

Севастопольский государственный университет  
Кафедра информационных систем

Курс лекций по дисциплине  
«Алгоритмизация и программирование»

Бондарев Владимир Николаевич  
Сметанина Татьяна Ивановна

## Лекция 3

Алгоритм и его свойства. Способы описания алгоритмов. Разновидности структур алгоритмов. Линейные структуры алгоритмов.

# Понятие алгоритм

**Алгоритм** – конечное множество указаний (правил), определяющих содержание и последовательность действий над исходными и промежуточными данными с целью получения решения конкретной задачи за конечное число шагов.



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

В своей книге "Об индийском счете" он изложил основы десятичной позиционной системы записи чисел и правила действий над ними. Термин алгоритм был впервые использован Лейбницем.

# Свойства алгоритма

- 1. Определенность (детерминированность)** — каждый шаг алгоритма должен интерпретироваться исполнителем однозначно.
- 2. Результативность** состоит в том, что за конечное число шагов алгоритм либо должен приводить к решению задачи, либо после конечного числа шагов останавливаться из-за невозможности получить решение с выдачей соответствующего сообщения
- 3. Дискретность** — алгоритм должен представлять процесс решения задачи как последовательное выполнение простых шагов.
- 4. Эффективность** — во время выполнения алгоритма должен использоваться ограниченный объем ресурсов компьютера.
- 5. Массовость** — алгоритм решения задачи разрабатывается в общем виде, т.е. он должен быть применим для некоторого класса задач, различающихся лишь исходными данными. При этом исходные данные могут выбираться из некоторой области, которая называется *областью применимости алгоритма*.

# Способы описания алгоритмов

1. Словесное описание (запись на естественном языке);
2. Описание с помощью схем алгоритмов;
3. Описание на псевдоязыках;
4. Структурограммы (диаграммы Насси-Шнейдермана);
5. Программа на алгоритмическом языке.

**Словесный способ** записи алгоритмов представляет собой описание последовательных этапов обработки данных на естественном языке.

Алгоритм нахождения **наибольшего общего делителя (НОД)** двух натуральных чисел (алгоритм Эвклида) :

- 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.



# Схемы алгоритмов

Правила оформления схем алгоритмов (программ) устанавливает **ГОСТ 19.701-90. Схемы алгоритмов (программ)** отображают последовательность операций в программе. Они состоят из имеющих заданное значение графических символов, краткого пояснительного текста и соединяющих линий.



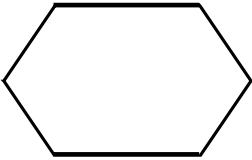
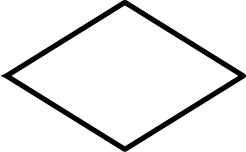
На схемах алгоритмов применяют следующие виды символов:

- ✓ **символы процесса**, указывающие фактические операции обработки данных;
- ✓ **символы линий**, указывающие поток управления;
- ✓ **специальные символы**, используемые для облегчения написания и чтения схемы.

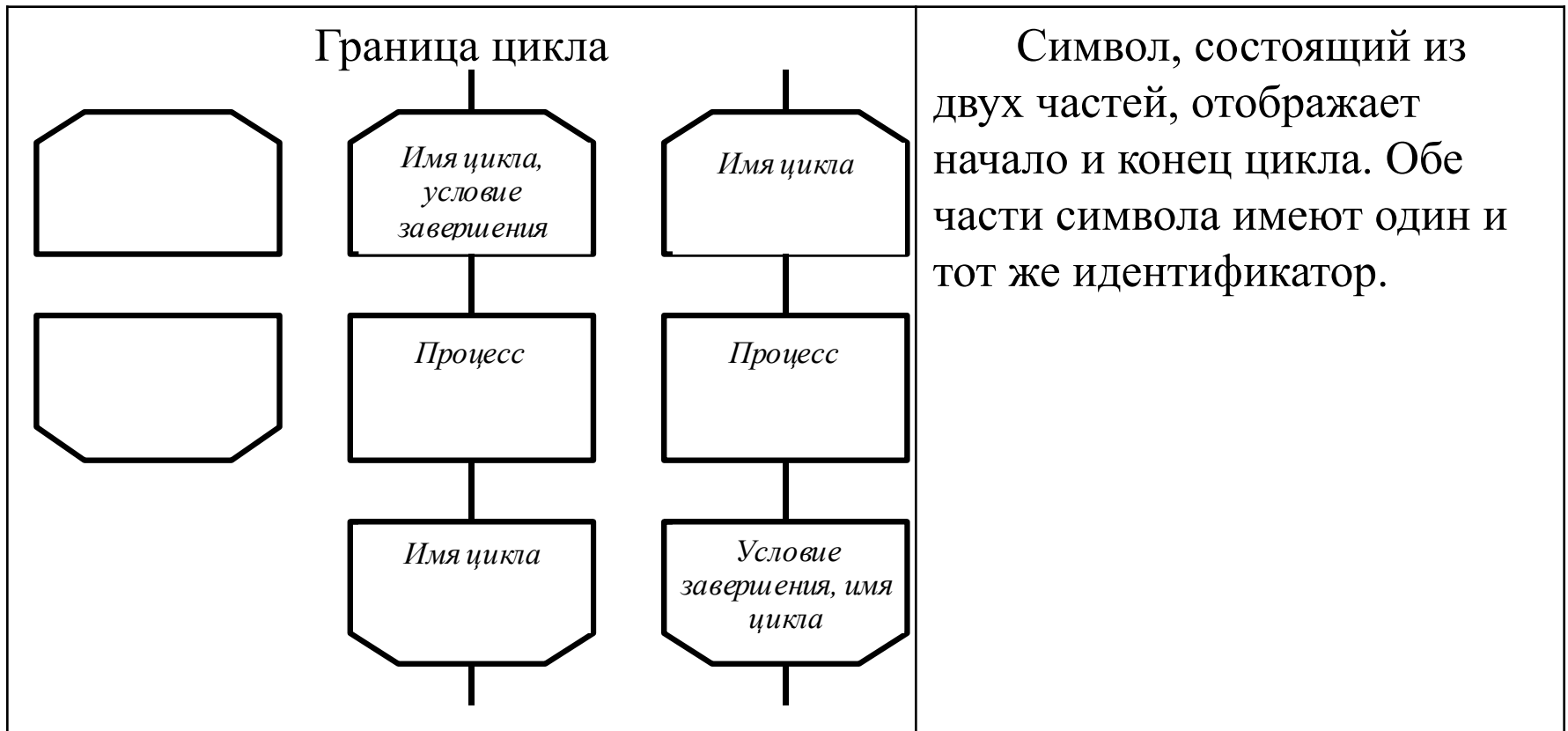
## Символы линий

Линия 	Отображает поток данных или управления. При необходимости могут быть добавлены стрелки-указатели.
Пунктирная линия 	Отображает альтернативную связь между двумя или более символами. Может использоваться для обведения участка.

## Схемы алгоритмов

Процесс 	Символ отображает функцию обработки данных любого вида (выполнение определенной операции или группы операций)
Предопределенный процесс 	Символ отображает предопределенный процесс, состоящий из одной или нескольких операций, которые определены в другом месте (в подпрограмме, модуле).
Подготовка 	Символ отображает модификацию команды или группы команд с целью воздействия на некоторую последующую функцию.
Решение 	Символ отображает решение или функцию переключательного типа, имеющую один вход и ряд альтернативных выходов, один из которых может быть активизирован после вычисления условий, определенных внутри этого символа.

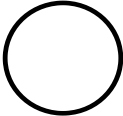

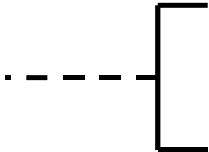
## Символы процесса




При необходимости определения **ориентировочных размеров** символов можно воспользоваться уже **недействующим** стандартом **ГОСТ 19.003-80**.



## Специальные символы

Соединитель 	Символ используется для обрыва линии и продолжения ее в другом месте.
Терминатор 	Символ отображает начало или конец схемы алгоритма.
Комментарий 	Символ используют для добавления комментариев в целях объяснения. Пунктирные линии в символе комментария связаны с соответствующим символом или могут обходить группу символов.

## Символы данных

Данные 	Символ отображает данные, носитель данных не определен.
---	---

## Правила оформления схем алгоритмов

Символы могут быть **вычерчены в любой ориентации**, но предпочтительной является горизонтальная ориентация.

Минимальное количество **текста**, необходимого для понимания функции символа, следует **помещать внутри символа**. Если объем текста превышает размеры символа, то следует использовать **символ комментария**.

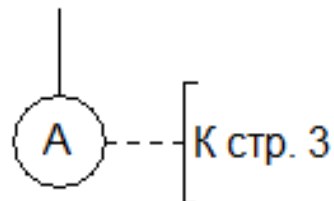
Символ может иметь **идентификатор** (номер), который располагают **слева над символом**. Его используют для ссылки на символ при описании схемы.

Если поток имеет направление, отличное от **стандартного**, то линии **изображают со стрелками**.

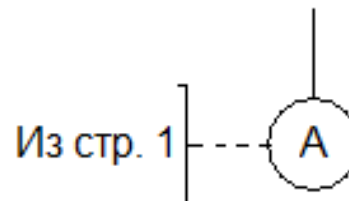
В схемах следует **избегать пересечения линий**. **Изменения направления линий в точках пересечения не допускается**.

При необходимости линии следует **разрывать** во избежание излишних пересечений, а также, если схема изображена на нескольких страницах.

Внешний соединитель



Внутренний соединитель

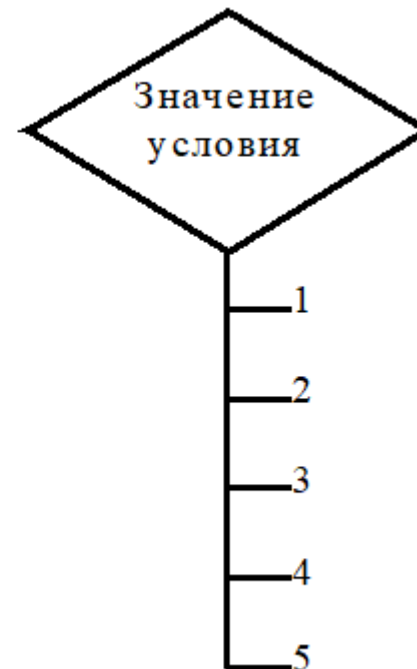


## Правила оформления схем алгоритмов

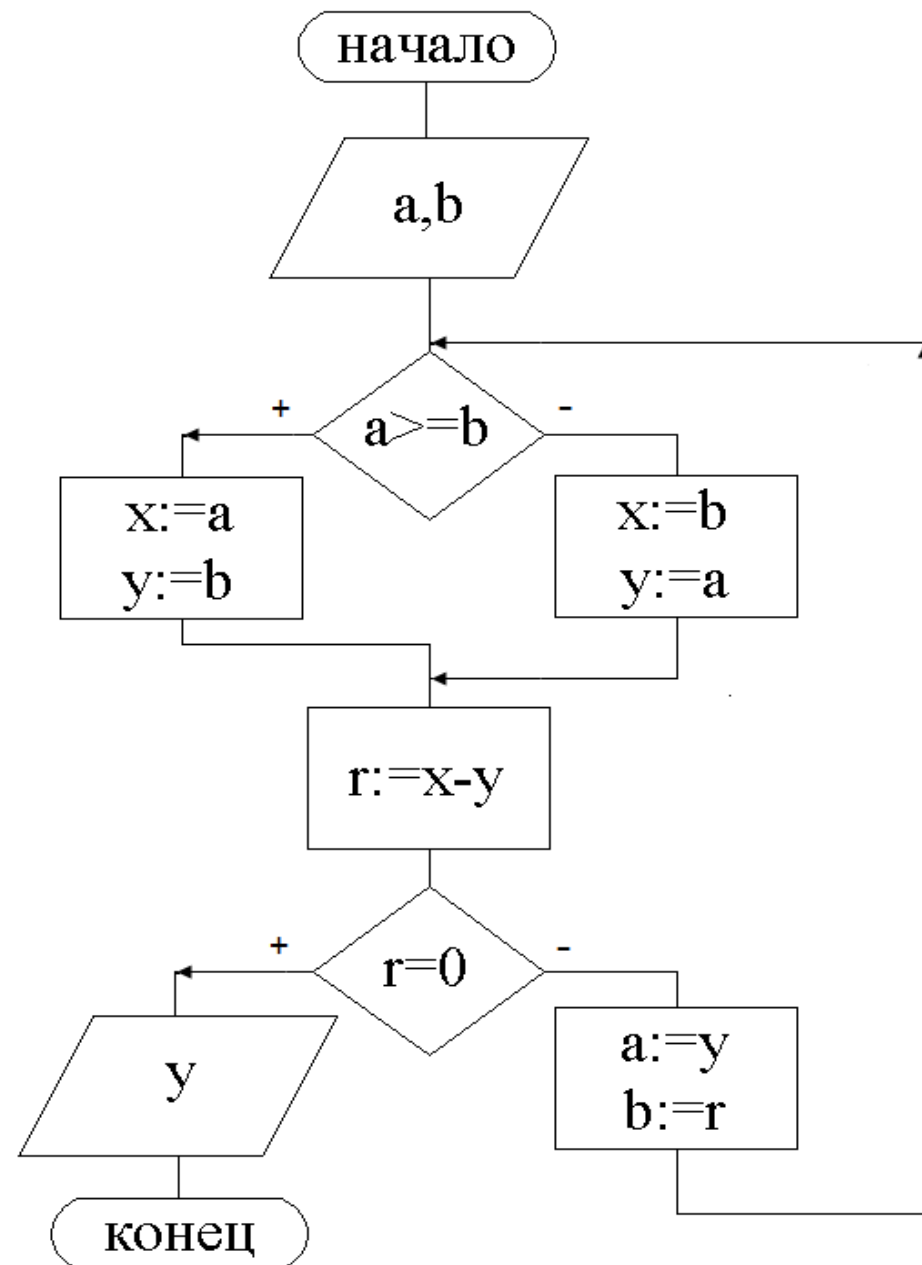
**Несколько выходов** из символа следует показывать:

- а) несколькими линиями от данного символа к другим символам;
- б) одной линией от данного символа, которая затем разветвляется в соответствующее число линий.

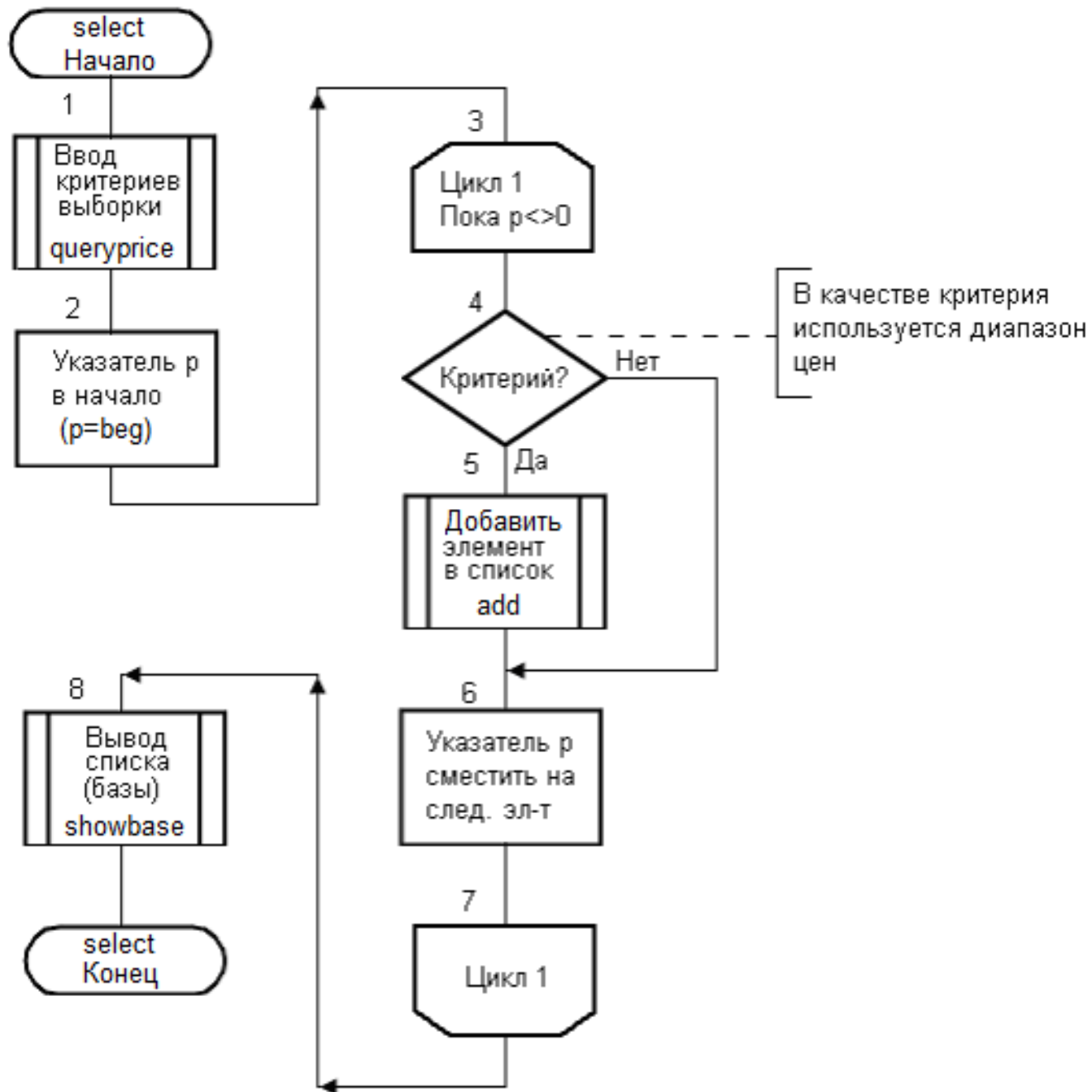
Каждый выход из символа должен сопровождаться соответствующими значениями условий, чтобы показать логический путь, который он представляет, с тем, чтобы эти условия и соответствующие ссылки были идентифицированы.



## Пример схемы алгоритма поиска НОД



## Пример схемы алгоритма



## Описание алгоритмов на псевдоязыке

**Псевдоязык** (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 стороны, в нём используются некоторые **формальные конструкции**, присущие формальным языкам, и **математическая символика**, что приближает запись алгоритма к общепринятой математической записи.

**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.

**Единого или формального определения псевдоязыка не существует**, поэтому возможны различные псевдоязыки, отличающиеся набором **служебных слов** и **основных (базовых) конструкций**.

**Мы будем использовать псевдоязык, приближенный к языку Паскаль.**

# Алгоритм поиска НОД на псевдоязыке

программа поиск\_НОД;

переменные

a, b, x, y, r: целый;

начало

ввод(a,b);

повторять

если  $a \geq b$  то начало

x:=a;

y:=b;

конец

иначе начало

x:=b;

y:=a;

конец

r:=x-y;

a:=y;

b:=r;

до r=0;

вывод(y);

конец.

# Разновидности структур алгоритмов

## *Теорема Бёма — Якопини*

Любой исполняемый алгоритм может быть преобразован к структурированному виду, то есть такому виду, когда ход его выполнения определяется только при помощи **трёх управляющих структур**:

- ✓ **следование** (англ. sequence);
- ✓ **ветвления** (англ. selection);
- ✓ **повторы или циклы** (англ. repetition, cycle).

Теорема была сформулирована и доказана итальянскими математиками Коррадо Бёмом (Corrado Böhm) и Джузеппе Якопини (Giuseppe Jacopini) в статье :

«Flow Diagrams, Turing Machines and Languages with Only Two Formation Rules». *Communications of the ACM* . – 1966. – **9** (5). – p. 366 –371.

В статье также описывались методы преобразования неструктурированных алгоритмов в структурированные.



# Линейные структуры алгоритмов

Такие структуры алгоритмов используют базовую структуру «следование», которая образуется последовательностью действий, следующих одно за другим и выполняющихся однократно.



действие 1;

действие 2;

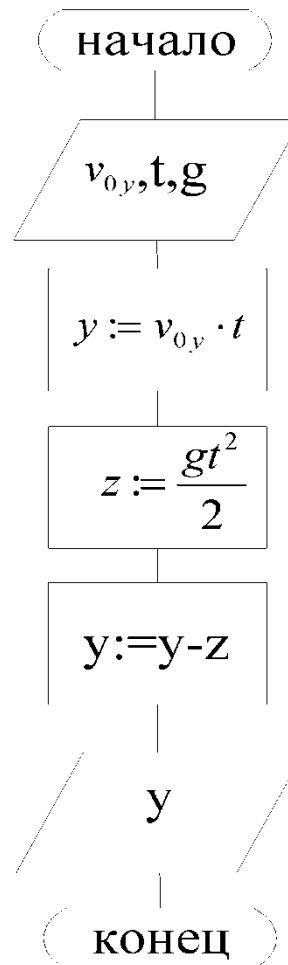
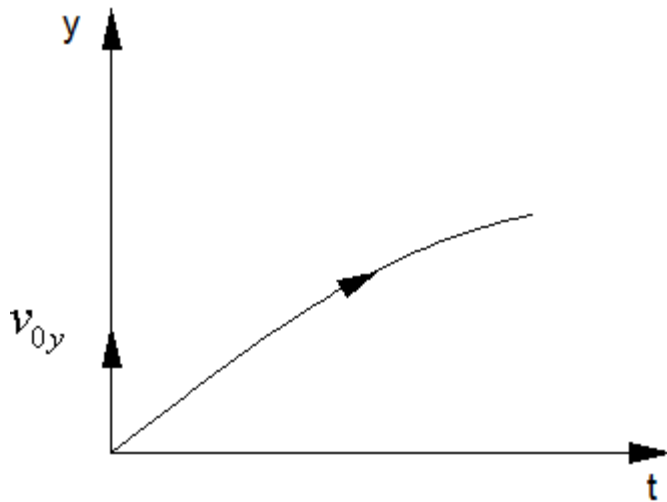
...

действие n;

## Линейные структуры алгоритмов

Пример: Пусть требуется для момента времени  $t$  вычислить высоту снаряда, выпущенного под углом к горизонту

$$y = V_{0y} \cdot t - \frac{gt^2}{2};$$



**ВВОД** ( $V_{0y}, t, g$ );  
 $y := V_{0y} \cdot t$ ;  
 $z := g \cdot t \cdot t / 2$ ;  
 $y := y - z$ ;  
**ВЫВОД** ( $y$ );