому алгоритмы могут на нем записываться и читаться как обычный текст. С другой строны, в псевдокоде используются некоторые формальные конструкции и математическая символика, что приближает запись алгоритма к общепринятой математической записи.

Что такое псевдокод?

В псевдокоде не приняты строгие синтаксические правила для записи команд, присущие формальным языкам, что облегчает запись алгоритма на стадии его проектирования и дает возможность использовать более широкий набор команд, рассчитанный на абстрактного исполнителя.

Однако в псевдокоде обычно **имеются некоторые конструкции, присущие формальным языкам**, что облегчает переход от записи на псевдокоде к записи алгоритма на формальном языке. В частности, в псевдокоде, так же, как и в формальных языках, есть **служебные слова**, смысл которых определен раз и навсегда. Они выделяются в печатном тексте жирным шрифтом, а в рукописном тексте подчеркиваются.

Что такое псевдокод?

- Примером псевдокода является школьный алгоритмический язык в русской нотации (**школьный АЯ**). Этот язык в дальнейшем мы будем называть просто "алгоритмический язык".

Как записываются алгоритмы на школьном алгоритмическом языке? Основные служебные слова

алг (алгоритм)
сим (символьный)
арг (аргумент)
лит (литерный)
рез (результат)
лог (логический)
нач (начало)
таб(таблица)
кон (конец)
нц (начало цикла)

Как записываются алгоритмы на школьном алгоритмическом языке? Основные служебные слова

дано
для
да
надо
от
нет
если
до
при
то

Как записываются алгоритмы на школьном алгоритмическом языке?

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

Как записываются алгоритмы на школьном алгоритмическом языке?

Часть алгоритма от слова **алг** до слова **нач** называется **заголовком**, а часть, заключенная между словами **нач** и **кон** — **телом** алгоритма.

После знака "|" записываются **комментарии**. Комментарии можно помещать в конце любой строки. Они не обрабатываются транслятором, но существенно облегчают понимание алгоритма.

Команды школьного АЯ

Команда присваивания.

Служит для вычисления выражений и присваивания их значений переменным.

Общий вид: **A:=B**

знак ":=" означает команду **заменить прежнее значение переменной, стоящей в левой части, на вычисленное значение выражения, стоящего в правой части.**

Например, a := (b+c) * sin(Pi/4); i := i+1.

Команды школьного АЯ

Команды ввода и вывода.

ввод имена переменных

вывод имена переменных, выражения, тексты.

Команды если и **выбор.** Применяют для организации ветвлений.

Команды для и **пока.** Применяют для организации циклов.

Пример записи алгоритма на школьном 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
```

Какие понятия используют алгоритмические языки?

Понятие языка определяется во взаимодействии синтаксических и семантических правил.

Синтаксические правила показывают, как образуется данное понятие из других понятий и букв алфавита,

а семантические правила определяют свойства данного понятия.

Какие понятия используют алгоритмические языки?

Основными понятиями в алгоритмических языках обычно являются следующие.

- **Имена** (идентификаторы) — **употребляются для обозначения объектов программы** (переменных, массивов, функций и др.).
- **Операции.** Типы операций:
  - **арифметические** операции + , —, *, / и др. ;
  - **логические** операции **и, или, не** ;

Какие понятия используют алгоритмические языки?

- операции **отношения** < ,>,<=,>= ,=, <>;
- операция **сцепки** (присоединения) символьных значений друг с другом с образованием одной длинной строки; изображается знаком "+«.

Какие понятия используют алгоритмические языки?

- **Данные — величины, обрабатываемые программой.** Имеется три основных вида данных: **константы, переменные и массивы.**
  - **Константы** — это данные, которые зафиксированы в тексте программы и не изменяются в процессе ее выполнения.
    Пример: да(истина); 7.5; «+»; «мир»

Какие понятия используют алгоритмические языки?

- **Переменные** обозначаются именами и могут изменять свои значения в ходе выполнения программы. Переменные бывают **целые, вещественные, логические, символьные и литерные**.

Какие понятия используют алгоритмические языки?

- **Массивы — последовательности однотипных элементов, число которых фиксировано и которым присвоено одно имя.** Положение элемента в массиве однозначно определяется его индексами (одним, в случае одномерного массива, или несколькими, если массив многомерный). Иногда массивы называют **таблицами.**

Какие понятия используют алгоритмические языки?

- **Выражения — предназначаются для выполнения необходимых вычислений**, состоят из констант, переменных, указателей функций (например, exp(x)), объединенных знаками операций.

Выражения записываются в виде **линейных последовательностей символов** (без подстрочных и надстрочных символов, "многоэтажных" дробей и т.д.), что позволяет вводить их в компьютер, последовательно нажимая на соответствующие клавиши клавиатуры.

Различают выражения **арифметические, логические и строковые.**

Какие понятия используют алгоритмические языки?

Арифметические выражения служат для определения одного числового значения. Например, (1+sin(x))/2. Значение этого выражения при x=0 равно 0.5, а при x=p/2 — единице.

Логические выражения описывают некоторые условия, которые могут удовлетворяться или не удовлетворяться. Таким образом, логическое выражение может принимать только два значения — **"истина"** или "**ложь"** (**да** или **нет**). Рассмотрим в качестве примера логическое выражение x*x + y*y < r*r , определяющее принадлежность точки с координатами (x, y) внутренней области круга радиусом r с центром в начале координат. При x=1, y=1, r=2 значение этого выражения — **"истина"**, а при x=2, y=2, r=1 — **"ложь".**

Строковые (литерные) выражения, значениями которых являются тексты. В строковые выражения могут входить литерные и строковые константы, литерные и строковые переменные, литерные функции, разделенные знаками операции сцепки. Например, A + B означает присоединение строки B к концу строки A . Если A = *"куст "*, а B = *"зеленый"*, то значение выражения A + B есть *"куст зеленый".*

Базовые алгоритмические структуры

1. **Базовая структура "следование".** Образуется последовательностью действий, следующих одно за другим:
2. **Базовая структура "ветвление".** Обеспечивает в зависимости от результата проверки условия (**да** или **нет**) выбор одного из альтернативных путей работы алгоритма. Каждый из путей ведет к **общему выходу**, так что работа алгоритма будет продолжаться независимо от того, какой путь будет выбран. Структура **ветвление** существует в четырех основных вариантах:

Базовые алгоритмические структуры

- если—то;
- если—то—иначе;
- выбор;
- выбор—иначе.

Базовые алгоритмические структуры

3. Базовая структура "**цикл**".

Обеспечивает многократное выполнение некоторой совокупности действий, которая называется телом цикла.

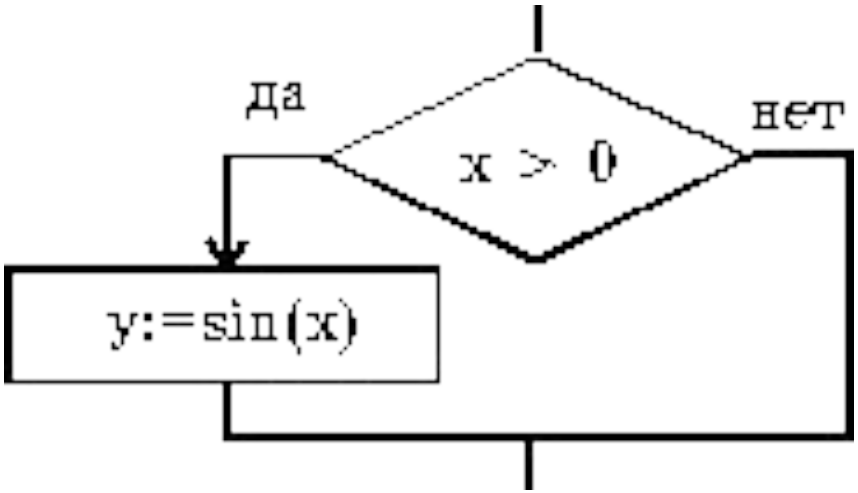
базовые алгоритмические структуры

Примеры структуры **ветвление**

Школьный алгоритмический язык

если x > 0
  то y := sin(x)
все

Язык блок-схем

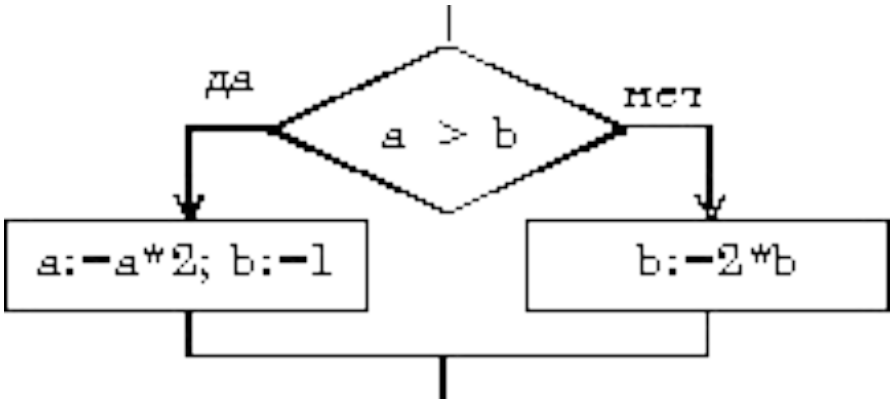


базовые алгоритмические структуры

Примеры структуры **ветвление**

Школьный алгоритмический язык	Язык блок-схем

если a > b
  то a := 2*a; b := 1
  иначе b := 2*b
все



базовые алгоритмические структуры

Примеры структуры **ветвление**

Школьный алгоритмический язык

выбор

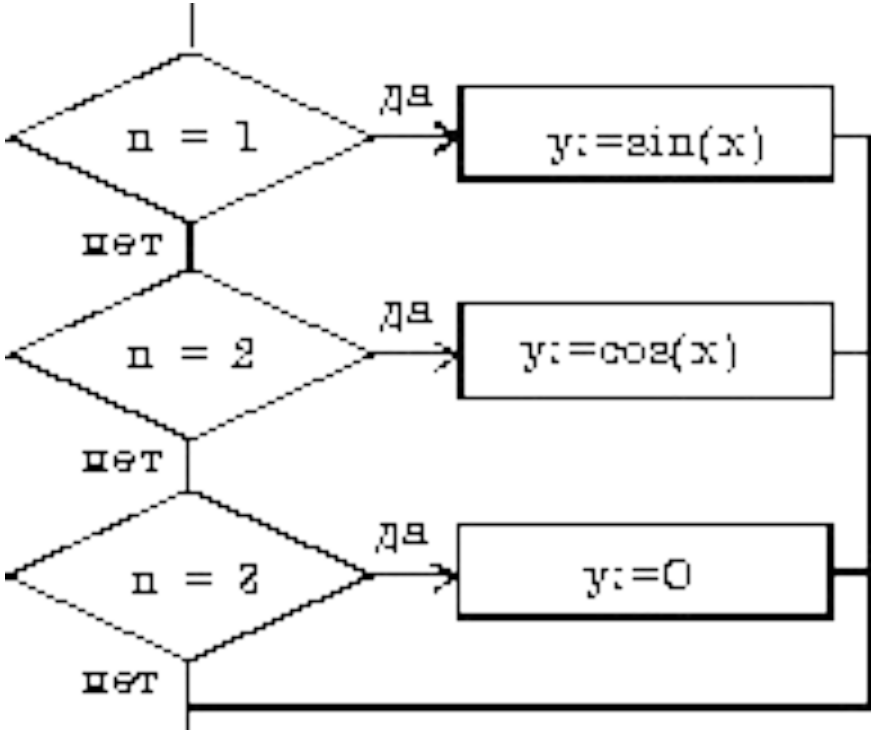
при n=1: y:=sin(x)

при n=2: y:= cos(x)

при n=3: y:=0

все

Язык блок-схем



Примеры структуры <u>цикл</u>

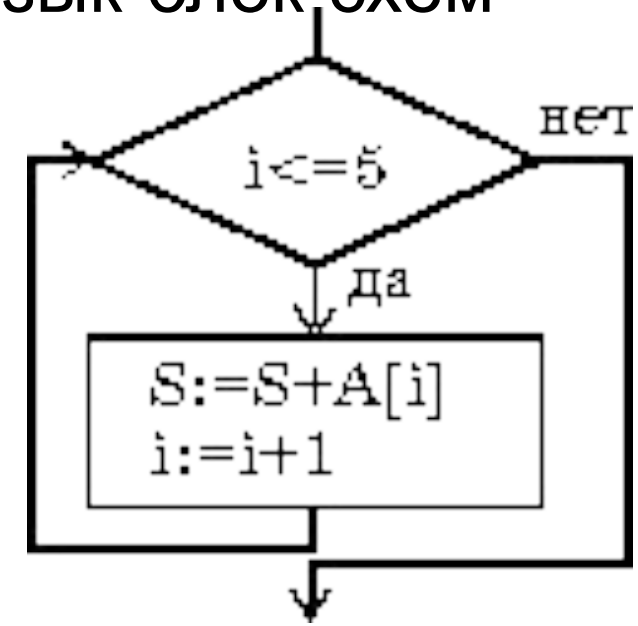
Цикл типа **<u>пока.</u>**

Предписывает выполнять тело цикла до тех пор, пока выполняется условие, записанное после слова пока.

Школьный алгоритмический язык

```
нц пока i <= 5
  S := S+A[i]   i := i+1
кц
```

Язык блок-схем



Примеры структуры <u>цикл</u>

Цикл типа **<u>для.</u>**

Предписывает выполнять тело цикла для всех значений некоторой переменной (параметра цикла) в заданном диапазоне.

Школьный алгоритмический язык

```
Нц  для i от 1 до 5
   X[i] := i*i*i
   Y[i] := X[i]/2
кц
```

Язык блок-схем