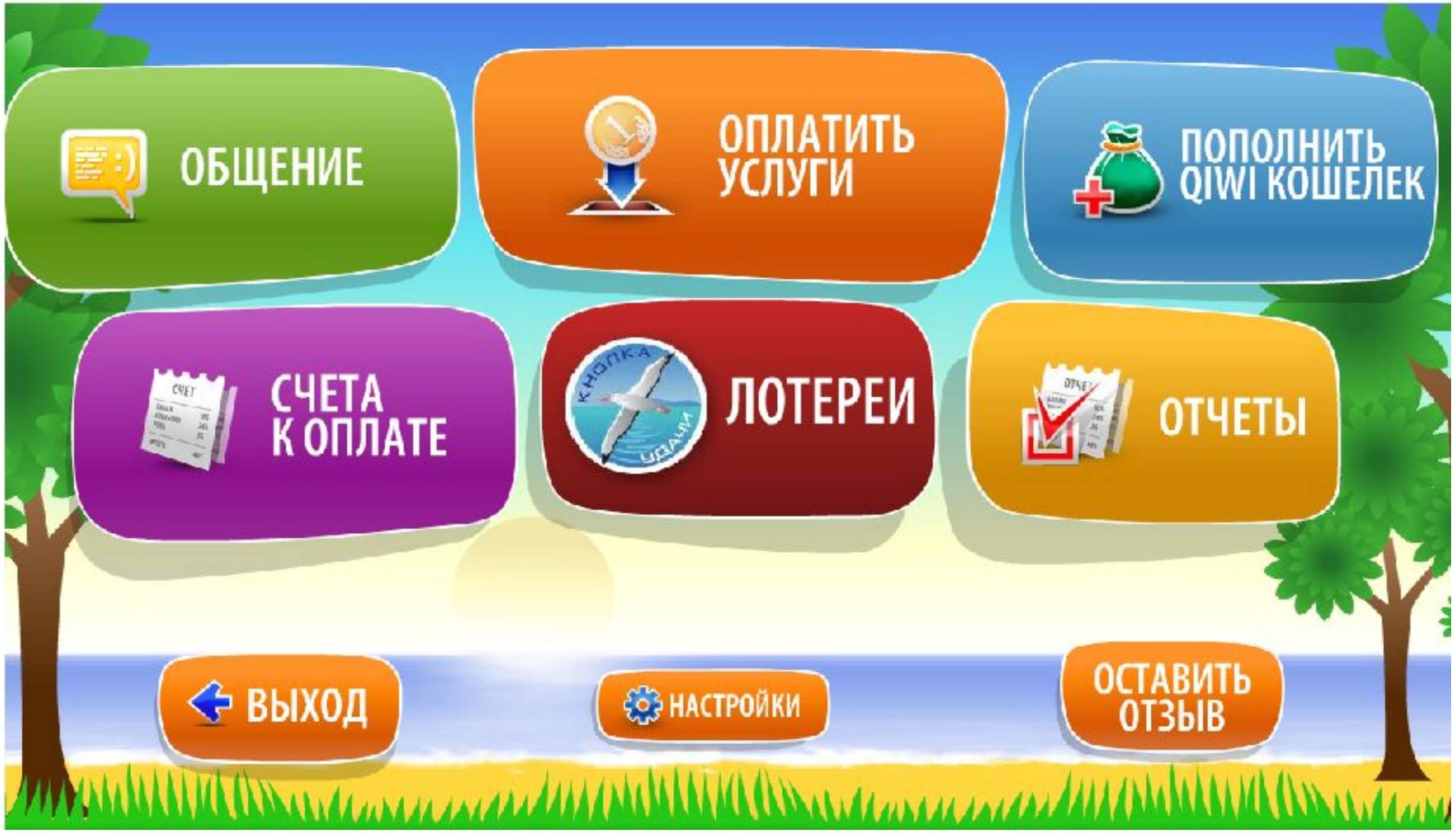
Примеры алгоритмов

Оплата через терминал

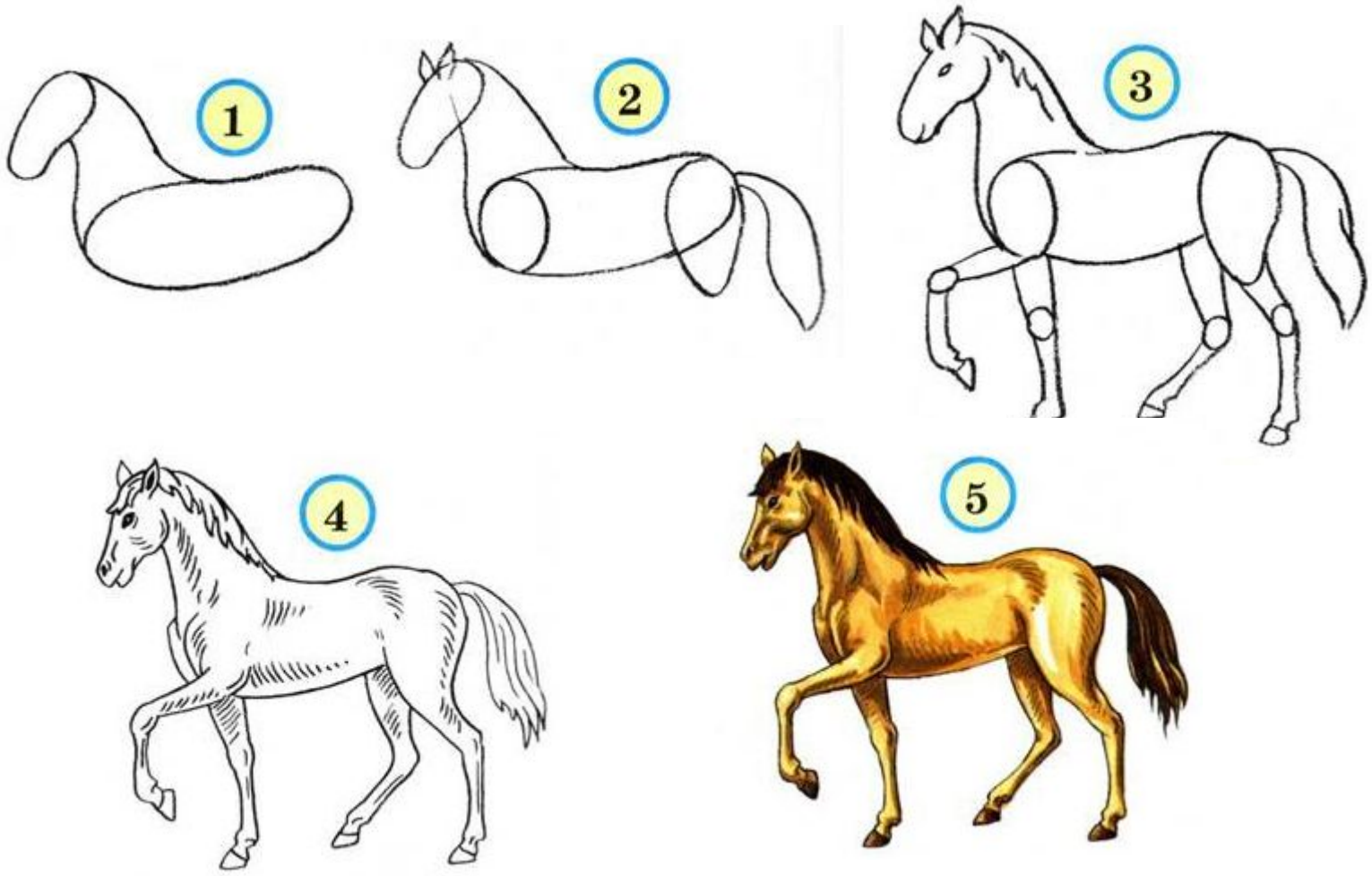


Выбрать пункт «Оплатить услуги»

равное введенной ранее в окне выбора суммы оплаты.

Примеры алгоритмов

Нарисовать лошадь



Примеры алгоритмов

Вычислительный алгоритм

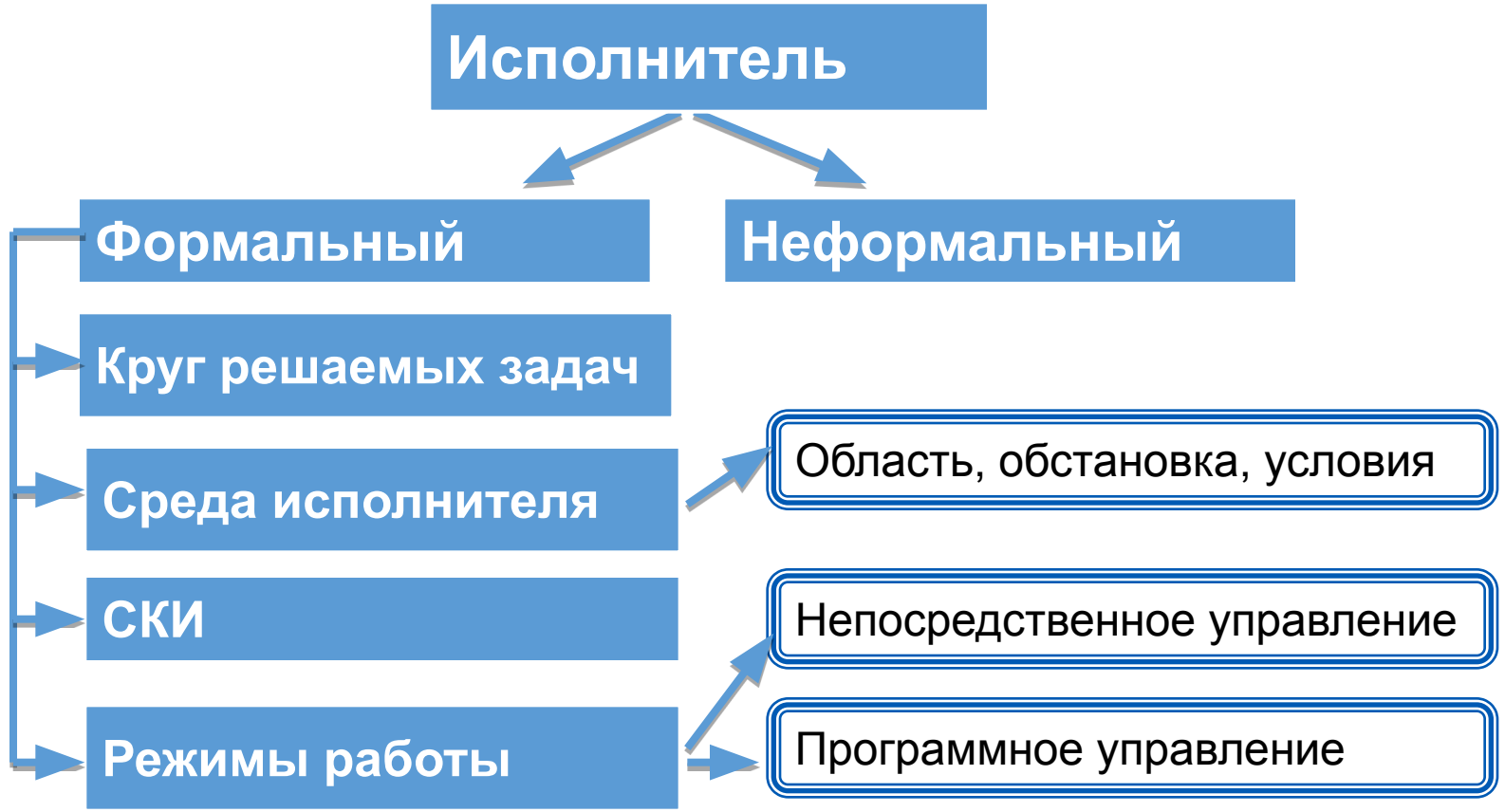
Среднее арифметическое двух чисел

1. **Задать два числа**
2. **Сложить заданные числа**
3. **Разделить сумму на 2**

Общая схема работы алгоритма

Исполнитель алгоритма

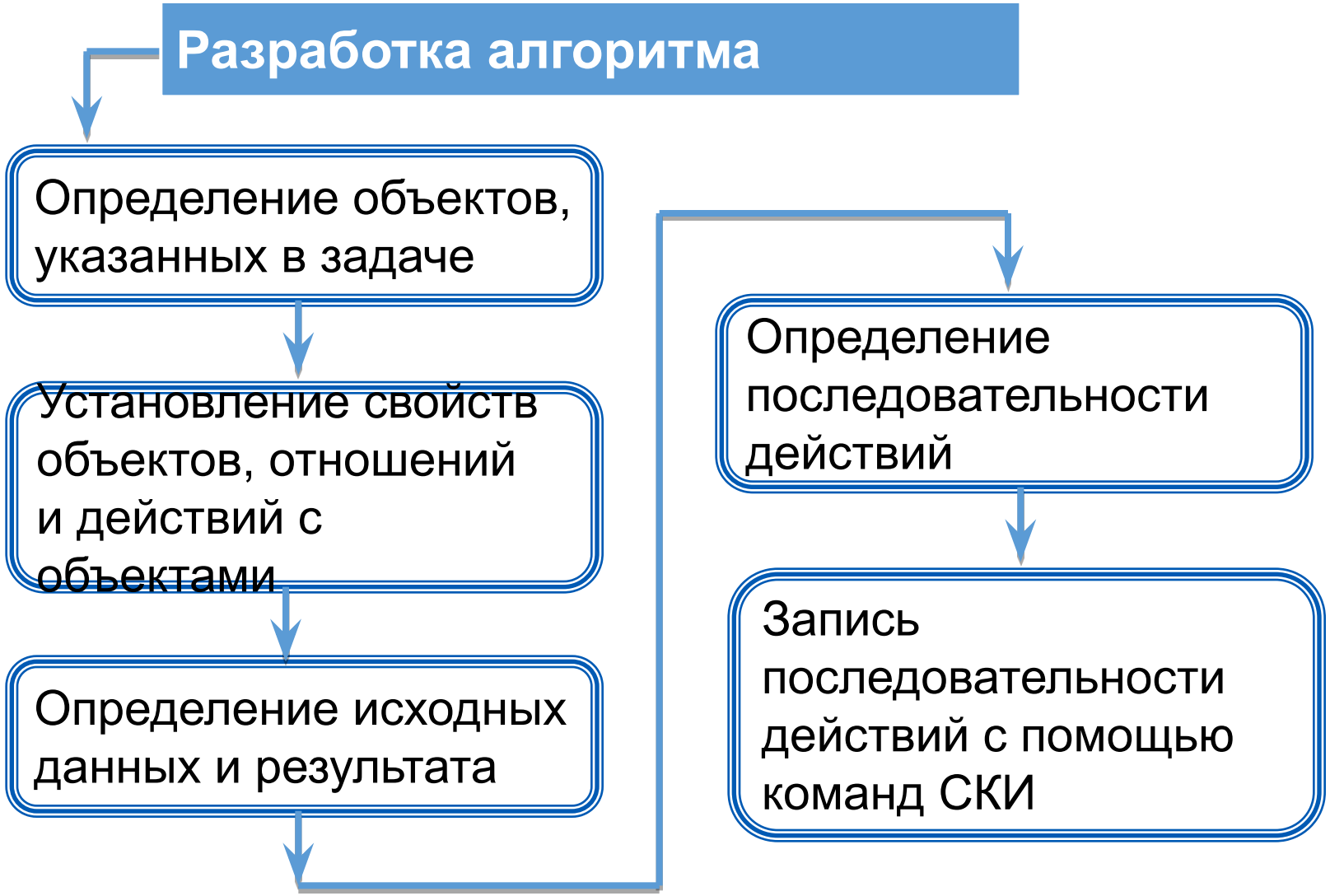
Исполнитель - это некоторый объект (человек, животное, техническое устройство), способный выполнять определённый набор команд.



Исполнители алгоритмов

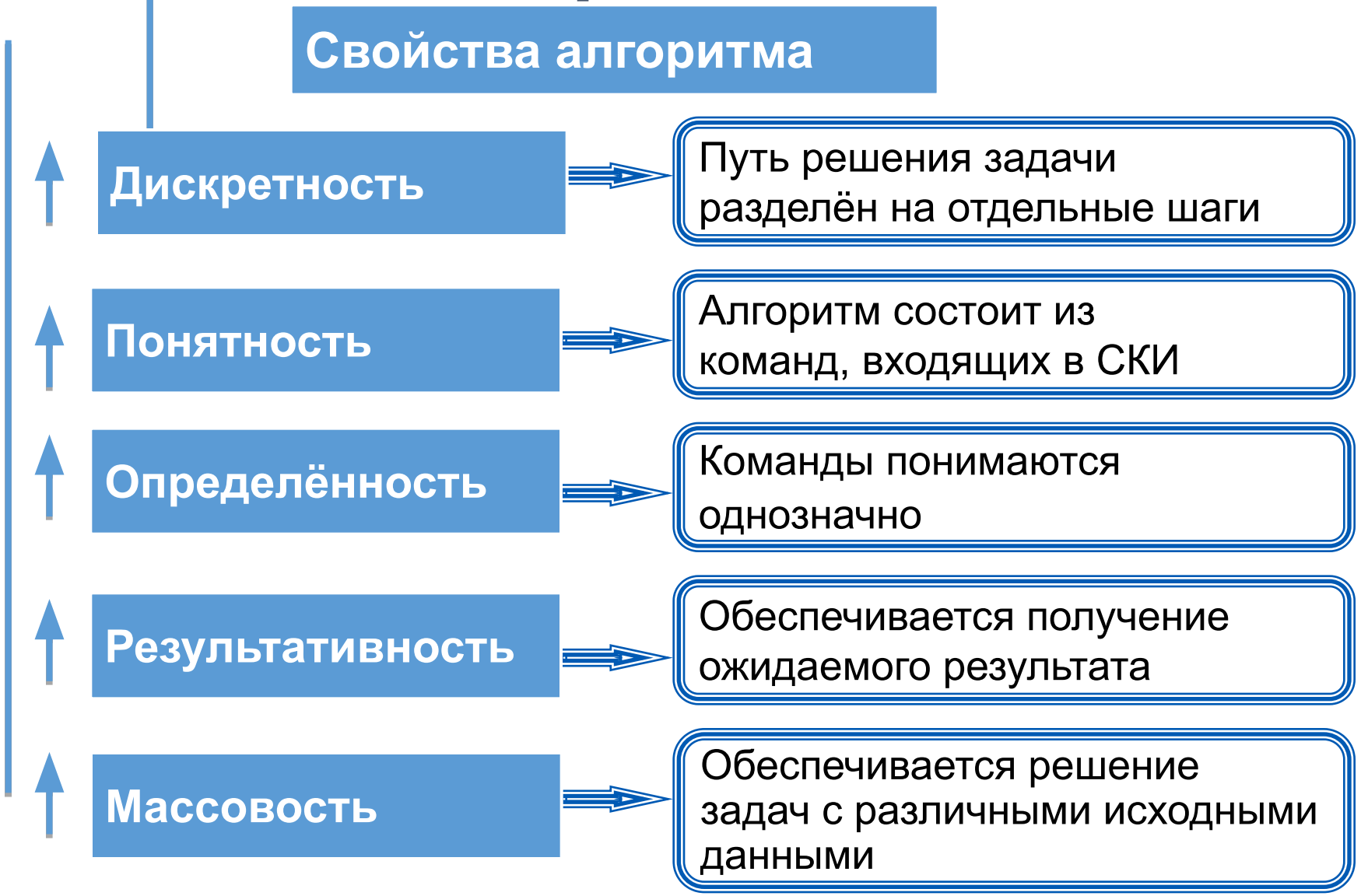
Ôàéë "SWF"

Разработка алгоритма

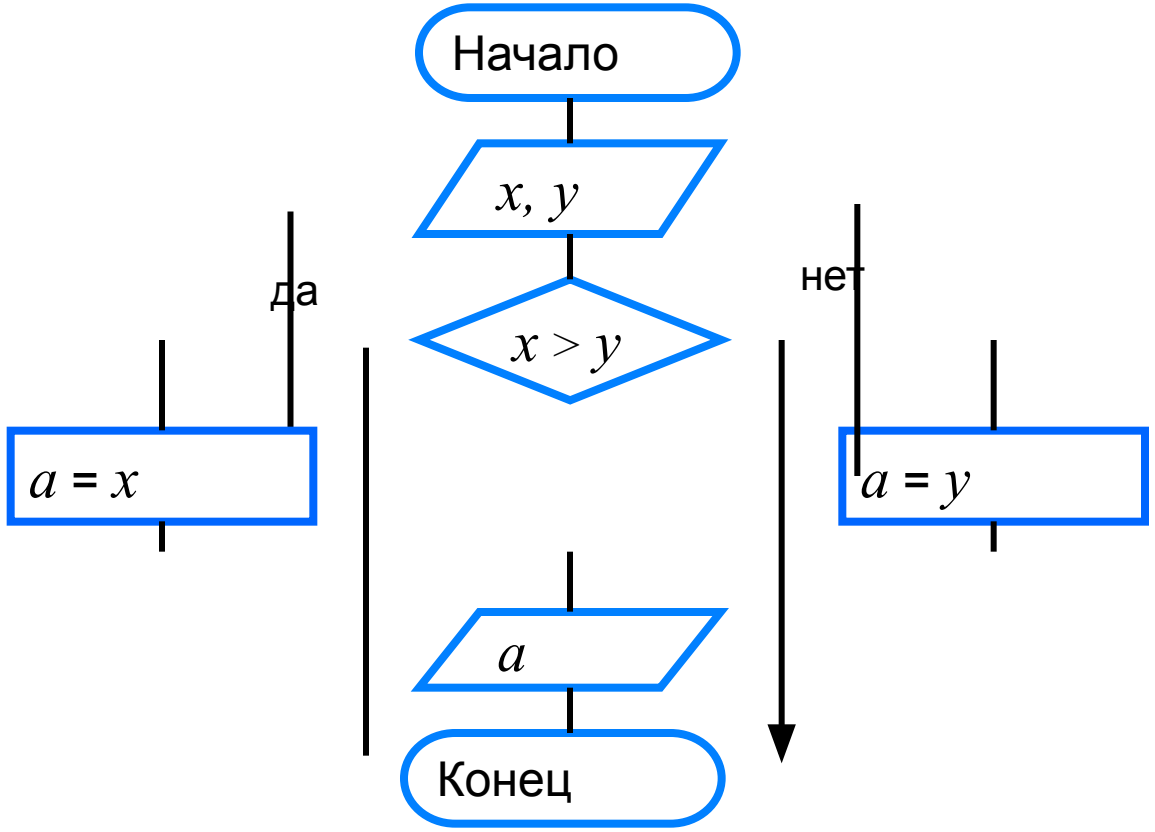


Алгоритм – модель деятельности исполнителя алгоритмов

Свойства алгоритма



Дискретность (от лат. discretus – разделенный, прерывистый) указывает, что любой алгоритм должен состоять из конкретных действий, следующих в определенном порядке. Образованная структура алгоритма оказывается дискретной: только выполнив одну команду, исполнитель сможет приступить к выполнению следующей.



Понятность означает, что алгоритм состоит только из команд, входящих в систему команд исполнителя, т. е. из таких команд, которые исполнитель может воспринять и по которым может выполнить требуемые действия.

Окрошка «Мясная»

-1.5 л кваса	300 г огурцов
500 г картофеля	зелень по вкусу
300 г колбасы	сметана
3 яйца	соль
200 г редиса	перец

Рецепт приготовления

Картофель отварить до готовности.
Остудить, почистить.
Нарезать кубиками.
Колбасу нарезать кубиками.
Яйца нарезать кубиками.
Редис тонко нарезать.
Огурцы нарезать кубиками.

Смешать картофель, колбасу, яйца, редис, огурцы.
Посолить, поперчить.
Выложить в тарелки.
Залить квасом, посыпать зеленью.
Подавать со сметаной.

Определённость означает, что в алгоритме нет команд, смысл которых может быть истолкован исполнителем неоднозначно; недопустимы ситуации, когда после выполнения очередной команды исполнителю неясно, какую команду выполнять на следующем шаге.

…оехать до стадиона

1. Идти прямо
2. Повернуть
3. Идти прямо
4. Сесть в автобус
5. Доехать до остановки «Стадион»

Алгоритм не уточняет, какое расстояние нужно пройти прямо.

В какую сторону повернуть.

В какой автобус сесть.

Результативность означает, что алгоритм должен обеспечивать возможность получения результата после конечного, возможно, очень большого, числа шагов. При этом результатом считается не только обусловленный постановкой задачи ответ, но и вывод о невозможности продолжения по какой-либо причине решения данной задачи.

Чт[illegible]е книги

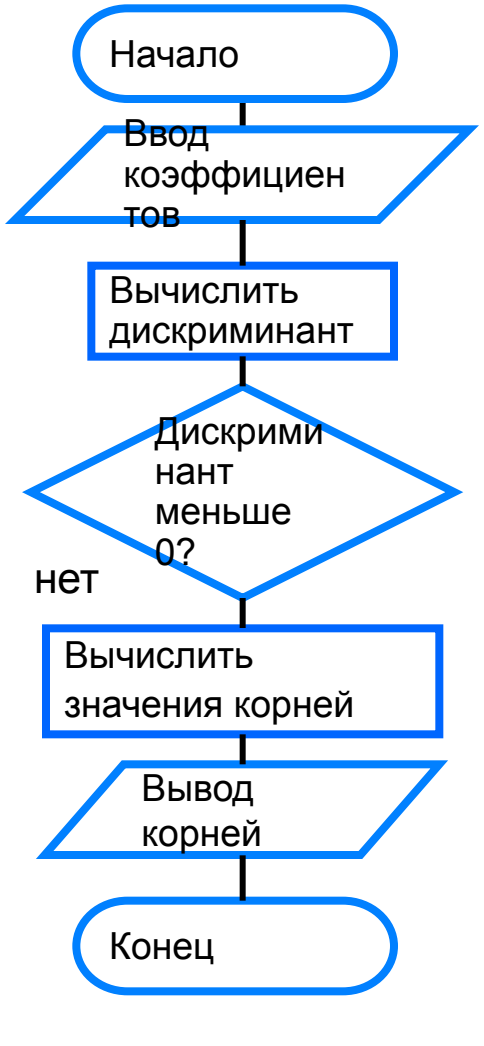
1. Взять книгу
2. Открыть первую страницу
3. Пока не конец книги выполнять следующие действия:
   - 3.1 Прочитать текст
   - 3.2 Перелистнуть страницу
   - 3.3 Прочитать текст
   - 3.4 Открыть первую страницу

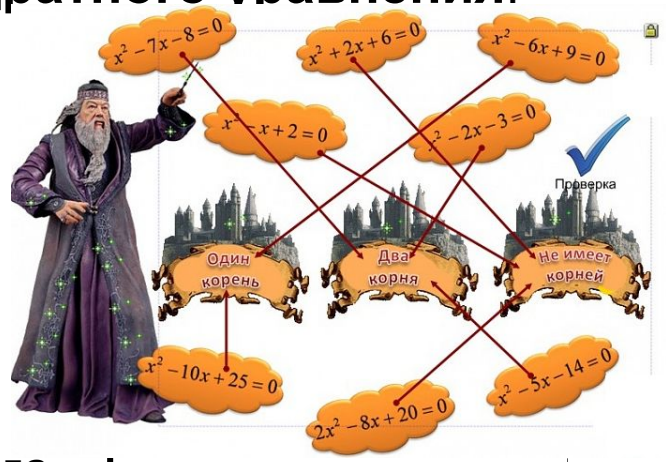
Данная последовательность команд не соответствует свойству результативности. Что нужно изменить?

Массовость означает, что алгоритм должен обеспечивать возможность его применения для решения любой задачи из некоторого класса задач с различными исходными данными.

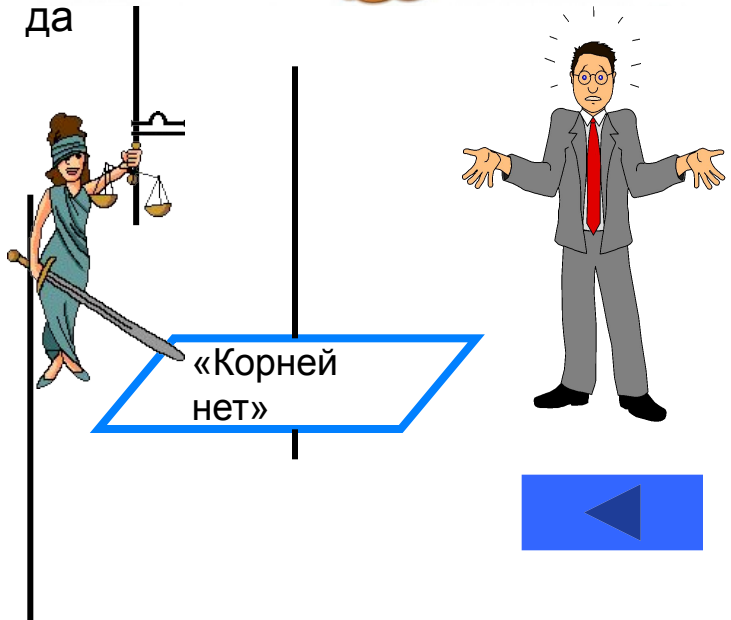
Алгоритм вычисления корней квадратного уравнения.

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$





да



Решето Эратосфена

Решето Эратосфена 

Рассмотренная последовательность действий является алгоритмом, так как она удовлетворяет свойствам:

- ***дискретности*** - процесс нахождения простых чисел разбит на шаги;
- ***понятности*** - каждая команда понятна ученику 9 класса, выполняющему этот алгоритм;
- ***определённости*** - каждая команда трактуется и выполняется исполнителем однозначно; имеются указания об очерёдности выполнения команд;
- ***результативности*** - через некоторое число шагов достигается результат;
- ***массовости*** - последовательность действий применима для любого натурального n.

Алгоритм **-** это предназначенное для конкретного исполнителя описание последовательности действий, приводящих от исходных данных к требуемому результату, которое обладает свойствами**:**

- ***дискретности***
- ***понятности***
- ***определённости***
- ***результативности***
- ***массовости***

Возможности автоматизации деятельности человека

Решение задачи по готовому алгоритму требует от исполнителя только строгого следования заданным предписаниям.

Формального исполнения алгоритма обеспечивает возможность автоматизации деятельности человека

↓

Процесс решения задачи представляется в виде последовательности операций

↓

Создается машина, способная выполнять эти операции в указанной последовательности

↓

Человек освобождается от рутинной работы, выполнение которой поручается автомату

Автоматизация деятельности человека

Исполнитель - некоторый объект (человек, животное, техническое устройство), способный выполнять определённый набор команд.

Формальный исполнитель одну и ту же команду всегда выполняет одинаково. Для каждого формального исполнителя можно указать: круг решаемых задач, среду, систему команд и режим работы.

Алгоритм - предназначенное для конкретного исполнителя описание последовательности действий, приводящих от исходных данных к требуемому результату, которое обладает свойствами дискретности, понятности, определённости, результативности и массовости.

Способность исполнителя действовать формально обеспечивает возможность автоматизации деятельности человека.

Опорный конспект

Алгоритм - это предназначенное для конкретного исполнителя описание последовательности действий, приводящих от исходных данных к требуемому результату, которое обладает свойствами ***дискретности, понятности, определённости, результативности и массовости***.

Марков А.А. (1903—1979) установил, что алгоритмы должны содержать предписания двух видов:

1) ***функциональные операторы -*** предписания, направленные на непосредственное преобразование информации;

2) ***логические операторы -*** предписания, определяющие дальнейшее направление действий.

Именно эти операторы положены в основу большинства способов записи алгоритмов.

Основные способы записи алгоритма

Словесные	Графические	На алгоритмических языках
Словесное описание	Последователь-ность рисунков	Школьный алгоритмический язык
Построчная запись	Структурограмма	Язык программирова-ния
	Блок-схема	

Словесное описание

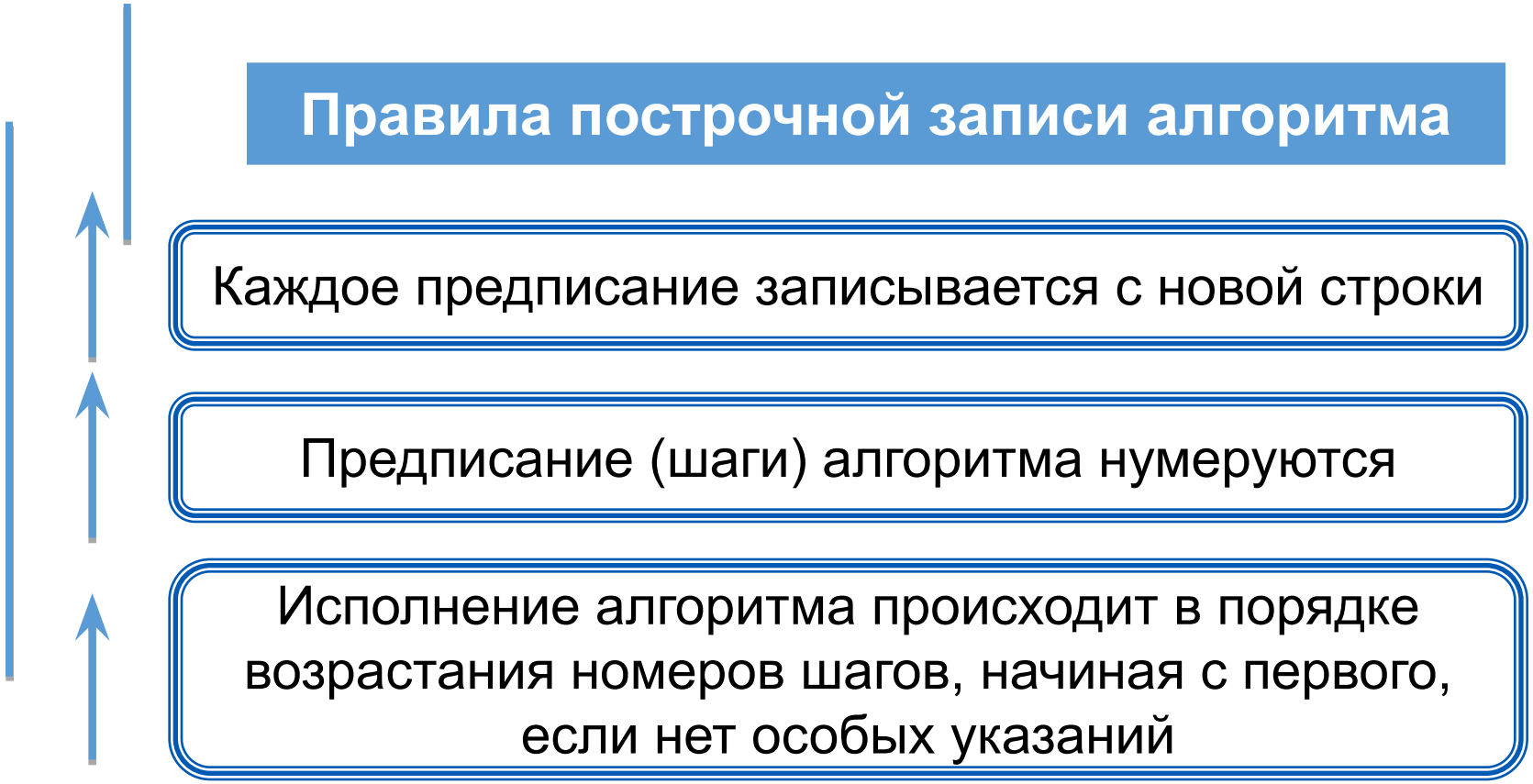
Словесное описание - самая простая запись алгоритма в виде набора высказываний на обычном разговорном языке.

Пример. Словесное описание алгоритма нахождения наибольшего общего делителя (НОД) пары целых чисел (алгоритм Евклида).

Чтобы найти НОД двух чисел, составьте таблицу из двух столбцов и назовите столбцы X и Y. Запишите первое из заданных чисел в столбец X, а второе - в столбец Y. Если данные числа не равны, замените большее из них на результат вычитания из большего числа меньшего.

Повторяйте такие замены до тех пор, пока числа не окажутся равными, после чего число из столбца X считайте искомым результатом.

Построчная запись



Кроме слов естественного языка предписания могут содержать математические выражения и формулы.

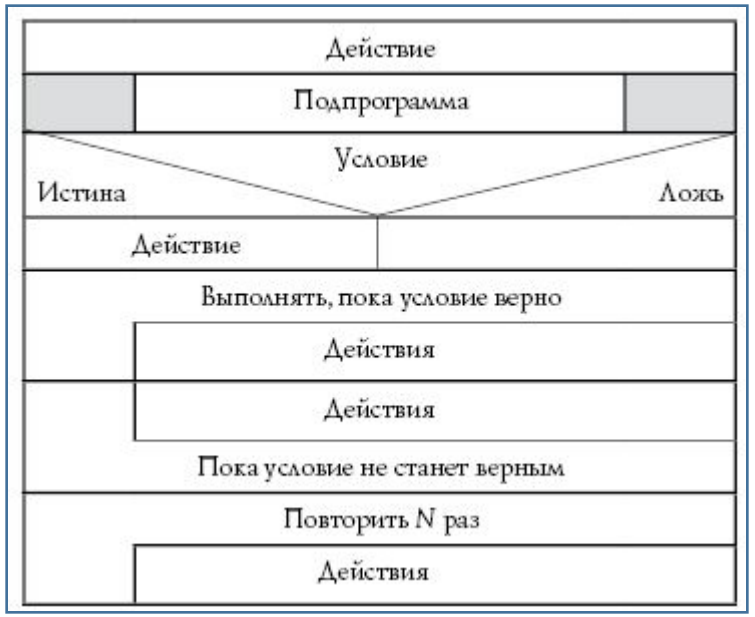
Построчная запись алгоритма Евклида

Построчная запись алгоритма Евклида

1. Начало.
2. Обозначить первое из заданных чисел X, второе - Y.
3. Если $X = Y$ то перейти к п. 9.
4. Если $X > Y$, то перейти к п. 5, иначе перейти к п. 7.
5. Заменить X на X-Y.
6. Перейти к п. 3.
7. Заменить Y на Y-X
8. Перейти к п. 3.
9. Считать X искомым результатом.
10. Конец.

Графические способы

Последовательные картинки



Структурограмма

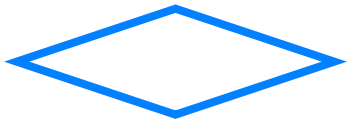
Блок-схемы

В блок-схеме предписания изображаются с помощью различных геометрических фигур, а последовательность выполнения шагов указывается с помощью линий.

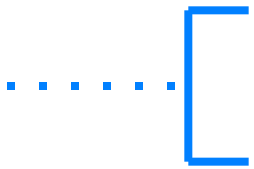
Блок начала или конца алгоритма

Блок ввода или вывода данных

Блок обработки данных

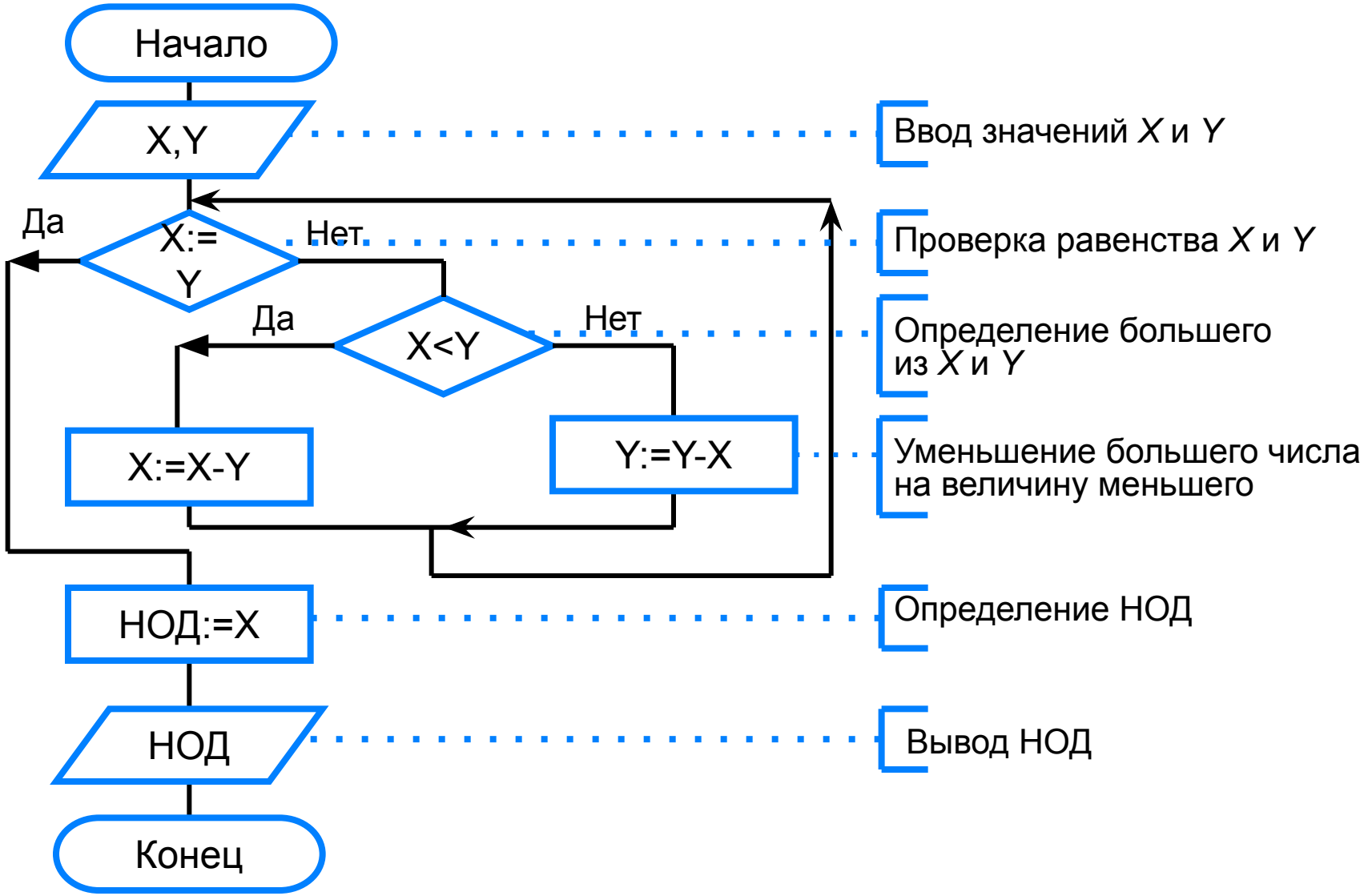


Блок проверки условия



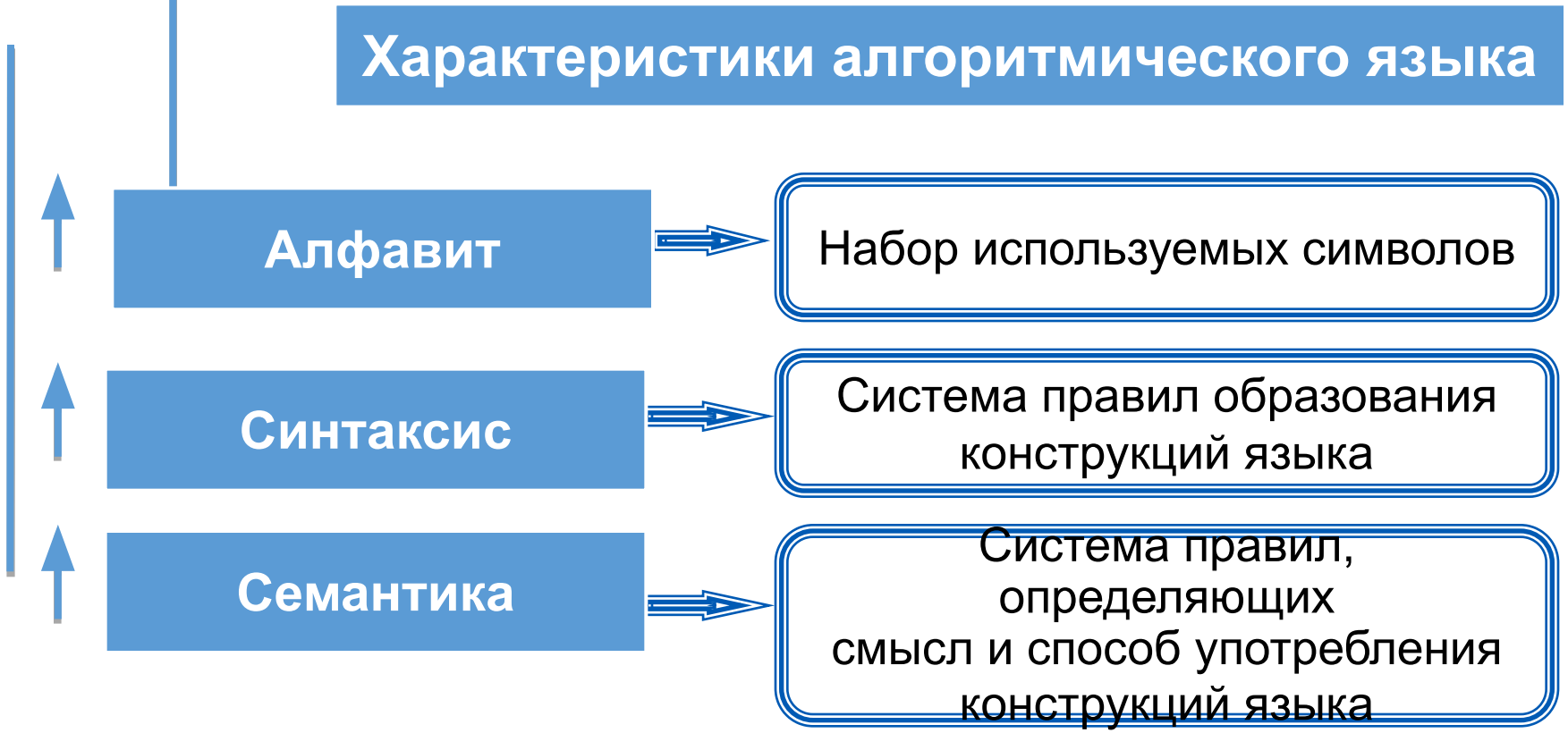
Блок пояснительных записей

Запись алгоритма Евклида с помощью блок-схемы



Алгоритмические языки

Алгоритмические языки - формальные языки, предназначенные для записи алгоритмов.

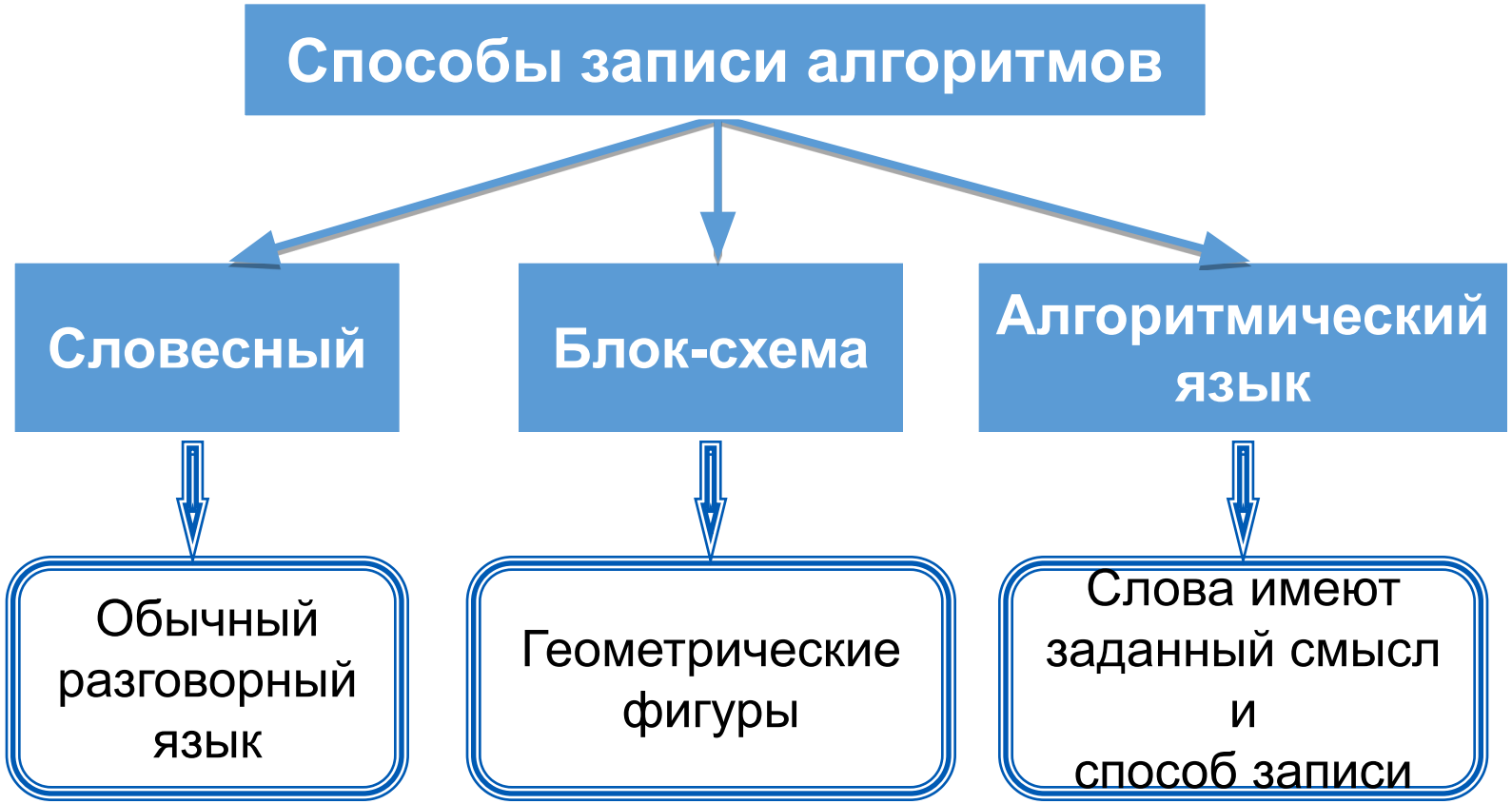


Существуют различные способы записи алгоритмов:

- ***словесное описание***
- ***построчная запись***
- ***блок-схема***
- ***школьный алгоритмический язык*** и другие.

Каждый из этих способов обладает своими достоинствами и недостатками.

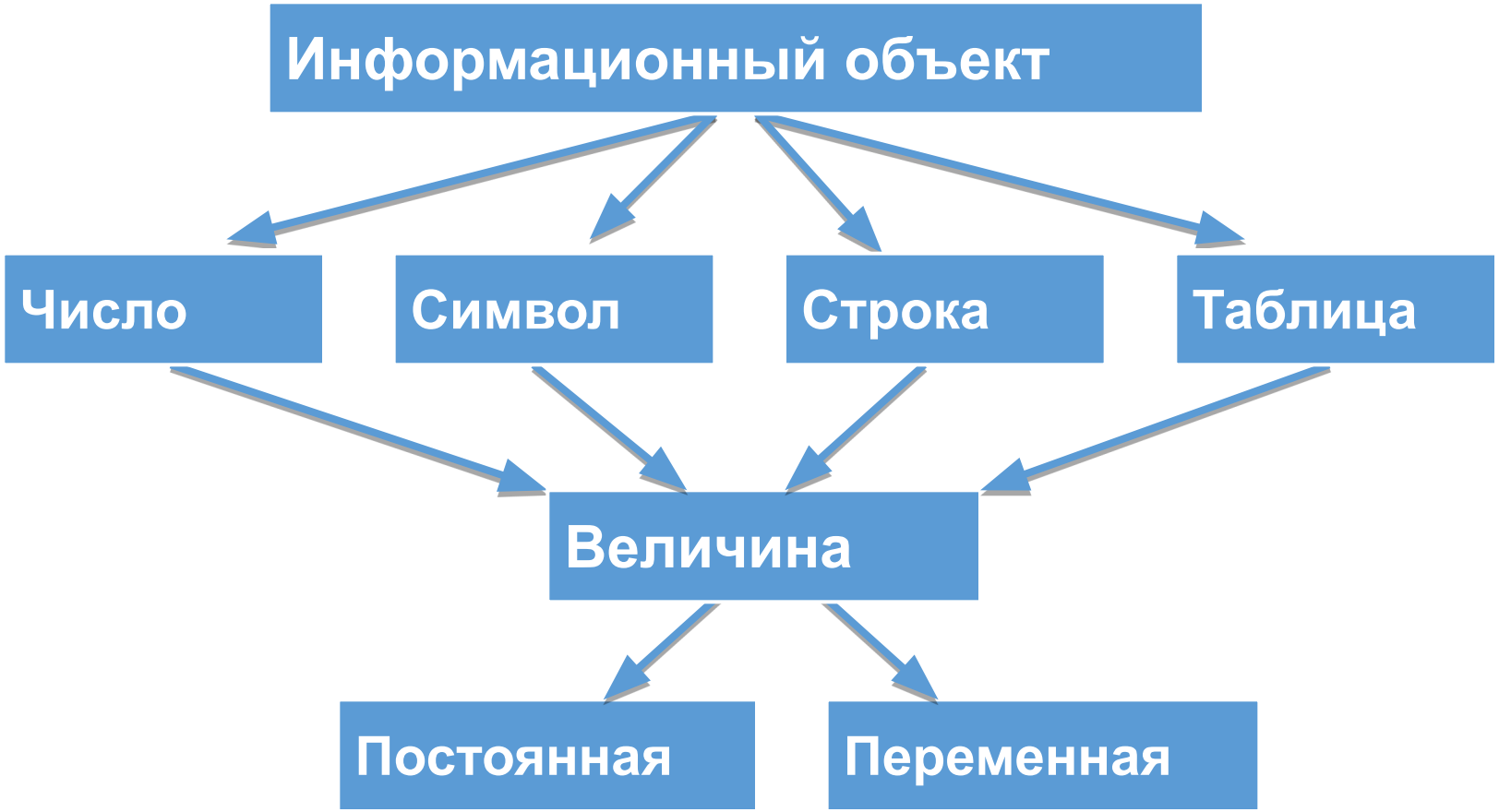
Опорный конспект



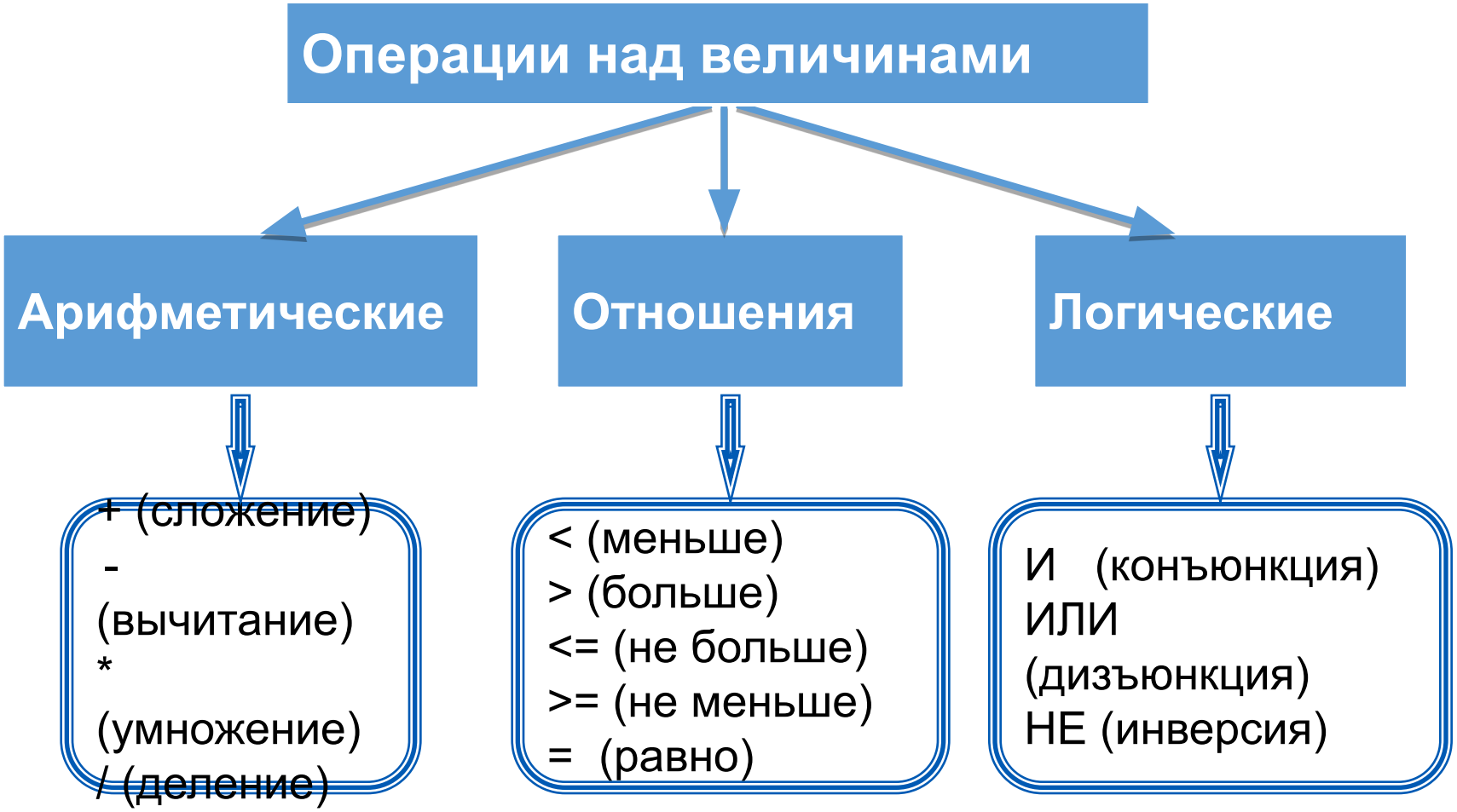
Величины

Алгоритмы описывают последовательность действий над некоторыми ***информационными объектами***.

Величина в информатике – это отдельный информационный объект.

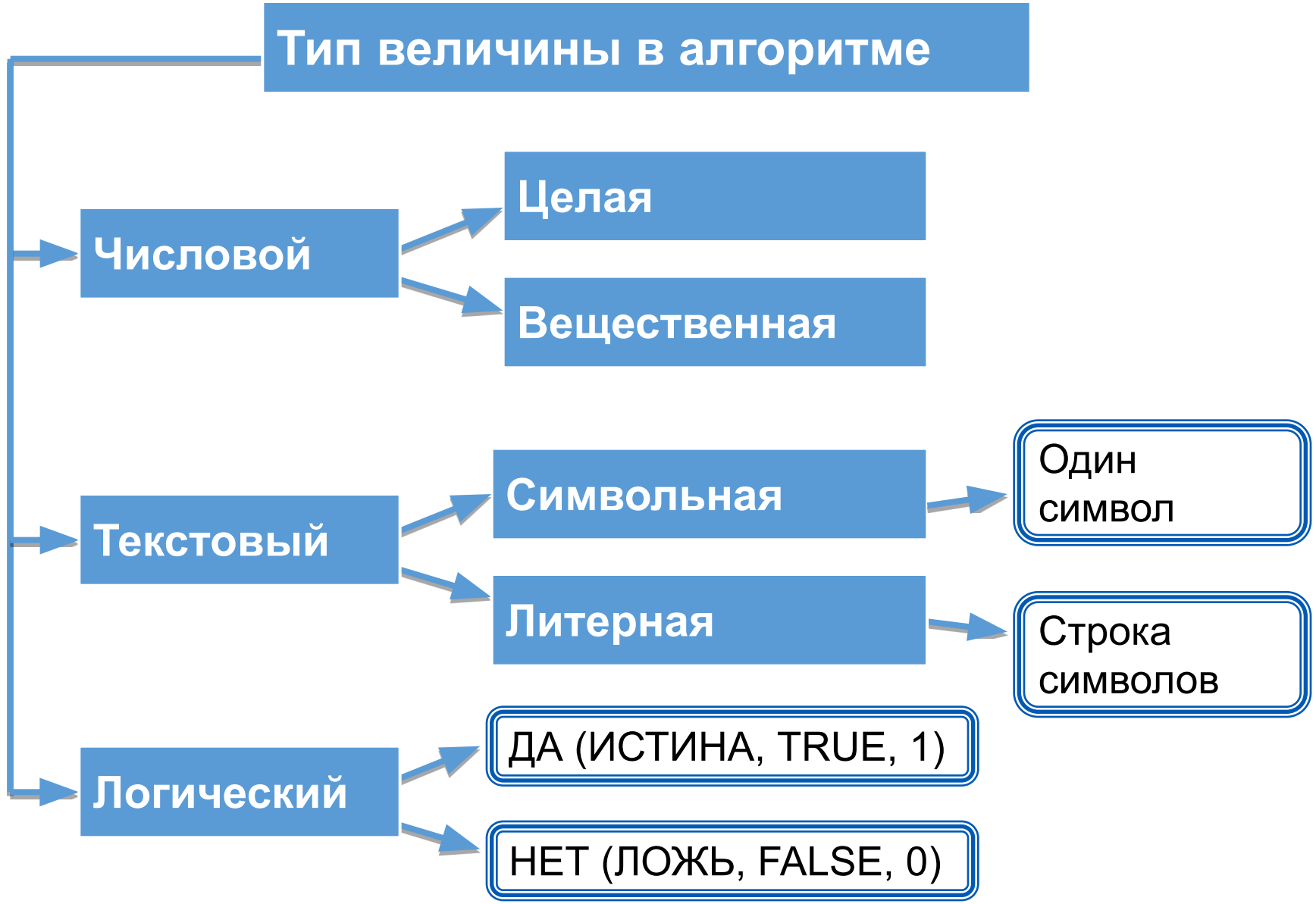


Операции над величинами

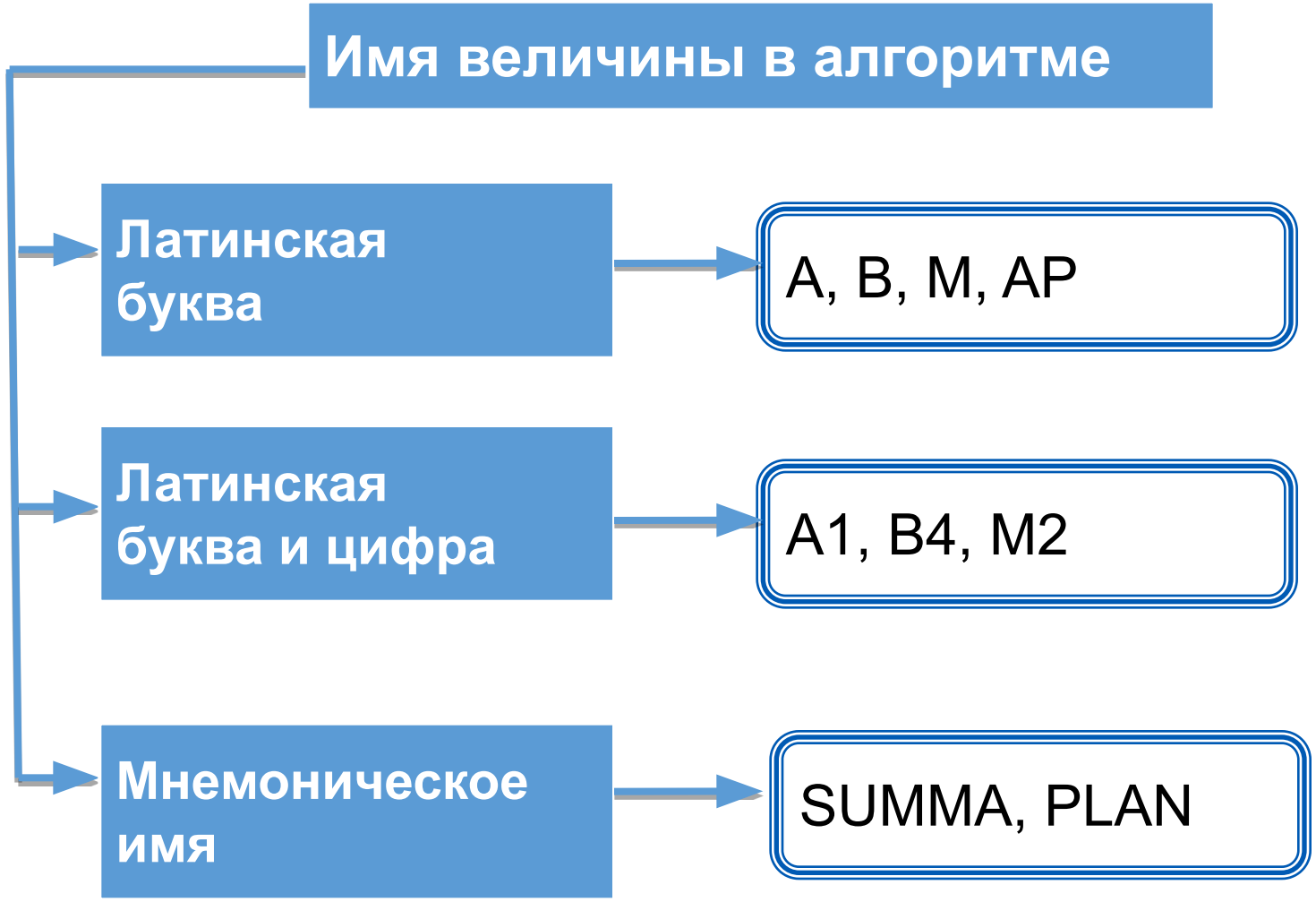


Операнды - объекты, над которыми выполняют операции.

Типы величин

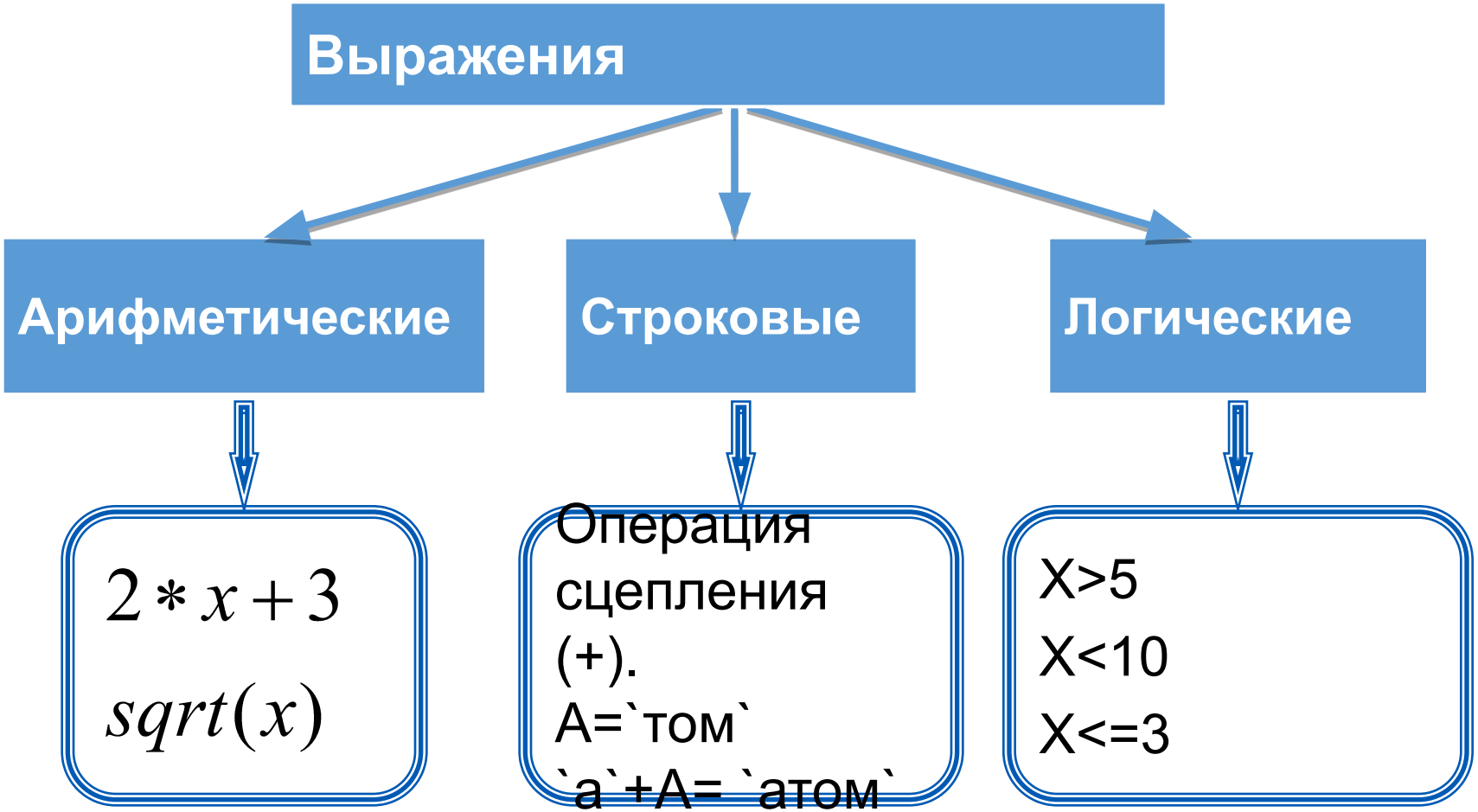


Имя величины



Выражения

Выражение - языковая конструкция для вычисления значения с помощью одного или нескольких операндов.



Команда присваивания

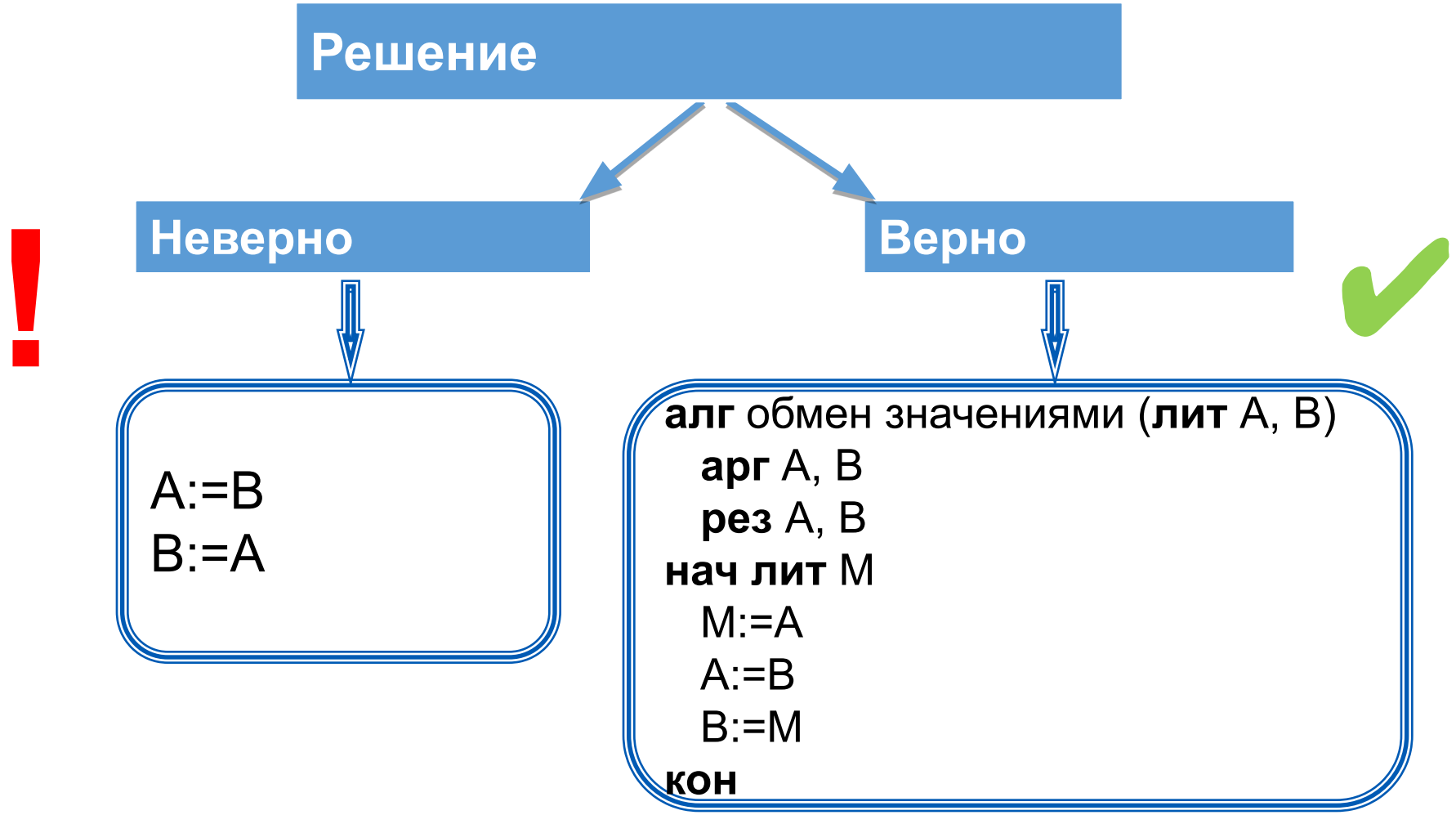
<имя переменной>:= <выражение>

Свойства присваивания

- Пока переменной не присвоено значение, она остаётся неопределённой
- Значение, присвоенное переменной, сохраняется до следующего присваивания
- Если переменной присваивается новое значение, то предыдущее её значение теряется

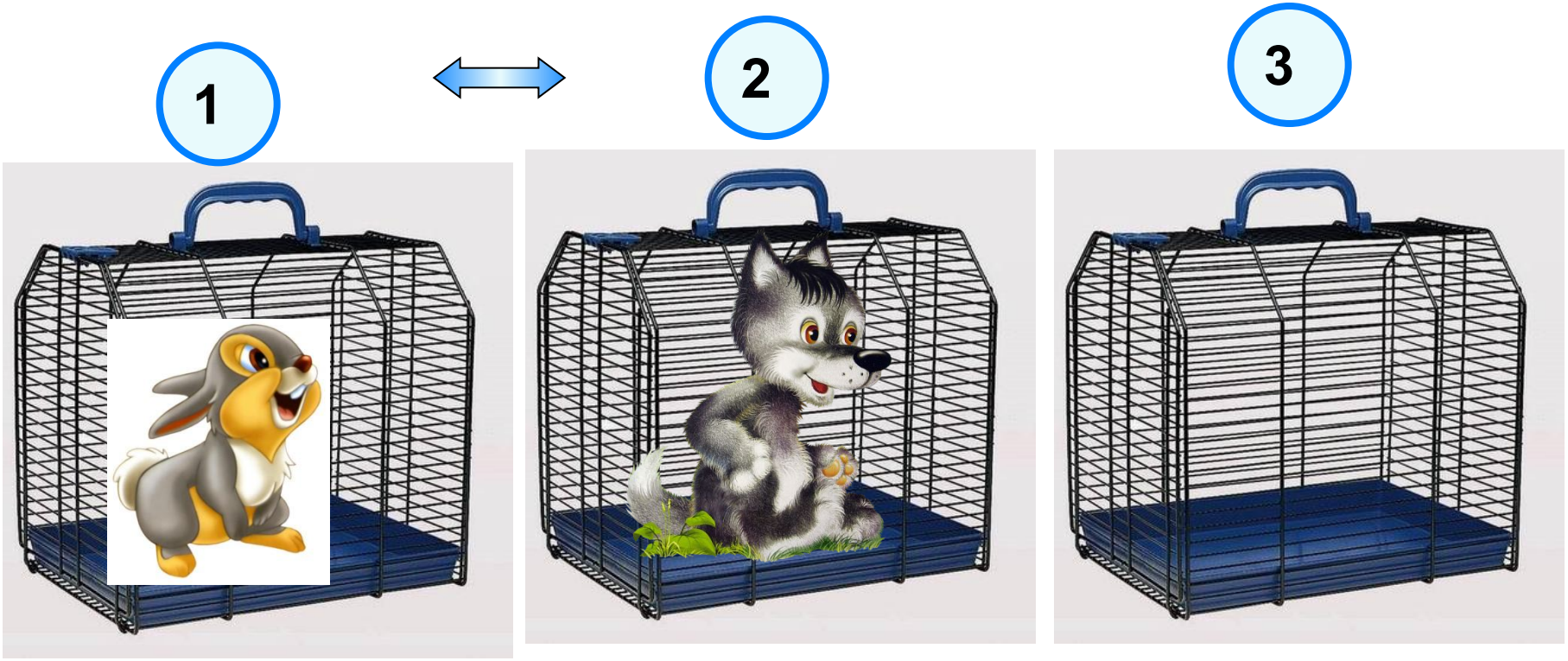
Алгоритм обмена значений переменных

Алгоритм, в результате которого переменные A и B литерного типа обменяются своими значениями.



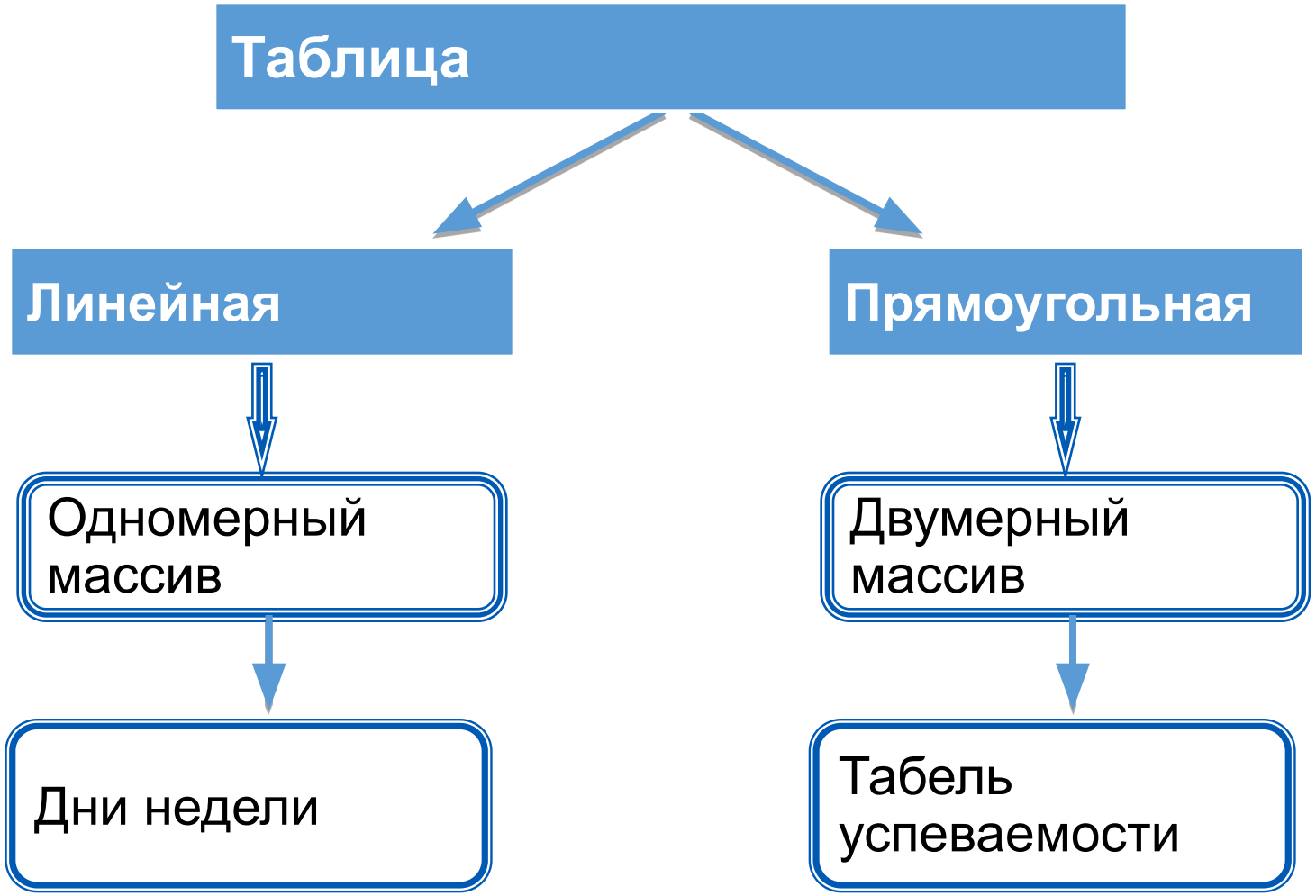
Аналогия с перемещением

Алгоритм перемещения зайца из клетки 1 в клетку 2, а волка - из клетки 2 - в клетку 1. Нужна клетка 3.

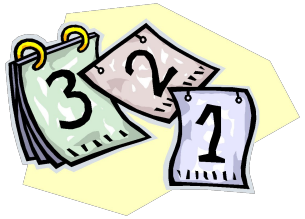


Табличные величины

В практической деятельности человека часто используются всевозможные таблицы.



Примеры линейных таблиц



1	Понедельник
2	Вторник
3	Среда
4	Четверг
5	Пятница
6	Суббота
7	Воскресенье

Дни недели

	1	2	3	4	5
Васечкин	6	6	1	0	0

Количество пропущенных учеником уроков

Пример прямоугольной таблицы

		1	2	3	4	5
1	Васечкин	6	6	1	0	0
2	Ионов	0	0	0	0	6
3	Радугина	0	0	1	0	0
.		.	.	.	.	.
.		.	.	.	.	.
.		.	.	.	.	.
19	Чабанюк	0	0	0	0	0

Количество уроков, пропущенных учениками класса

Величина в информатике – это отдельный информационный объект (число, символ, строка, таблица и др.).

Величины делятся на:

постоянные - значения указываются в тексте алгоритма и не меняются в процессе его исполнения

переменные - значения меняются в процессе исполнения алгоритма.

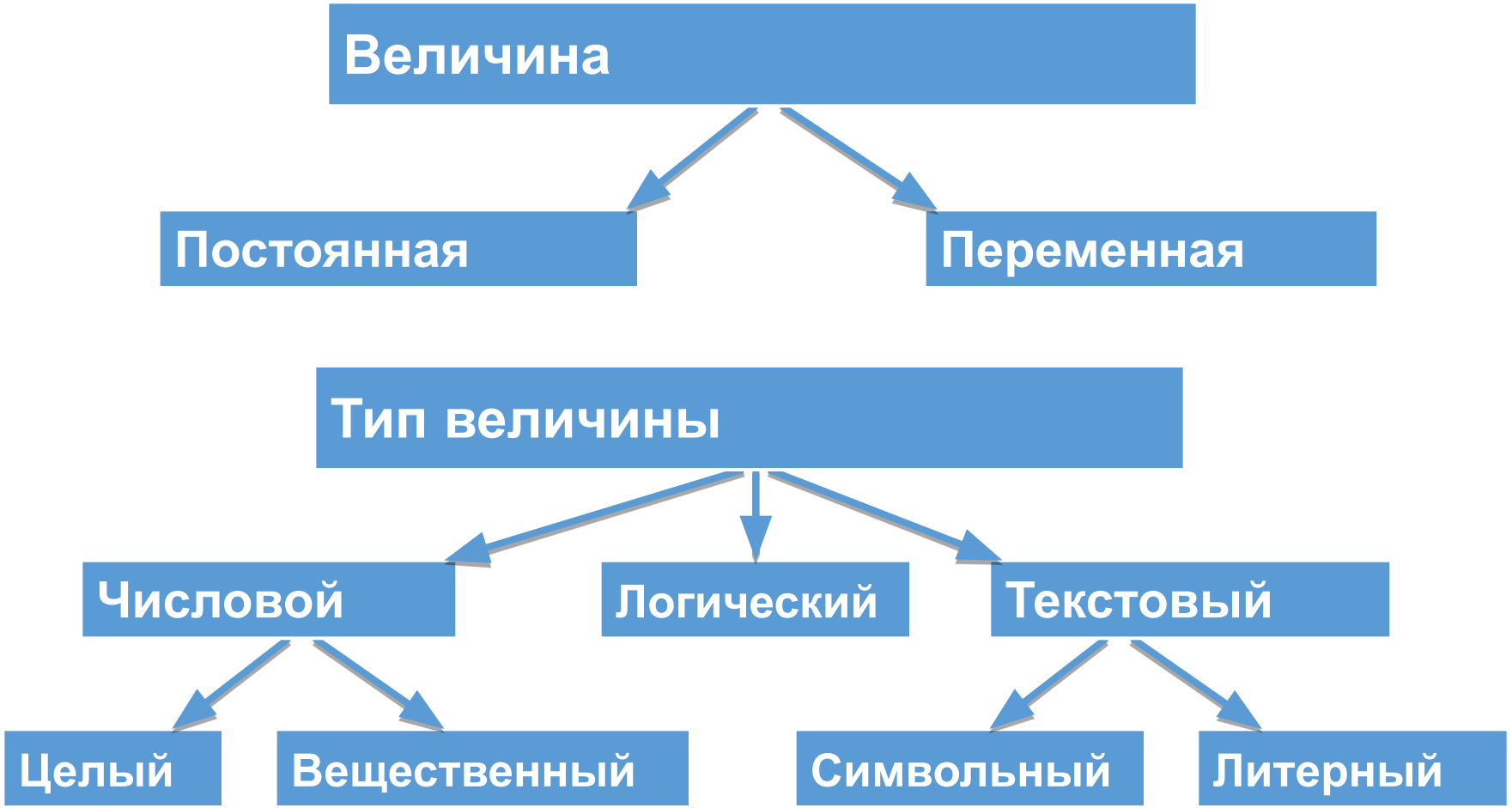
Тип величины: целый, вещественный, логический, символьный и литерный.

Для ссылок на величины используют их ***имена*** (идентификаторы). Имя величины может состоять из одной или нескольких латинских букв, из латинских букв и цифр.

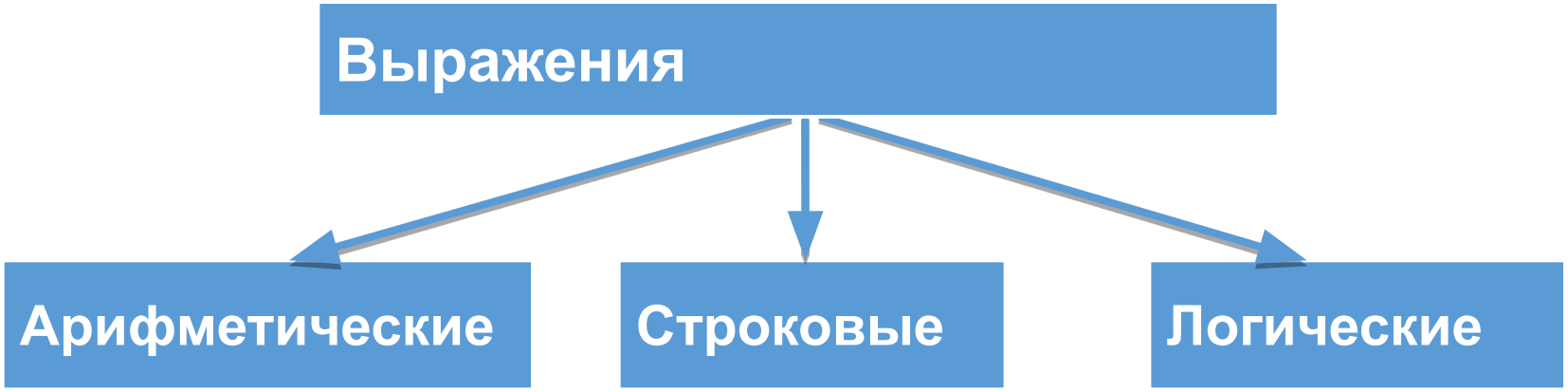
Таблица (массив) - набор некоторого числа однотипных элементов, которым присвоено одно имя. Положение элемента в таблице однозначно определяется его индексами.

Опорный конспект

Величина в информатике – это отдельный информационный объект (число, символ, строка, таблица и др.).



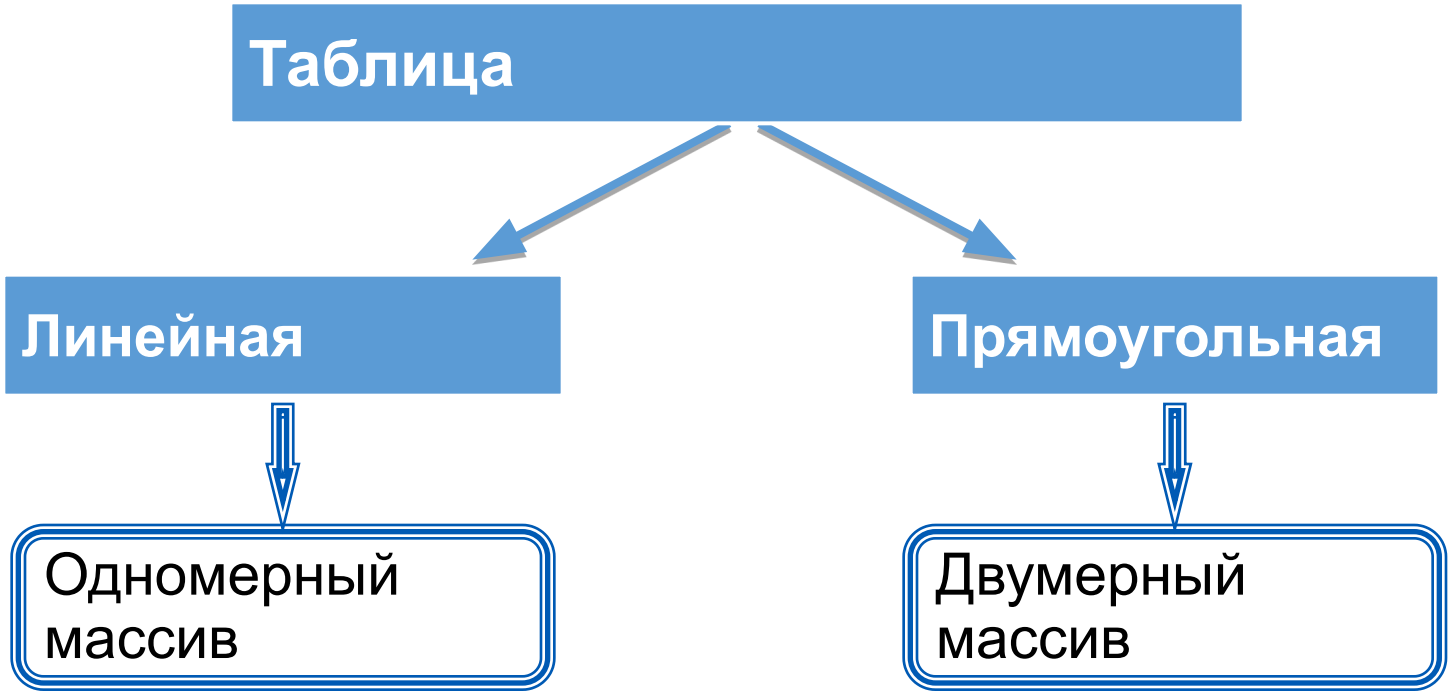
Опорный конспект



Команда присваивания

<имя переменной>:= <выражение>

Опорный конспект



Основные алгоритмические конструкции

Для записи любого алгоритма достаточно трёх основных алгоритмических конструкций:

- следования,
- ветвления,
- Повторения.

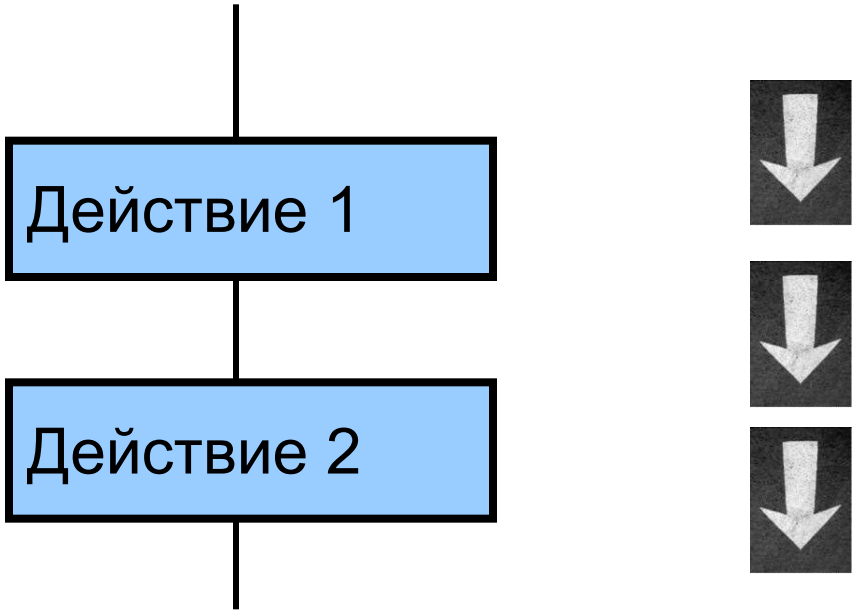
(*Э. Дейкстра*)

Эдсгер Вибе Дейкстра (1930–2002). Выдающийся нидерландский учёный, идеи которого оказали огромное влияние на развитие компьютерной индустрии.

Следование

Следование - алгоритмическая конструкция, отображающая естественный, последовательный порядок действий.

Алгоритмы, в которых используется только структура «следование», называются **линейными алгоритмами**.



Алгоритмическая структура «следование»

Линейный алгоритм приготовления отвара шиповника

Вычисления по алгоритму

Алгоритм

$x:=2$

$y:=x*x$

$y:=y*y$

$x:=y*x$

$s:=x+y$

Шаг алгоритма	Переменные		
	x	y	s
1	2	-	-
2	2	4	-
3	2	16	-
4	32	16	-
5	32	16	48

Ответ: $s = 48$

Целочисленная арифметика

С помощью операции **div** вычисляется целое частное, с помощью операции **mod** - остаток.

7 : 3 = 2 (ост.1)	7 **div** 3 = 2	7 **mod** 3 = 1
8 : 3 = 2 (ост.2)	8 **div** 3 = 2	8 **mod** 3 = 2
10: 3 = 3 (ост.1)	10 **div** 3 = 3	10 **mod** 3 = 1
13 : 4 = 3 (ост.1)	13 **div** 4 = 3	13 **mod** 4 = 1
11 : 4 = 2 (ост.3)	11 **div** 4 = 2	11 **mod** 4 = 3
8 : 3 = 2 (ост.2)	8 **div** 3 = 2	8 **mod** 3 = 2

Алгоритм работы кассира

Алгоритм работы кассира, выдающего покупателю сдачу (s) наименьшим количеством банкнот по 500 ($k500$), 100 ($k100$), 50 ($k50$) и 10 ($k10$) рублей.

```
k500:=s div 500
s:=s mod 500
k100:=s div 100
s:=s mod 100
k50:=s div 50
s:=s mod 50
k10:=s div 10
```

Линейный алгоритм

Package

Для записи любого алгоритма достаточно трёх основных алгоритмических конструкций (структур): ***следования, ветвления, повторения.***

Следование - алгоритмическая конструкция, отображающая естественный, последовательный порядок действий.

Алгоритмы, в которых используется только структура «следование», называются ***линейными***.

Опорный конспект

Следование - алгоритмическая конструкция, отображающая естественный, последовательный порядок действий.

Алгоритмы, в которых используется только структура «следование», называются ***линейными***.

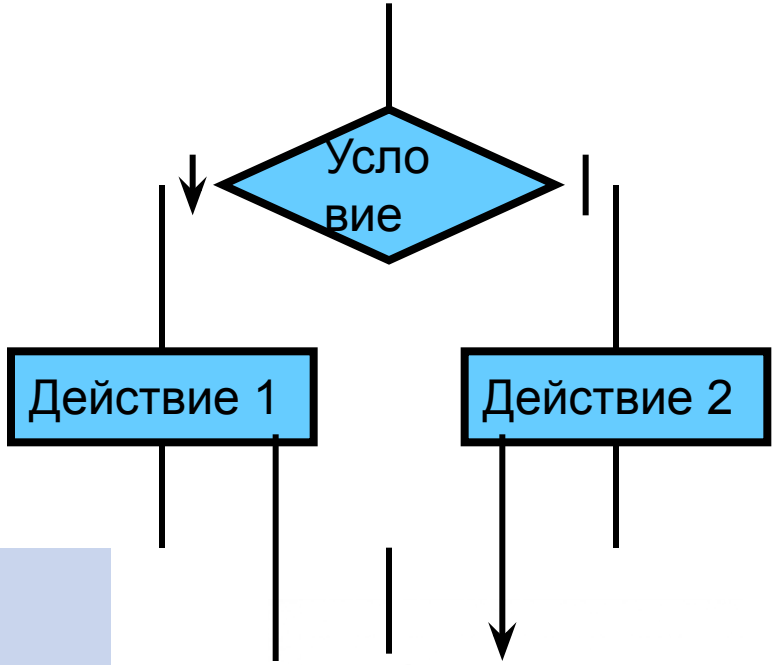
Ветвление

Ветвление - алгоритмическая конструкция, в которой в зависимости от результата проверки условия («да» или «нет») предусмотрен выбор одной из двух последовательностей действий (ветвей).

Алгоритмы, в основе которых лежит структура «ветвление», называют **разветвляющимися.**

Полная форма ветвления

если <условие>
то <действия 1>
иначе <действия 2>
все



Пример

алг правописание частиц НЕ, НИ
нач
если частица под ударением
то писать НЕ
иначе писать НИ
все
кон

Сокращённая форма ветвления

если <условие>
то <действия 1>
все

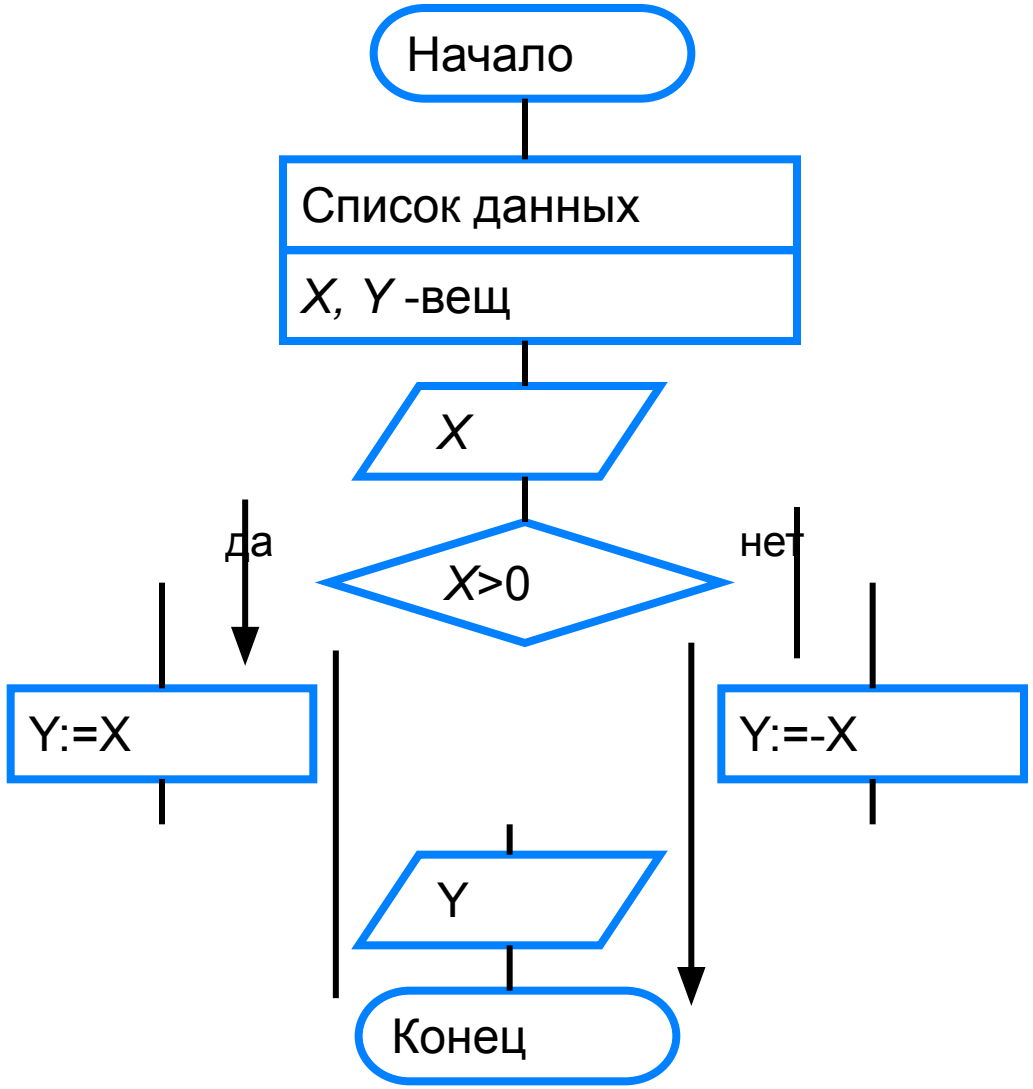
Пример:

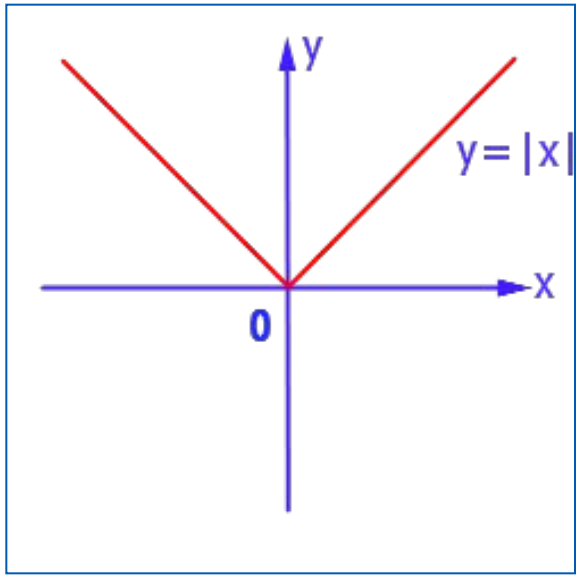
алг сборы на прогулку
нач
если на улице дождь
то взять зонтик
все
кон

Операции сравнения

A < B	A меньше B
A <= B	A меньше или равно B
A = B	A равно B
A > B	A больше B
A >= B	A больше или равно B
A <> B	A не равно B

Вычисление функции $f(x)=|x|$

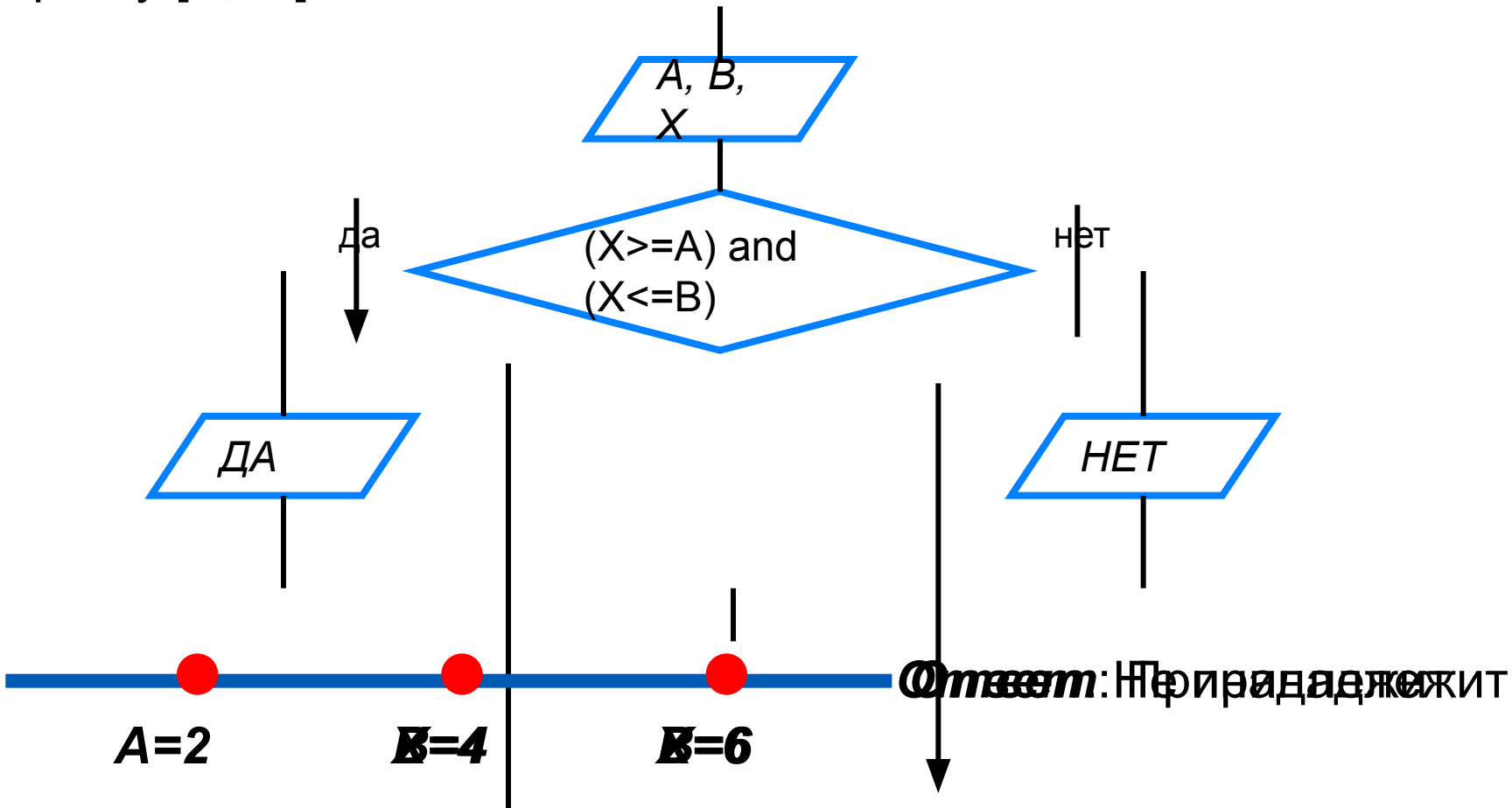




Простые и составные условия

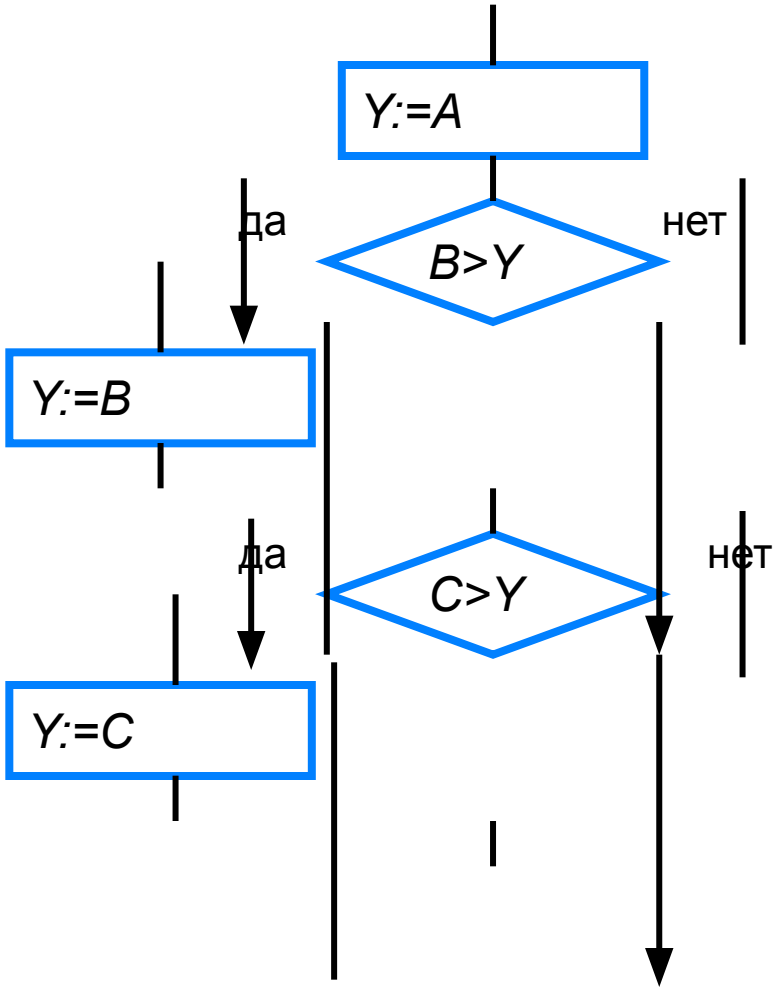
Простые условия состоят из одной операции сравнения. ***Составные*** условия получаются из простых с помощью логических связок *and* (**и**), *or* (**или**), *not* (**не**).

Пример. Алгоритм определения принадлежности точки ***X*** отрезку [***A; B***].



Наибольшая из 3-х величин

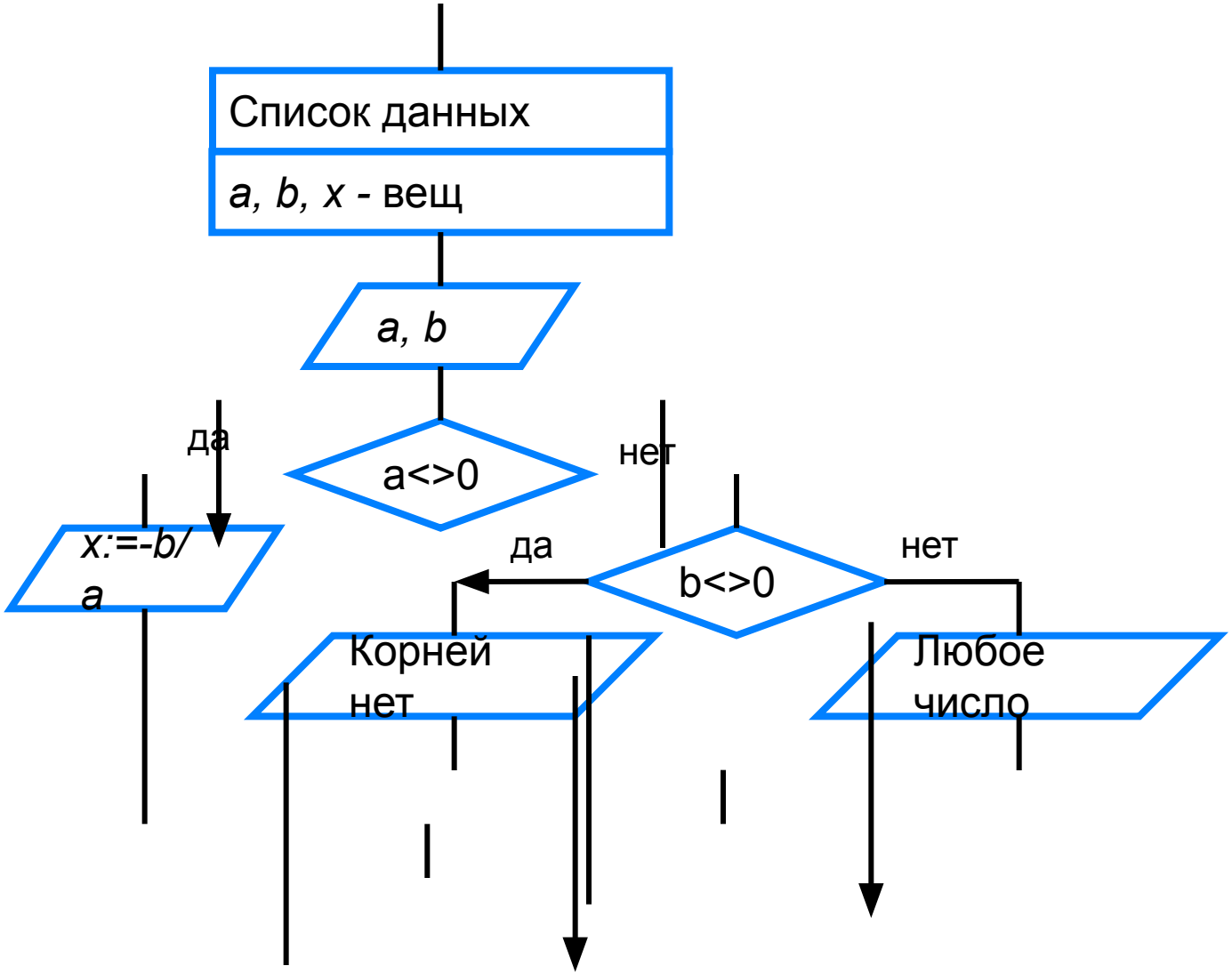
Переменной ***Y*** присваивается значение большей из трёх величин ***A***, ***B*** и ***C***.



Шаг	Константы			Переменная	Условие
	A	B	C	Y	
	10	30	20		
1				10	
2					30 > 10 (Да)
3				30	
4					20 > 30 (Нет)

Ответ: ***Y*** **= 30**

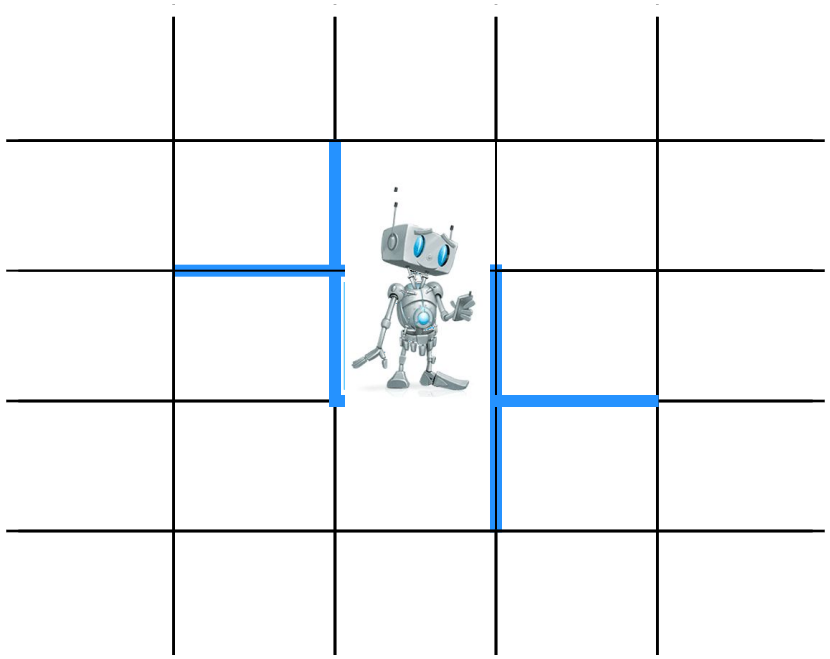
Решение линейного уравнения $ax + b = 0$



Разветвляющийся алгоритм для Робота

В какую клетку переместится Робот после выполнения следующего фрагмента алгоритма.

если справа свободно **или** снизу свободно
  то закрасить
все
если справа стена
  то влево
все
если слева стена
  то вправо
все



б а

Для записи любого алгоритма достаточно ***трёх основных алгоритмических конструкций*** (структур): следования, ветвления, повторения.

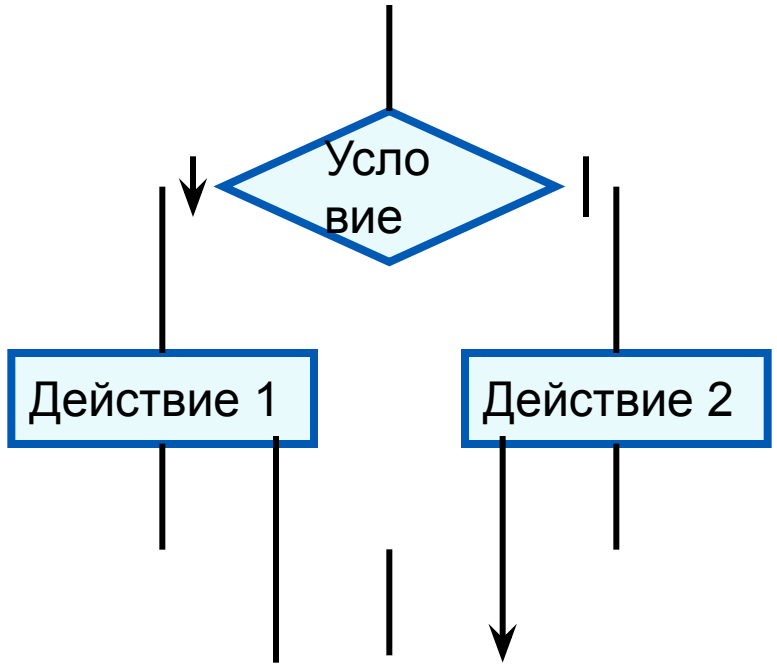
Ветвление - алгоритмическая конструкция, в которой в зависимости от результата проверки условия (да или нет) предусмотрен выбор одной из двух последовательностей действий (ветвей).

Алгоритмы, в основе которых лежит структура «ветвление», называют ***разветвляющимися***.

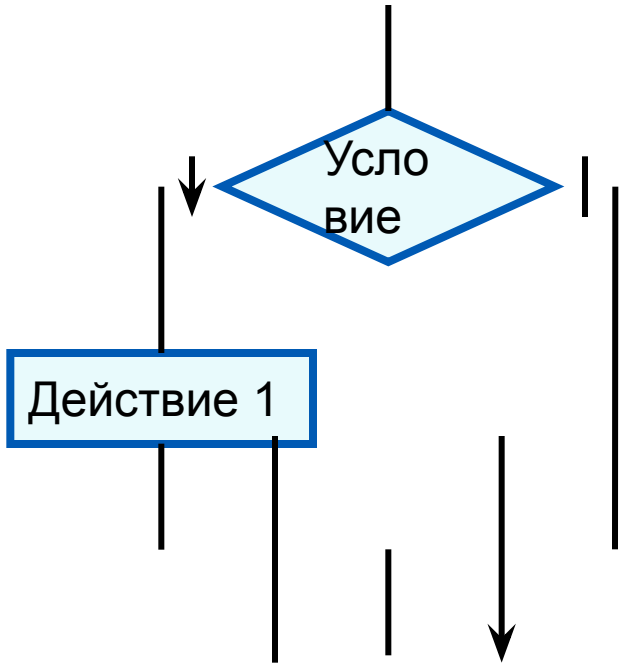
Опорный конспект

Ветвление - алгоритмическая конструкция, в которой в зависимости от результата проверки условия (да или нет) предусмотрен выбор одной из двух последовательностей действий (ветвей).

Алгоритмы, в основе которых лежит структура «ветвление», называют ***разветвляющимися***.



Полная форма ветвления



Сокращённая форма ветвления

Повторение

Повторение - последовательность действий, выполняемых многократно.

Алгоритмы, содержащие конструкцию повторения, называют ***циклическими*** или **циклами**.

Последовательность действий, многократно повторяющаяся в процессе выполнения цикла, называется ***телом цикла***.

Типы циклов

Могут быть

Заданы условия продолжения работы

Пока есть кирпич

Заданы условия окончания работы

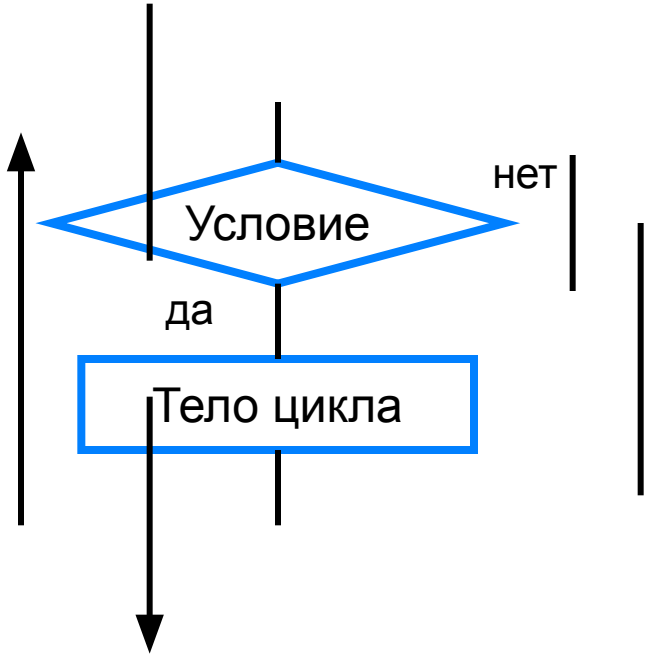
Пока не наступит ночь

Задано число повторений

Ровно 100 кирпичей

Цикл с заданным условием продолжения работы (цикл-ПОКА, цикл с предусловием)

нц пока <условие>
  <тело цикла (последовательность действий)>
кц



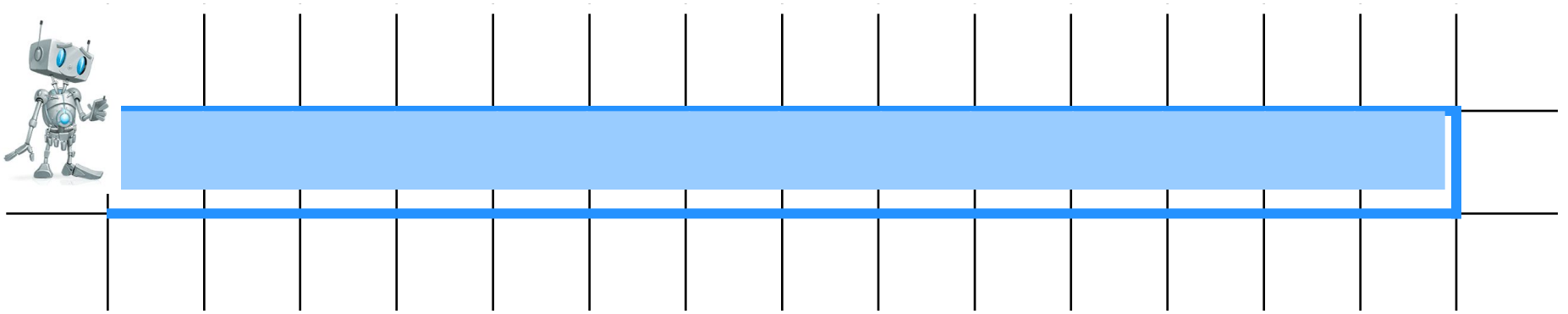
Погрузка кирпичей

алг погрузка
нач
нц пока есть кирпичи
взять один кирпич
если кирпич целый
то положить кирпич в машину
иначе отложить кирпич в сторону
все
кц
кон

Робот в коридоре

Правее Робота расположен коридор неизвестной длины. Необходимо, чтобы Робот закрасил все клетки этого коридора.

нц пока справа свободно
    вправо
    закрась
кц



Частное и остаток

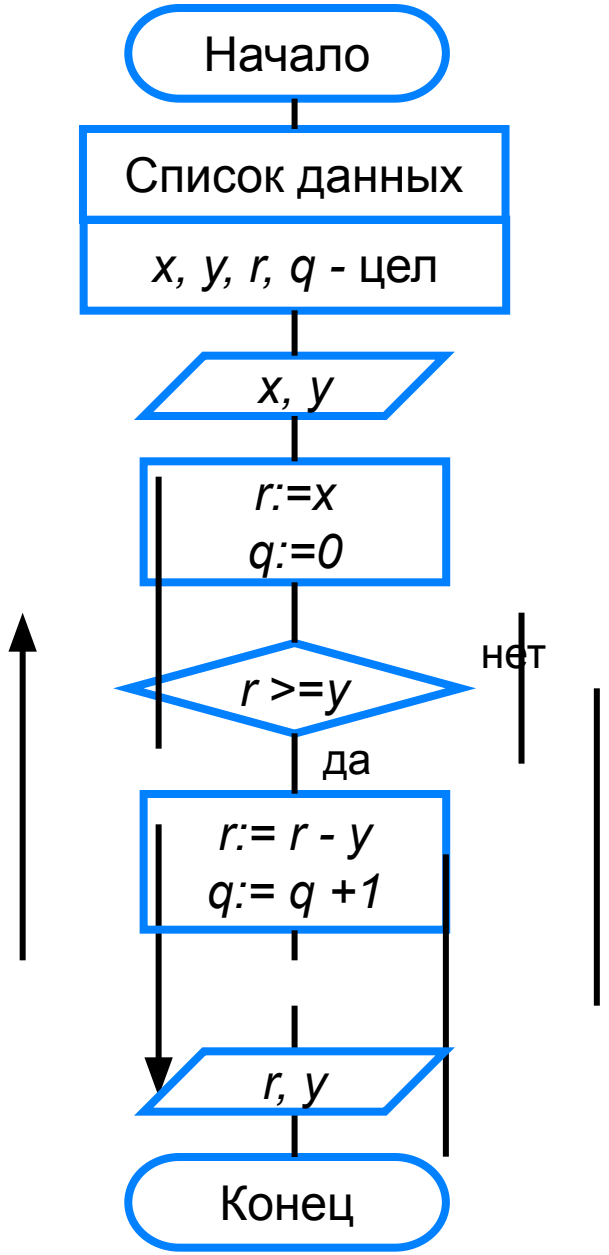
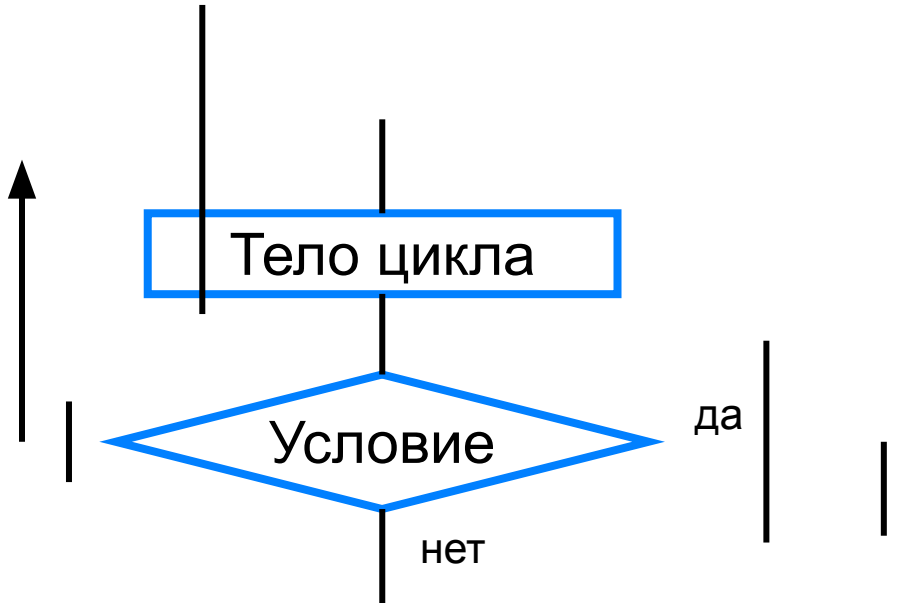


Таблица значений переменных

Шаг алгоритма	Операция	Переменная				Условие $r >= y$
		x	y	r	q	
1	Ввод x	17				
2	Ввод y	17	5			
3	$r := x$	17	5	17		
4	$q := 0$	17	5	17	0	
5	$r >= y$					17 > 5 (Да)
6	$r := r - y$	17	5	12	0	
7	$q := q + 1$	17	5	12	1	
8	$r >= y$					12 > 5 (Да)
9	$r := r - y$	17	5	7	1	
10	$q := q + 1$	17	5	7	2	
11	$r >= y$					7 > 5 (Да)
12	$r := r - y$	17	5	2	2	
13	$q := q + 1$	17	5	2	3	
17	$r >= y$					2 > 5 (Нет)
18	Вывод r			2		
19	Вывод q				3	

Цикл с заданным условием окончания работы (цикл-ДО, цикл с постусловием)



Запись на алгоритмическом языке:

нц
  <тело_цикла (последовательность действий)>
кц при <условие>

Цикл с постусловием

Пример. Алгоритм по выучиванию наизусть четверостишия.

алг четверостишие
нач
  нц
    прочитать четверостишие по книге 1 раз
    рассказать четверостишие
  кц при не сделал ошибку
кон

Вычисление переменной b

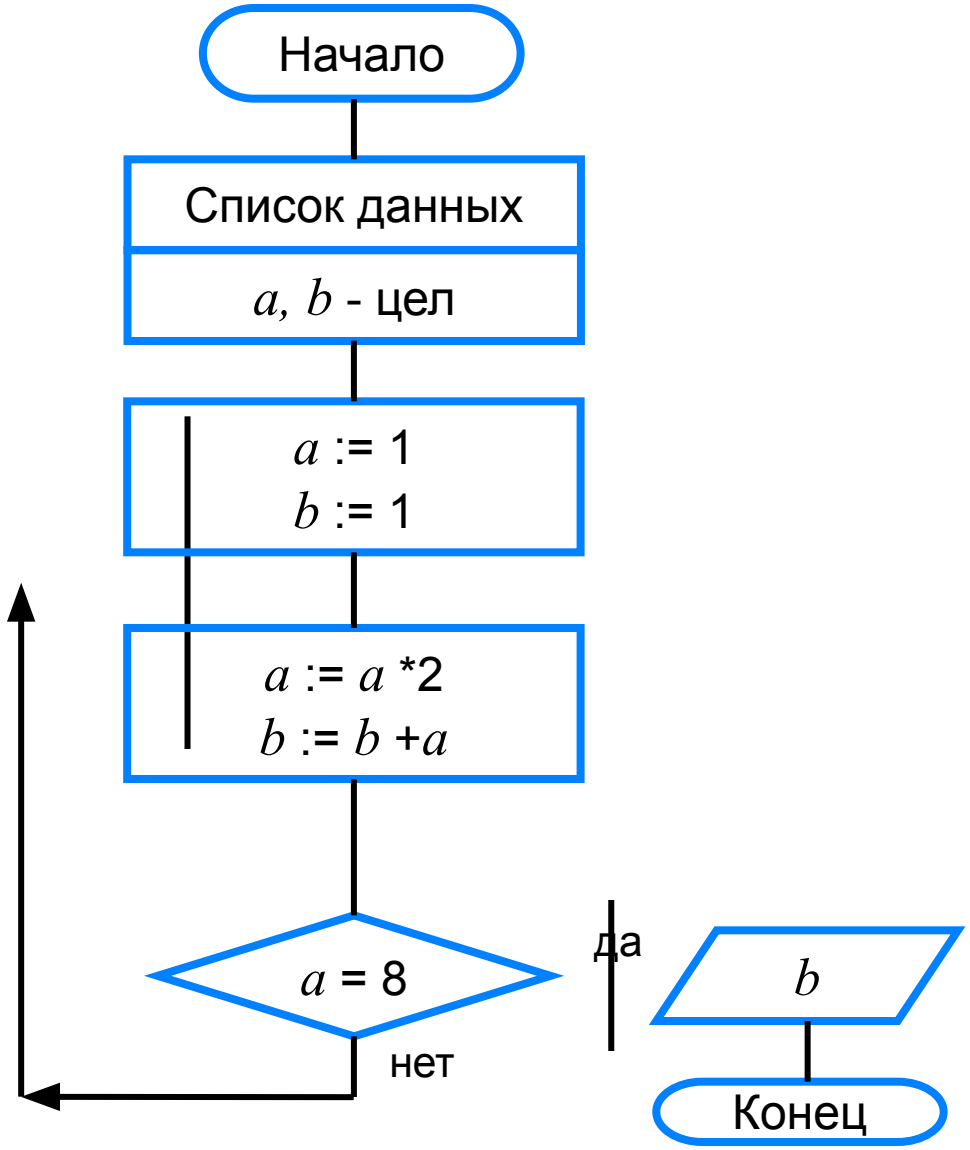


Таблица значений переменных

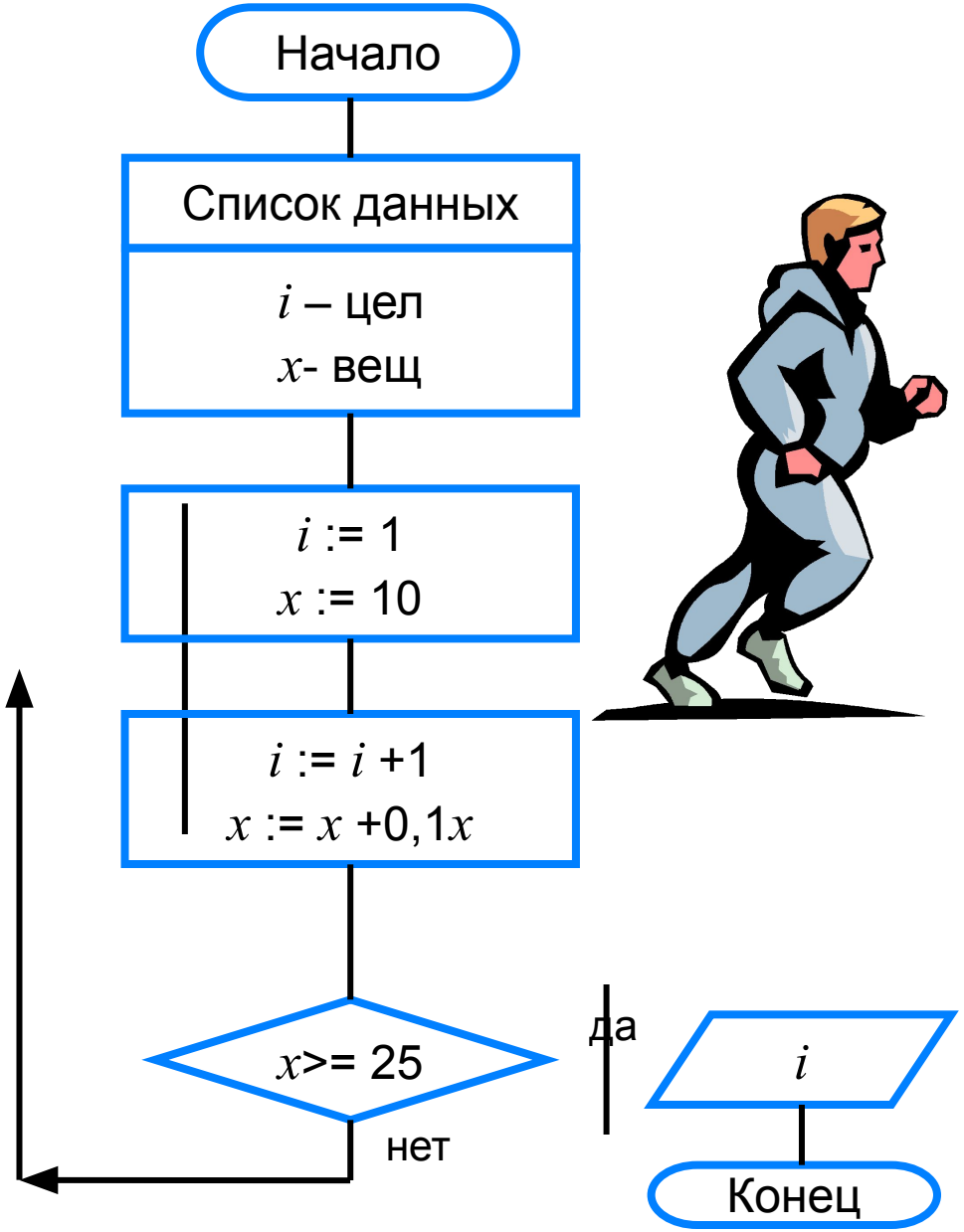
Шаг алгоритма	Операция	Переменные		Условие
		a	b	$a = 8$
1	$a := 1$	1		
2	$b := 1$	1	1	
3	$a := a * 2$	2	1	
4	$b := b+a$	2	3	
5	$a = 8$			2 = 8 (Нет)
6	$a := a * 2$	4	3	
7	$b := b+a$	4	7	
8	$a = 8$			4 = 8 (Нет)
9	$a := a * 2$	8	7	
10	$b := b+a$	8	15	
11	$a = 8$			8 = 8 (Да)

Задача о тренировках

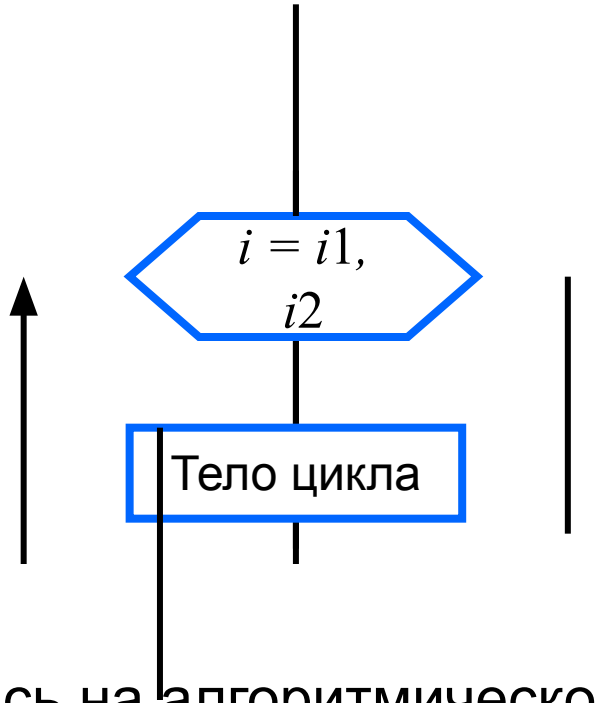
В 1-й день - пробежать 10 км; каждый следующий день увеличивать на 10% от нормы предыдущего дня. Как только достигнет или превысит 25 км, необходимо прекратить увеличение и пробегать 25 км.

Начиная с какого дня спортсмен будет пробегать 25 км?

Пусть x — количество километров, которое спортсмен пробежит в некоторый i-й день. Тогда в следующий (i + 1)-й день он пробежит x + 0,1x километров (0,1x — это 10% от x).



Цикл с заданным числом повторений (цикл-ДЛЯ, цикл с параметром)



Запись на алгоритмическом языке:

нц для i **от** i1 **до** i2
  <тело_цикла (последовательность действий)>
кц

Цикл с заданным числом повторений

алг переправа
нач
нц для i **от** 1 **до** 5
два мальчика переправляются на противоположный берег
один мальчик высаживается на берег, другой плывёт обратно
солдат переправляется через реку
мальчик возвращается на исходную позицию
кц
кон

Вычисление степени

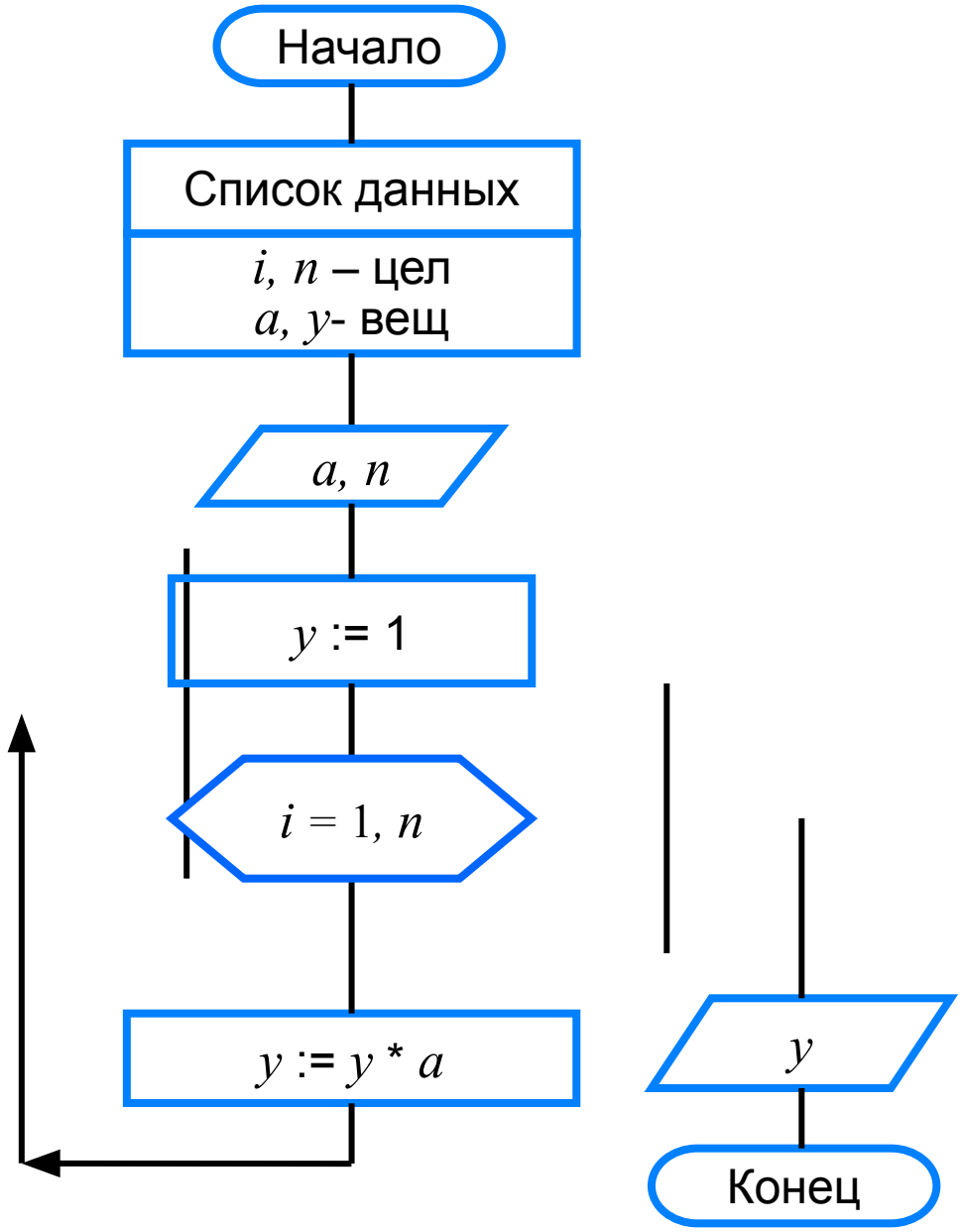


Таблица значений переменных

Шаг алгоритма	Операция	Переменная				Условие $i <= n$
		a	n	y	i	
1	Ввод a, n	4	3			
2	$y := 1$	4	3	1		
3	$i := 1$	4	3	1	1	
4	$i <= n$					1 <= 3 (Да)
5	$y := y * a$	4	3	4	1	
6	$i := i + 1$	4	3	4	2	
7	$i <= n$					2 <= 3 (Да)
8	$y := y * a$	4	3	16	2	
9	$i := i + 1$	4	3	16	3	
10	$i <= n$					3 <= 3 (Да)
11	$y := y * a$	4	3	64	3	
12	$i := i + 1$	4	3	64	4	
13	$i <= n$					4 <= 3 (Нет)

Повторение

Пример. Для исполнителя Робот цикл с известным числом повторений реализуется с помощью следующей конструкции:

нц <число повторений> **раз**
  <тело цикла>
кц

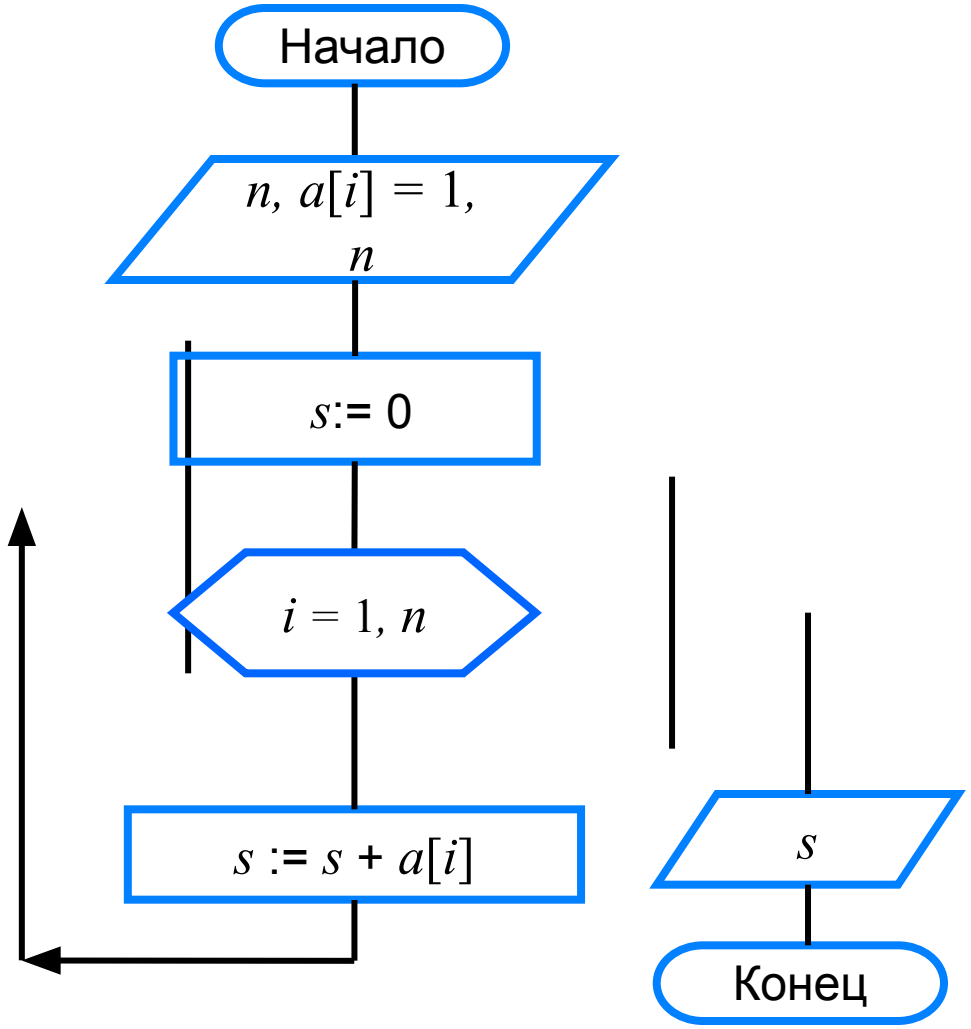
Так, если правее Робота не встретится препятствий, то, выполнив приведённый ниже алгоритм, он переместится на пять клеток вправо и закрасит эти клетки:

алг
нач
  нц 5 раз
    вправо; закрасить
  кц
кон

Повторение

Пример. В населённом пункте N домов. Известно количество людей, проживающих в каждом из домов. Составим алгоритм подсчёта жителей населённого пункта.

Исходные данные (количество жильцов) представим с помощью линейной таблицы A, содержащей N элементов:
$A[1]$ — количество жильцов дома 1, …, $A[N]$—количество жильцов дома N.
В общем случае $A[i]$ – количество жильцов дома i, где i принимает все значения от 1 до n (i =1,n).
Результат работы алгоритма обозначим через s.



Для записи любого алгоритма достаточно ***трёх основных алгоритмических конструкций*** (структур): следования, ветвления, повторения.

Повторение - алгоритмическая конструкция, представляющая собой последовательность действий, выполняемых многократно.

Алгоритмы, содержащие конструкцию «повторение», называют ***циклическими*** или ***циклами***.

Последовательность действий, многократно повторяющаяся в процессе выполнения цикла, называется ***телом цикла***.

В зависимости от способа организации повторений различают три типа циклов:

1) цикл с заданным условием продолжения работы;

2) цикл с заданным условием окончания работы;

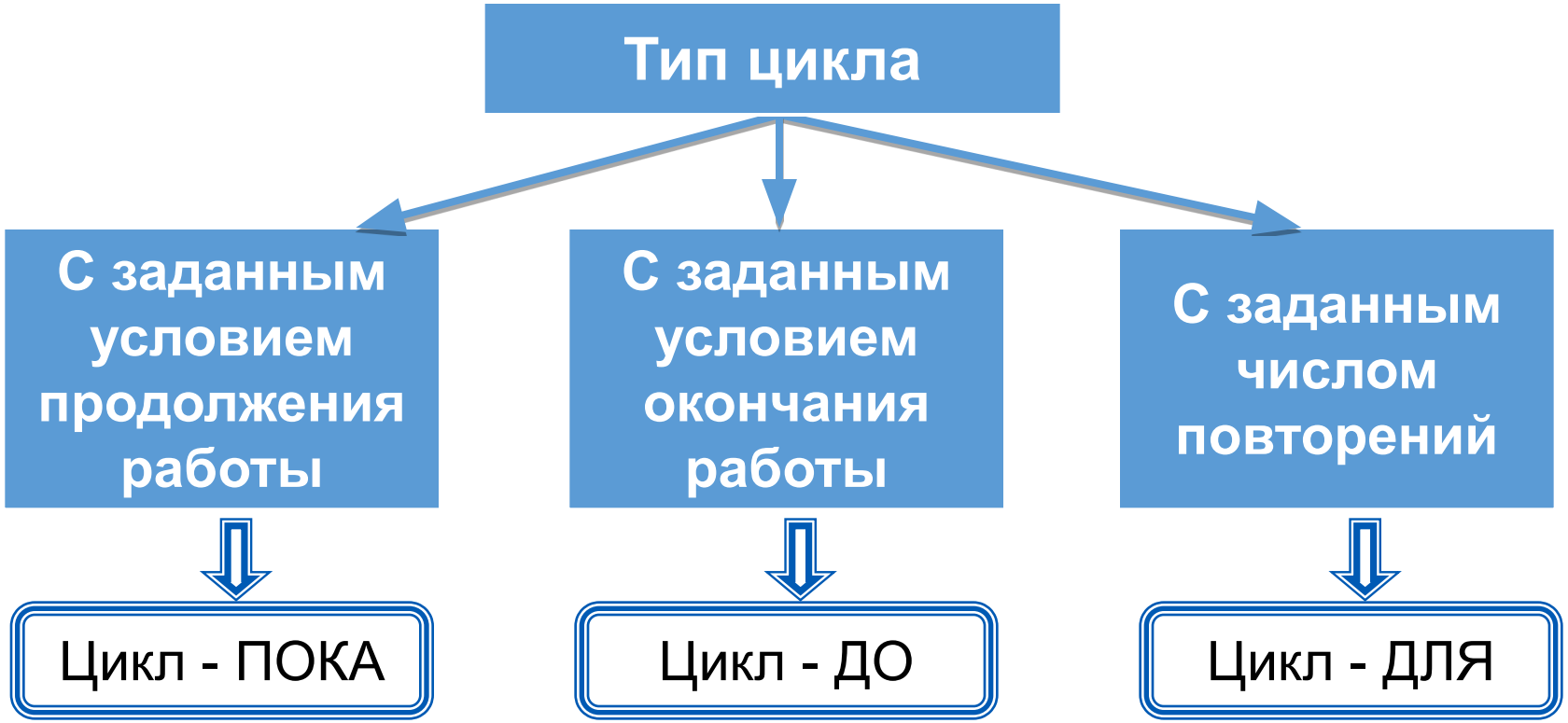
3) цикл с заданным числом повторений.

Опорный конспект

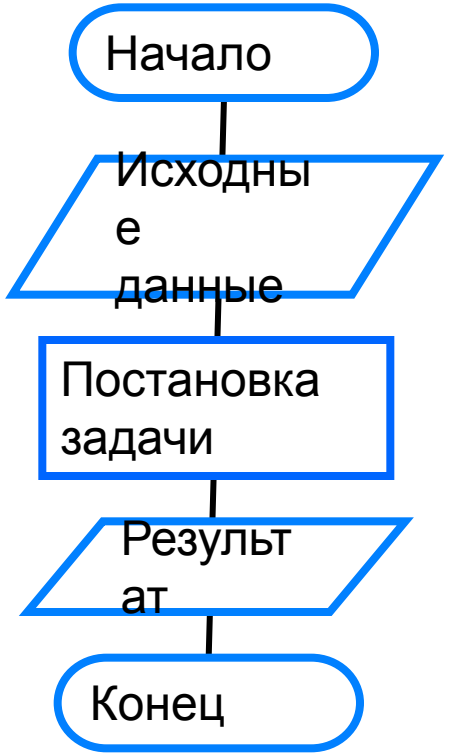
Повторение - алгоритмическая конструкция, представляющая собой последовательность действий, выполняемых многократно.

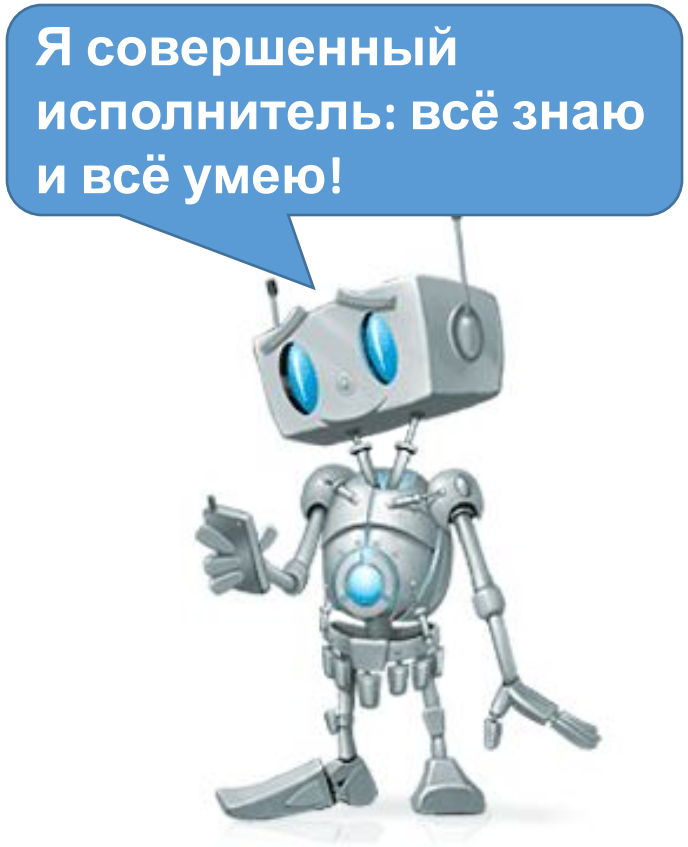
Алгоритмы, содержащие конструкцию «повторение», называют ***циклическими*** или ***циклами***.

Последовательность действий, многократно повторяющаяся в процессе выполнения цикла, называется ***телом цикла***.

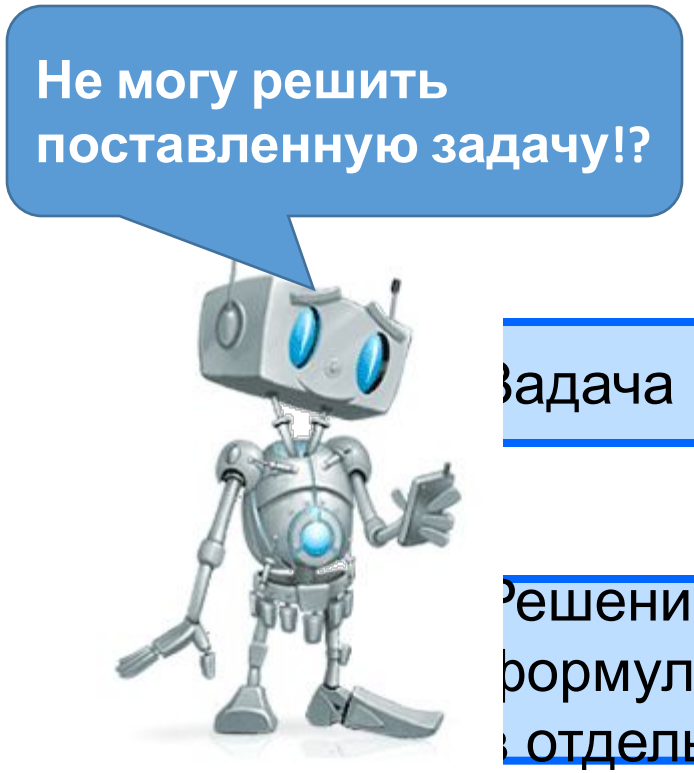


Последовательное построение алгоритма





Последовательное построение алгоритма



Упрощение команд постановки задачи

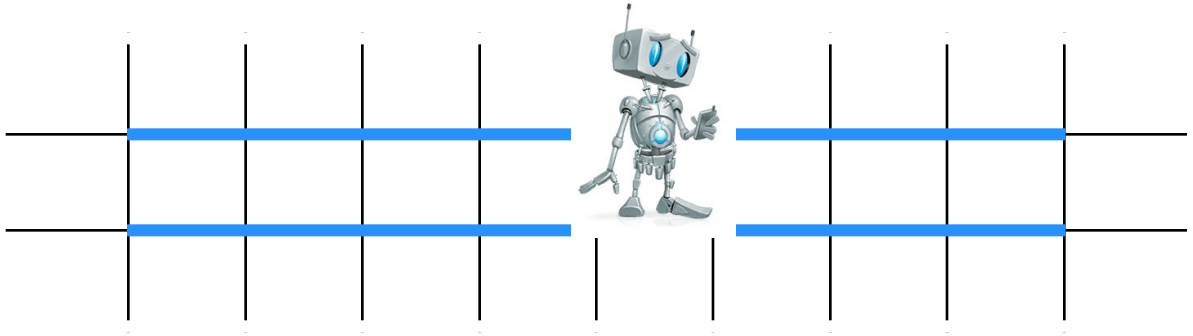
Задача разбивается на более простые части

Решение каждой части задачи формулируется в отдельной команде (предписании)

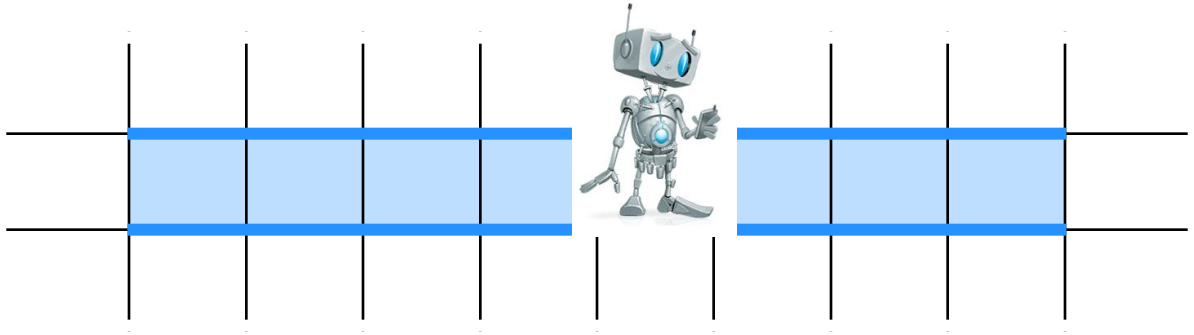
Предписания, выходящие за пределы возможностей исполнителя, представляют в виде более простых команд

Разработка алгоритма методом последовательного уточнения для исполнителя Робот

Робот находится в некоторой клетке горизонтального коридора. Ни одна из клеток коридора не закрашена.



Робот должен закрасить все клетки этого коридора и вернуться в исходное положение.



Укрупнённый план действий Робота

Начало

1. Закраска всех клеток коридора левее исходной

2. Возвращение в исходное положение

3. Закраска всех клеток коридора правее исходной

4. Возвращение в исходное положение

5. Закраска исходной клетки

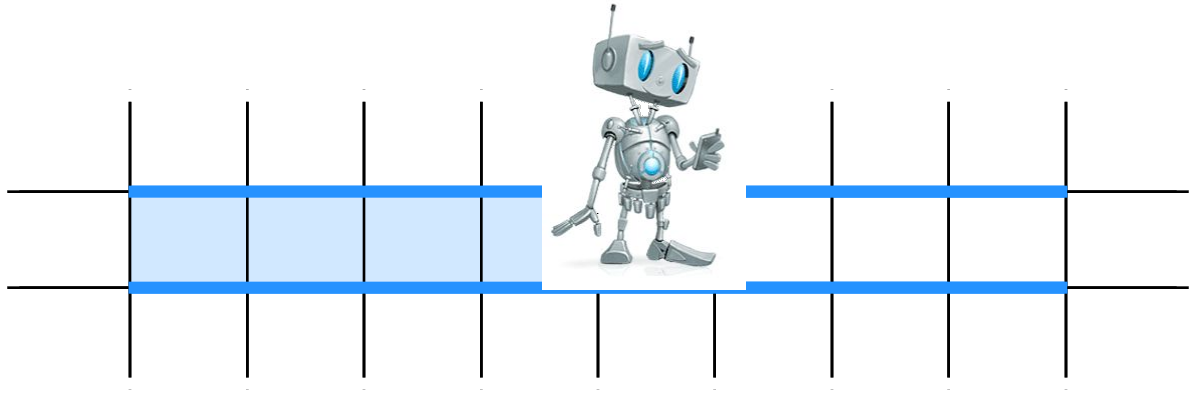
Конец

Детализация плана действий Робота

1. Закраска всех клеток коридора, находящихся левее Робота:

влево
нц пока сверху стена **и** снизу стена
  закрасить; влево
кц

Положение Робота после выполнения этого алгоритма:

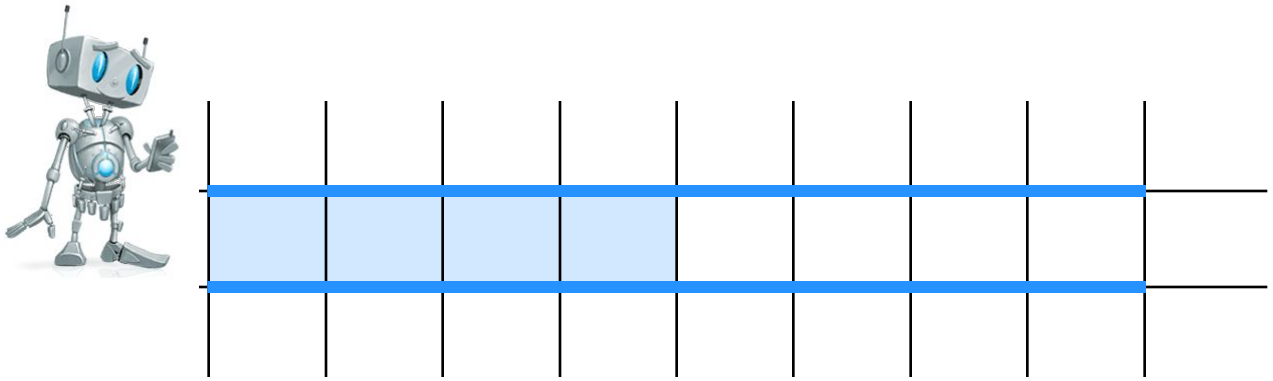


Детализация плана действий Робота

2. Возвращение Робота в коридор в исходную точку:

```
вправо
нц пока клетка закрашена
  вправо
кц
```

Положение Робота после выполнения этого алгоритма:

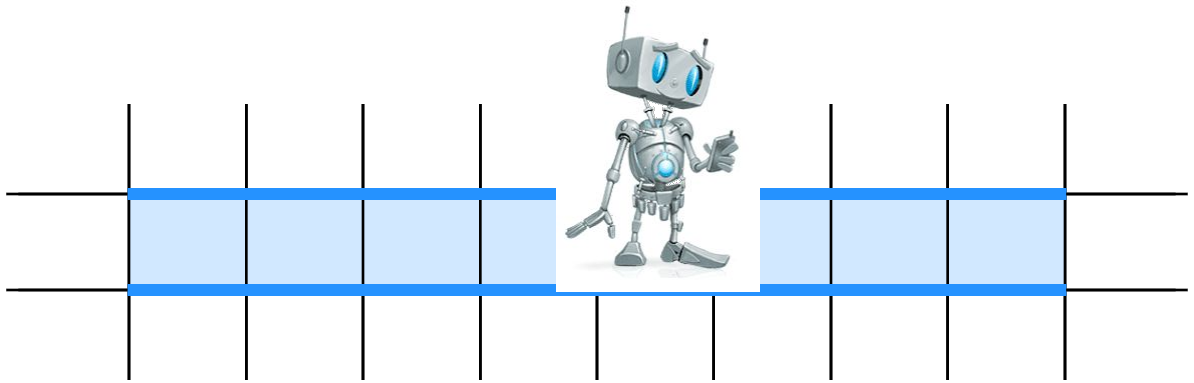


Детализация плана действий Робота

3. Закраска всех клеток коридора, находящихся правее Робота:

вправо
нц пока сверху стена **и** снизу стена
  закрасить; вправо
кц

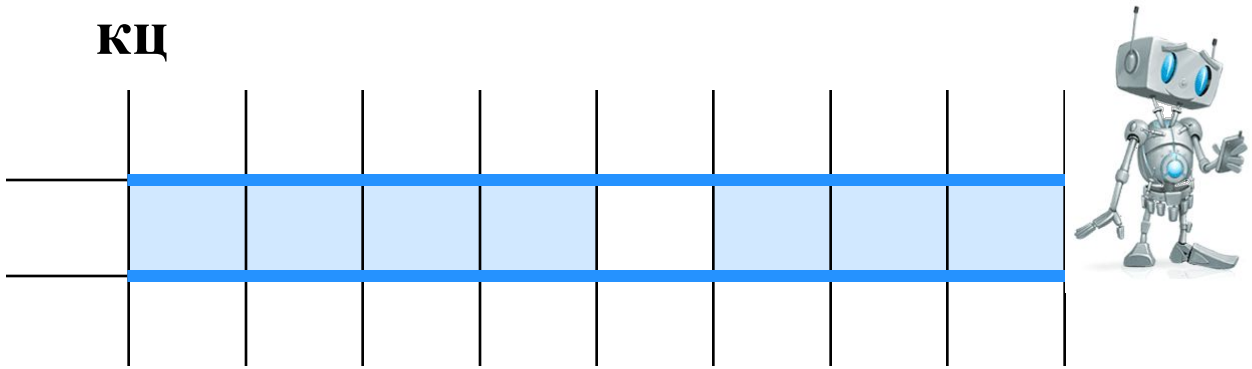
Положение Робота после выполнения этого алгоритма:



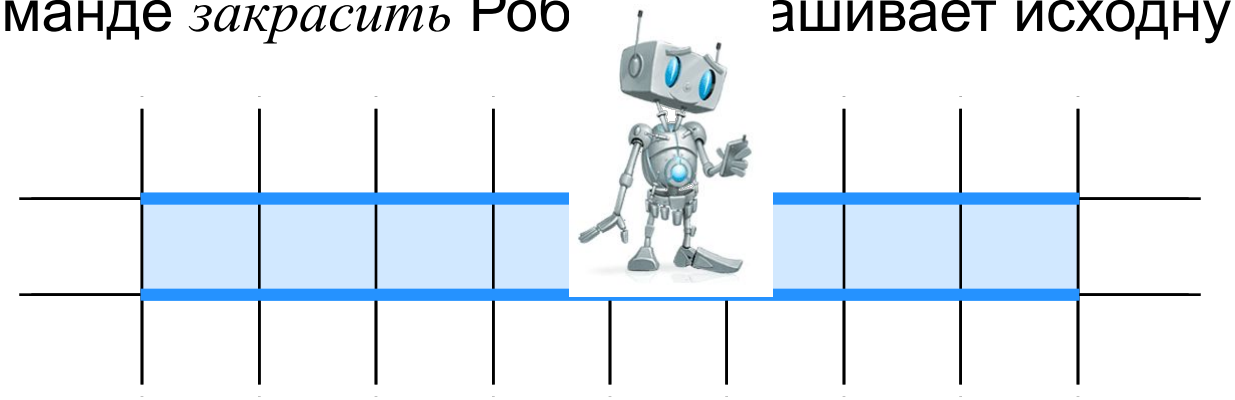
Детализация плана действий Робота

4.Возвращение Робота в коридор в исходную точку:

влево
нц пока клетка закрашена
  влево
кц



5. По команде *закрасить* Роб ашивает исходную точку.

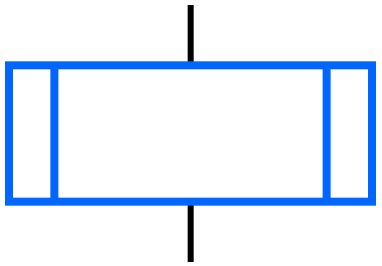


Программа для Робота

```
алг
нач
  влево
  нц пока сверху стена и снизу стена
    закрасить; влево
  кц
    вправо
  нц пока клетка закрашена
    вправо
  кц
    вправо
  нц пока сверху стена и снизу стена
    закрасить; вправо
  кц
    влево
  нц пока клетка закрашена
    влево
  кц
    закрасить
кон
```

Вспомогательный алгоритм

Вспомогательный алгоритм - алгоритм, целиком используемый в составе другого алгоритма.



Блок «предопределённый процесс»

Вспомогательный алгоритм делает структуру алгоритма более простой и понятной.

Алгоритм вычисления степени

$y = a^x$, где x - целое число, $a \neq 0$.

По определению степени с целым показателем:

$$a^0 = 1, a \neq 0;$$

$$a^{-n} = 1/a^n, a \neq 0, n \notin N$$

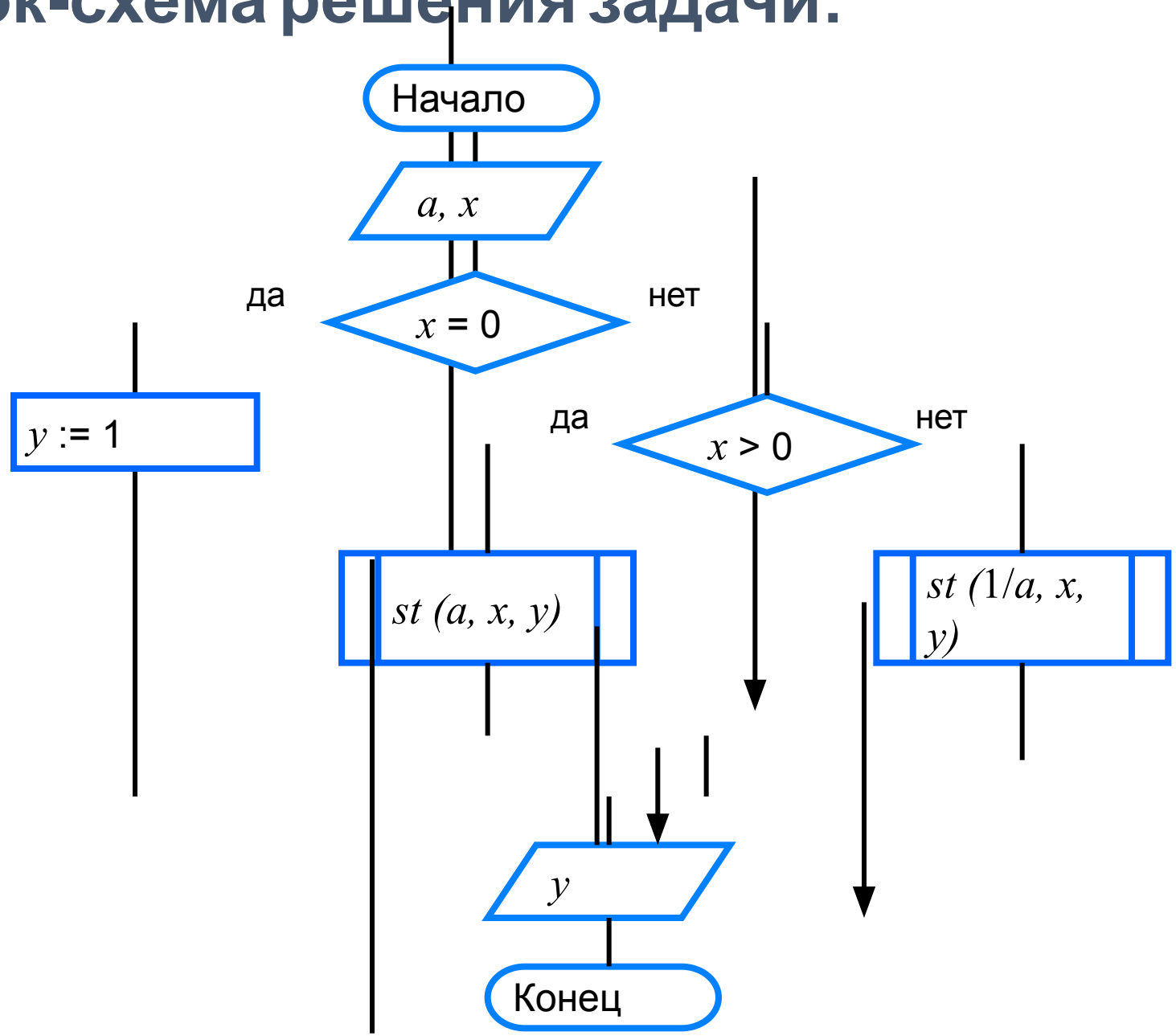
$$\frac{1}{a^{-x}} = \left(\frac{1}{a}\right)^{-x}$$

$$y = \begin{cases} 1 & \text{при } x = 0 \\ a^x & \text{при } x > 0, \\ \left(\frac{1}{a}\right)^{-x} & \text{при } x < 0. \end{cases}$$

Обозначим алгоритм возведения числа в степень ***st(a, n, y).***

Это вспомогательный алгоритм.

Блок-схема решения задачи:



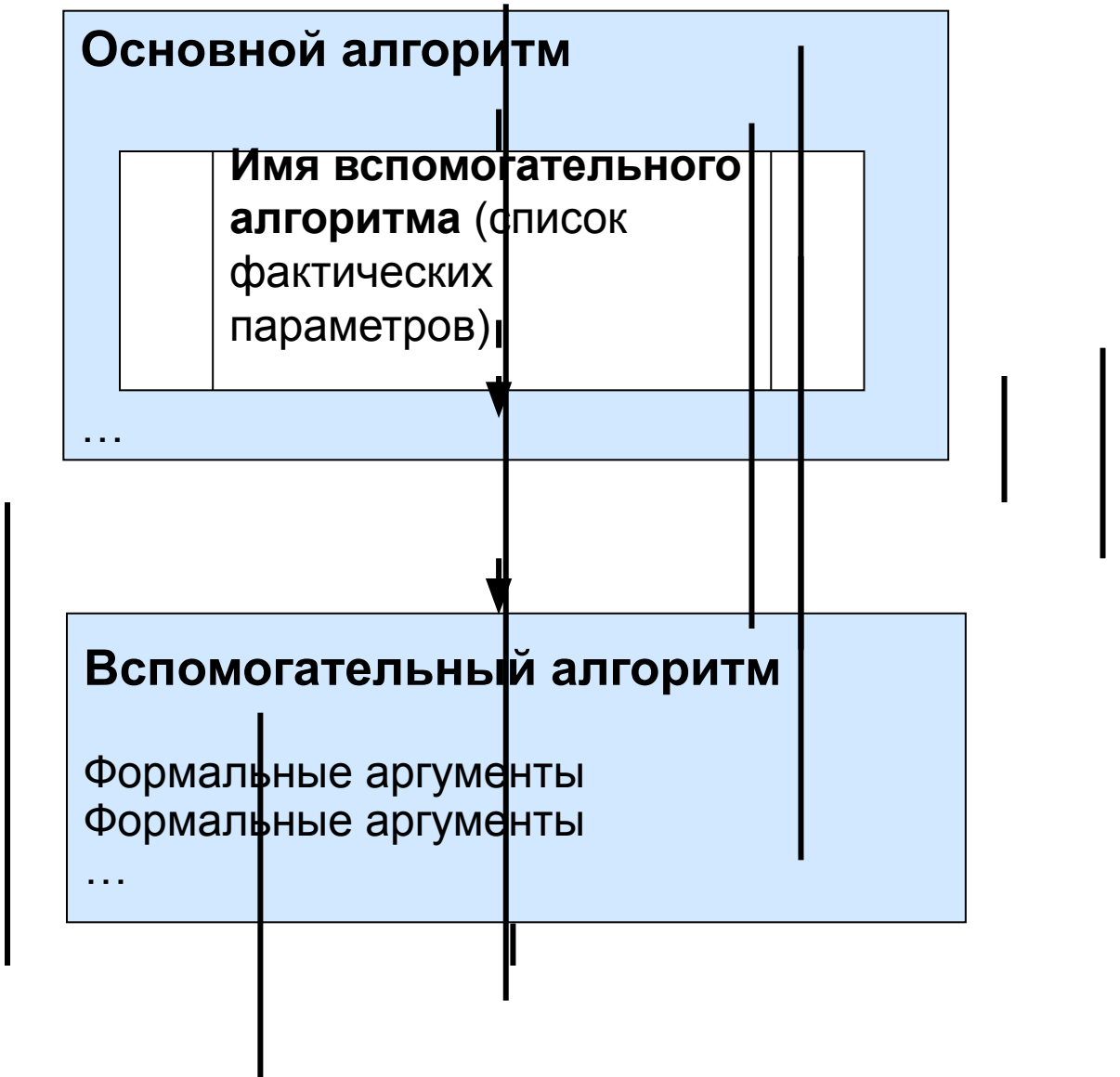
Формальные и фактические параметры

Формальные параметры используются при описании алгоритма.

Фактические параметры - те величины, для которых будет исполнен вспомогательный алгоритм.

Типы, количество и порядок следования формальных и фактических параметров должны совпадать.

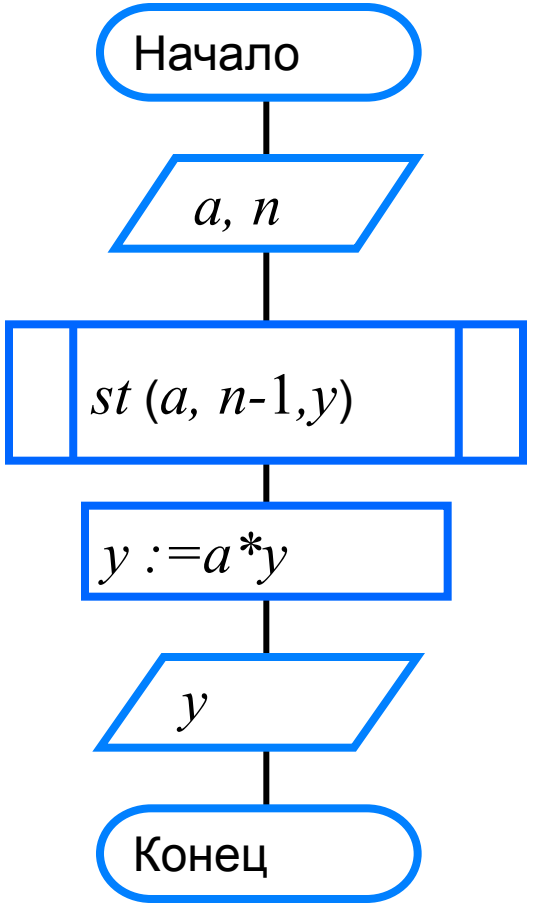
Схема вызова вспомогательного алгоритма



Рекурсивный алгоритм

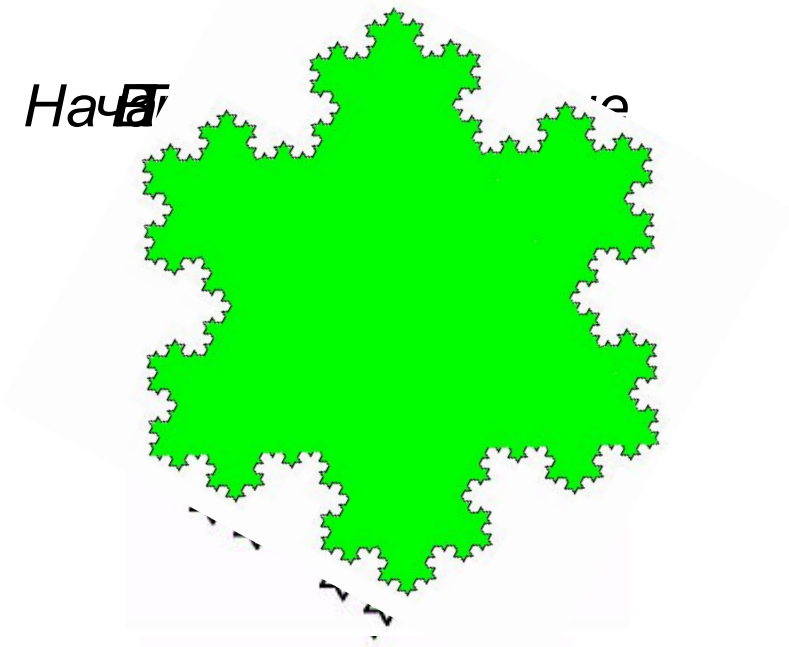
Алгоритм, в котором прямо или косвенно содержится ссылка на него же как на вспомогательный алгоритм, называют ***рекурсивным***.

Пример. Алгоритм вычисления степени с натуральным показателем n для любого вещественного числа a, представленный в виде рекурсивного алгоритма



Снежинка Коха

Пример. Рассмотрим алгоритм построения геометрической фигуры, которая называется снежинкой Коха. Шаг процедуры построения состоит в замене средней трети каждого из имеющихся отрезков двумя новыми той же длины.



С каждым шагом фигура становится всё причудливее. Граница снежинки Коха - положение кривой после выполнения бесконечного числа шагов.

Метод последовательного построения алгоритма:

- исходная задача разбивается на несколько частей, каждая из которых проще всей задачи, и решение каждой части формулируется в отдельной команде;
- если получаются команды, выходящие за пределы возможностей исполнителя, то они представляются в виде совокупности ещё более простых предписаний;
- процесс продолжается до тех пор, пока все предписания не будут понятны исполнителю.

Вспомогательный алгоритм - алгоритм, целиком используемый в составе другого алгоритма.

Алгоритм, в котором прямо или косвенно содержится ссылка на него же как на вспомогательный алгоритм, называют

.

Опорный конспект

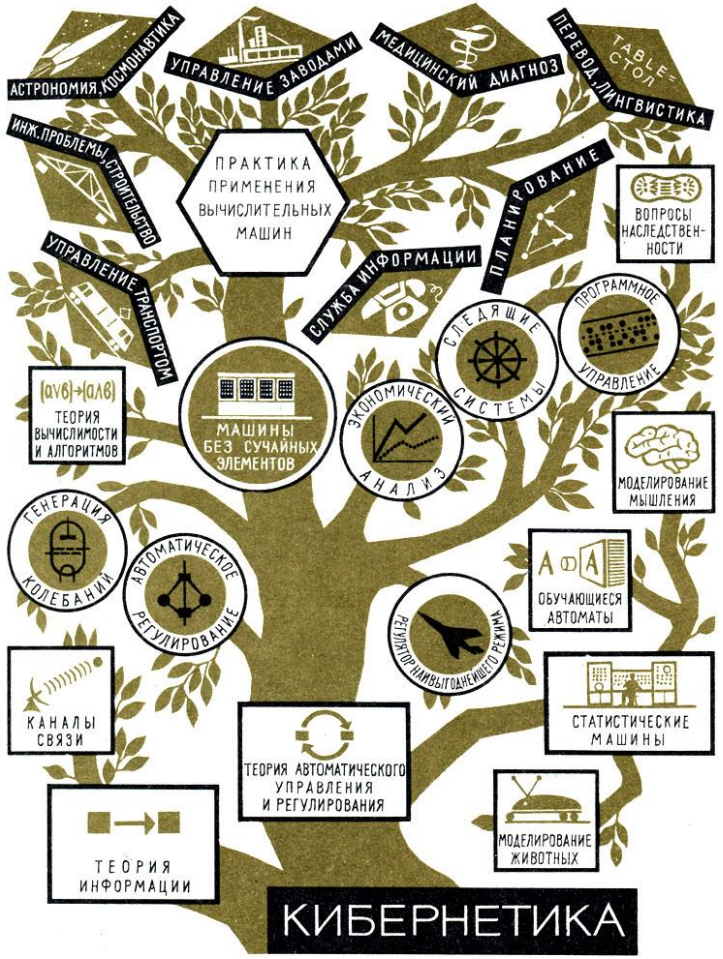
Метод последовательного построения алгоритма - один из основных методов конструирования алгоритмов.

Вспомогательный алгоритм - алгоритм, целиком используемый в составе другого алгоритма.

Управление

Управление - это процесс целенаправленного воздействия на объект; осуществляется для организации функционирования объекта по заданной программе.

Норберт Винер (1894—1964), основоположник **кибернетики** - науки об управлении.



Управляемый объект – техническое устройство

Последовательность команд по управлению объектом, приводящая к заранее поставленной цели, называется **алгоритмом управления**.

Информация и управление

Информация для управляющего объекта

- **Цель действий**
- **Способы достижения цели**

Пример. Управление движением автомашин (объект управления) на перекрёстке с помощью светофора (управляющий объект).

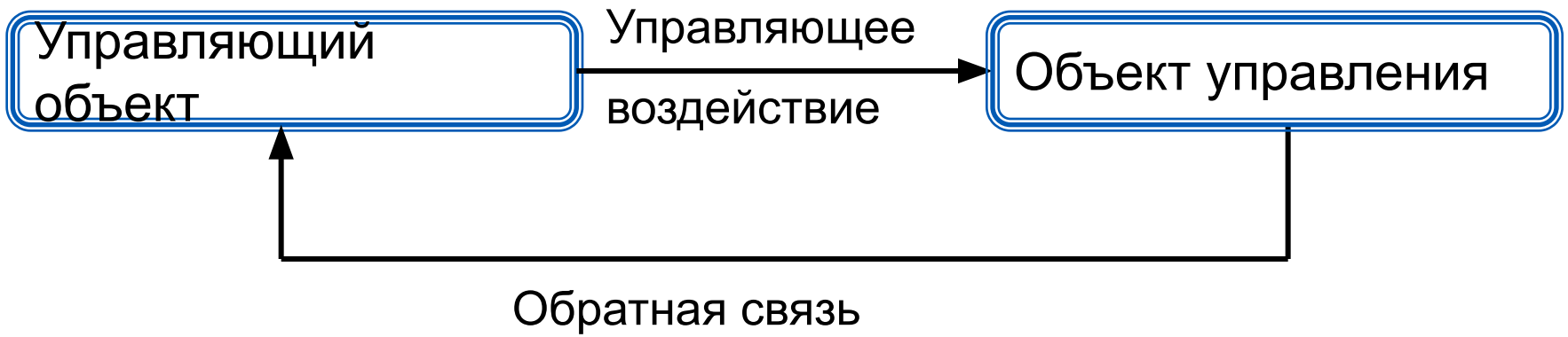
Управляющее воздействие зависит от заложенной в управляющем объекте исходной информации.

Обратная связь

Обратная связь - это процесс передачи информации о состоянии объекта управления в управляющую систему.

Обратная связь позволяет корректировать управляющие воздействия управляющей системы на объект управления в зависимости от состояния объекта управления.

Кибернетическая модель управления

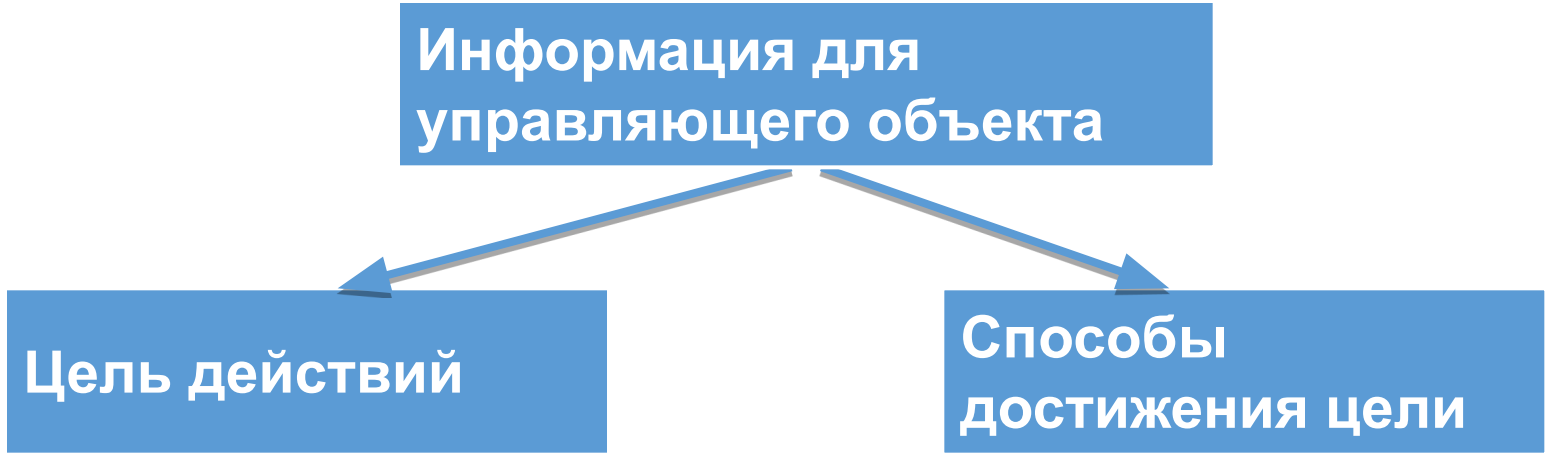


Управление - процесс целенаправленного воздействия на объект; осуществляется для организации функционирования объекта по заданной программе.

Последовательность команд по управлению объектом, приводящая к заранее поставленной цели, называется ***алгоритмом управления***.

Опорный конспект

Управление - это процесс целенаправленного воздействия на объект; осуществляется для организации функционирования объекта по заданной программе.



Последовательность команд по управлению объектом, приводящая к заранее поставленной цели, называется **алгоритмом управления.**

Обратная связь - это процесс передачи информации о состоянии объекта управления в управляющую систему.