

Урок 4

Графика в Java

(создание анимации)

Основы алгоритмизации

Алгоритмы



Что такое алгоритм?

Какие способы записи алгоритмов?

Какие существуют алгоритмы?

Кто выполняет алгоритмы?

Примеры алгоритмов.

Алгоритм открывания двери.

Достать ключ.

Вставить ключ в замочную скважину.

Повернуть ключ дважды против часовой стрелки.

Вынуть ключ.

Алгоритм "Как ехать в гости".

Выйти из дома.

Повернуть направо.

Пройти два квартала до автобусной остановки.

Сесть в автобус № 25, идущий к центру города.

Проехать три остановки.

Выйти из автобуса.

1. Налить в чайник воду.
2. Зажечь спичку.
3. Открыть кран газовой горелки.
4. Поднести спичку к горелке.
5. Поставить чайник на плиту.
6. Ждать, пока вода закипит.
7. Выключить газ.

Что может означать это перечисление?

СЛОВО

алгоритм

произошло от **algorithm**
– латинского написания
имени аль – Хорезми,
величайшего ученого из
города Хорезма,
Мухамеда бен Мусу,
жившего в 783 – 850 гг.



Алгоритм — набор инструкций,
описывающих порядок действий
исполнителя для достижения результата
решения задачи за конечное число
действий.

Кто может являться исполнителем?

СВОЙСТВА алгоритма

Дискретность

Результативность

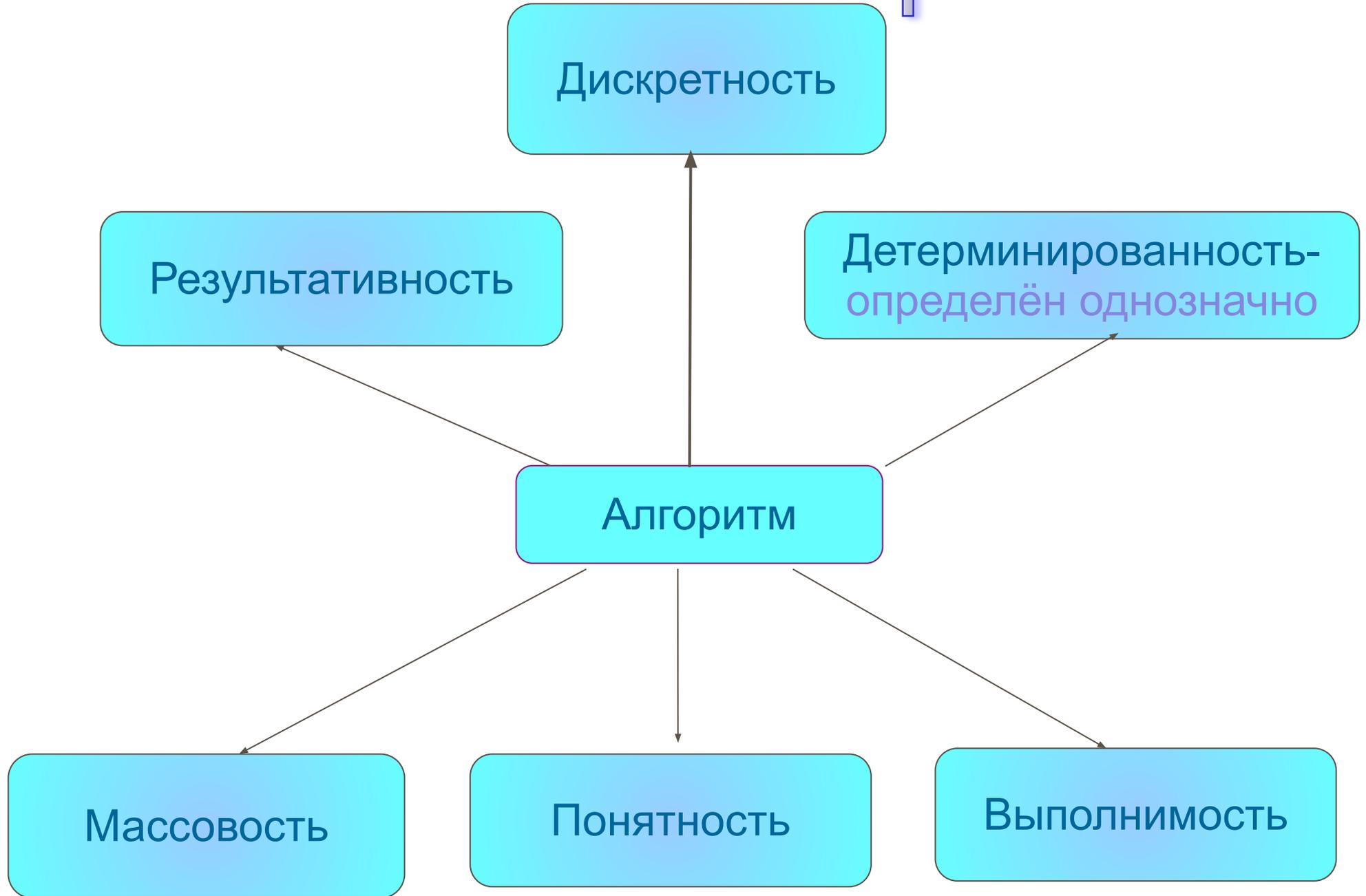
Детерминированность-
определён однозначно

Алгоритм

Массовость

Понятность

Выполнимость



**Алгоритм можно описать
следующими способами:**

□ Словесный

□ Программный

***□ Графический – например,
блок-схема (псевдокоды).***

Блок-схема



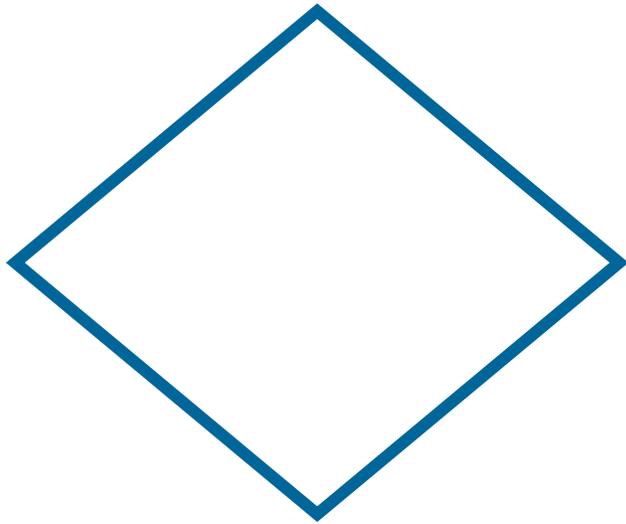
начало



конец



Выполнение действия

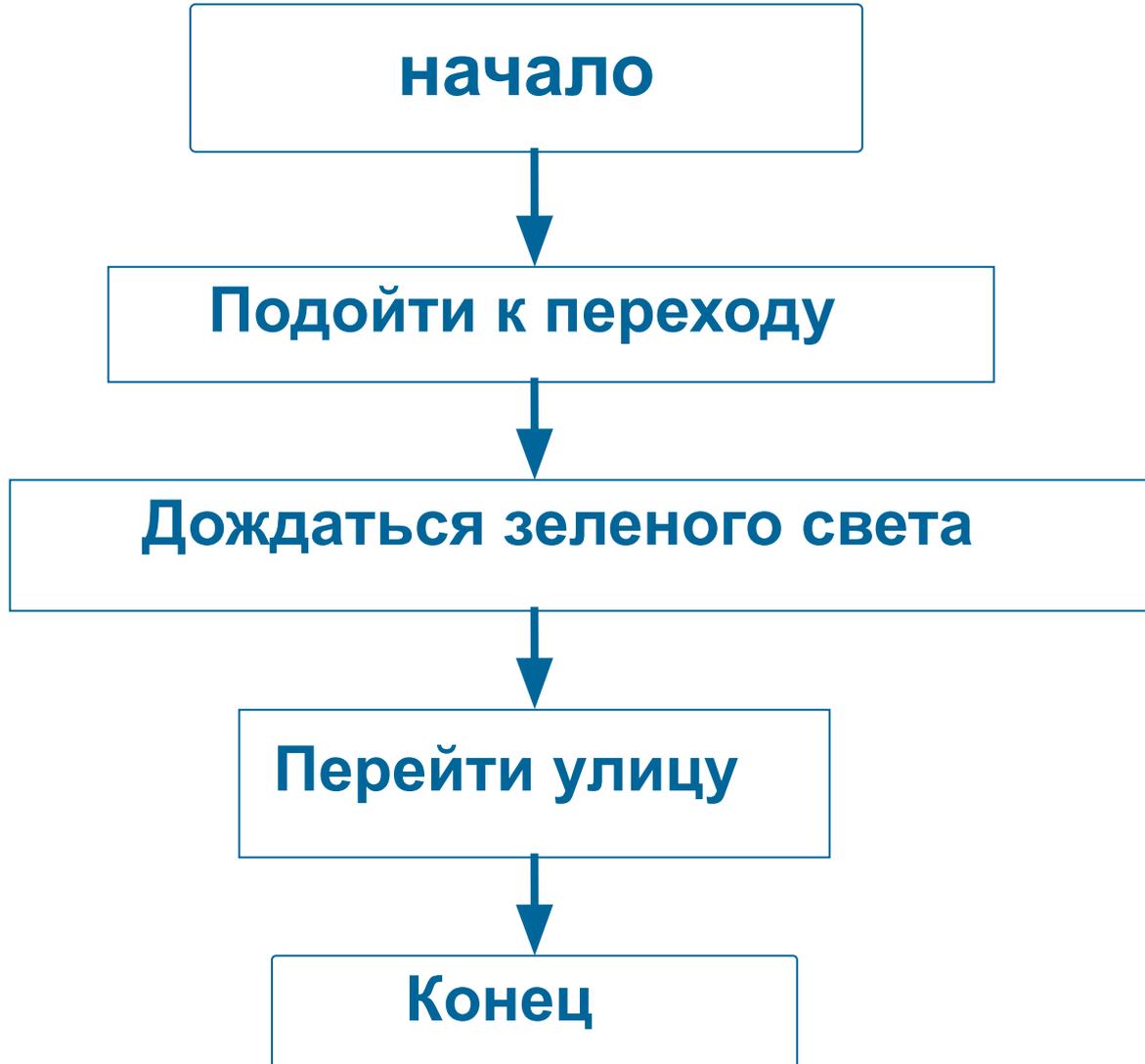


Проверка условия



Ввод/вывод данных

Алгоритм действий при переходе улицы



Кто исполнитель?
Как называется
способ описания
алгоритма?

Алгоритм посадки дерева

- 1) Выкопать в земле ямку;
- 2) Опустить в ямку саженец;
- 3) Засыпать ямку с саженцем землей;
- 4) Полить саженец водой.



Кто исполнитель?

Как называется способ описания алгоритма?

начало

Выкопать в земле ямку

Опустить в ямку саженец

Засыпать ямку с саженцем землей

Полить саженец водой

Конец

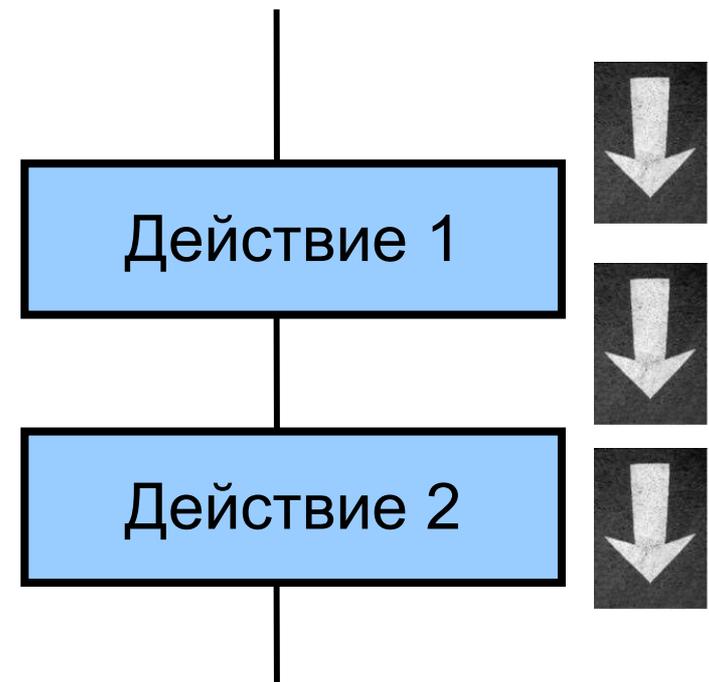


Виды алгоритмов

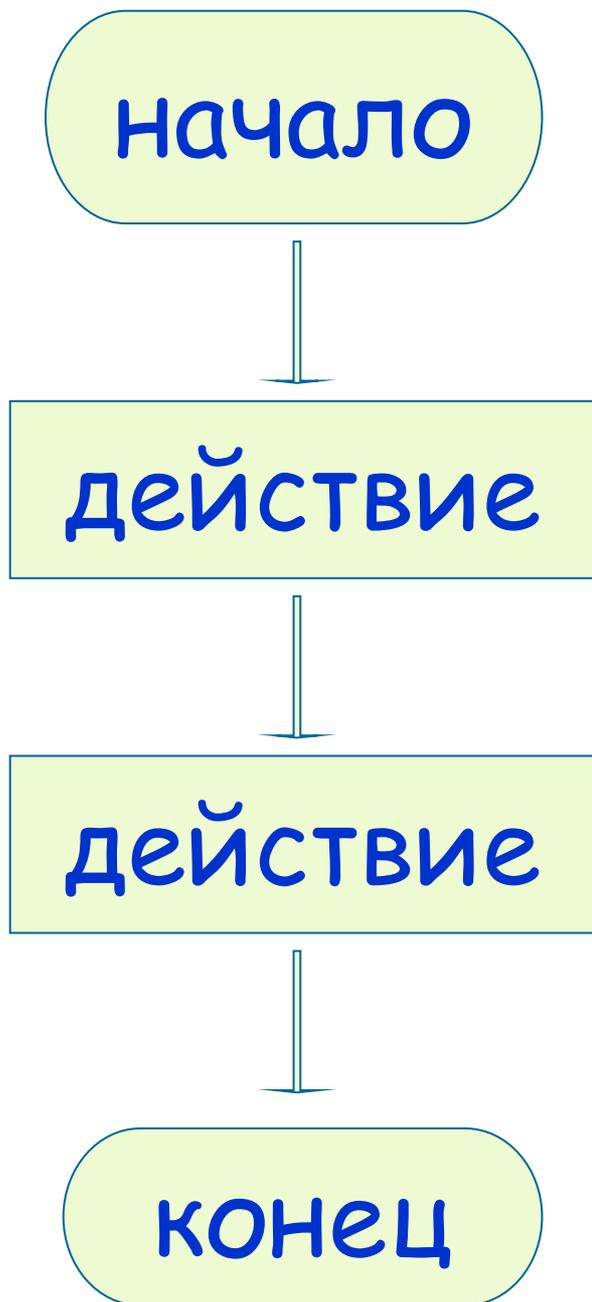
Следование - алгоритмическая конструкция, отображающая естественный, последовательный порядок действий. Алгоритмы, в которых используется только структура «следование», называются **линейными алгоритмами**.



Скажите, примеры алгоритмов которые мы рассматривали, можно отнести к линейным? Почему?



Линейный



Задание №1.

Составьте алгоритм утреннего подъема с постели.

начало



Пойди на кухню



Открой холодильник



Возьми яблоко

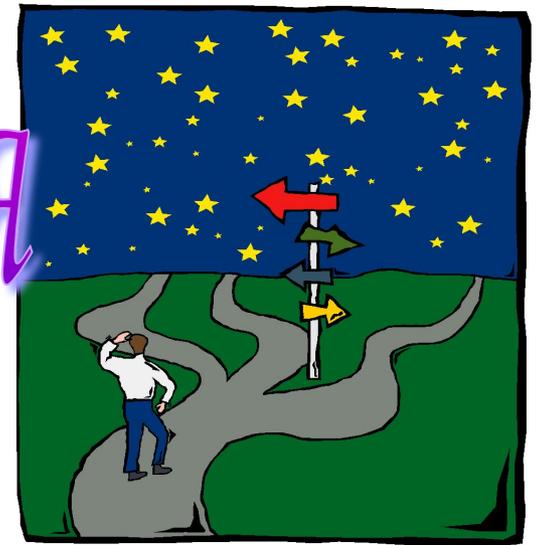


Закрой холодильник



конец

Алгоритм ветвления



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**.

ЕСЛИ <условие> ТО <действие 1>

ИНАЧЕ <действие 2>

ЕСЛИ хочешь быть здоров, ТО закаляйся

ИНАЧЕ можешь часто болеть

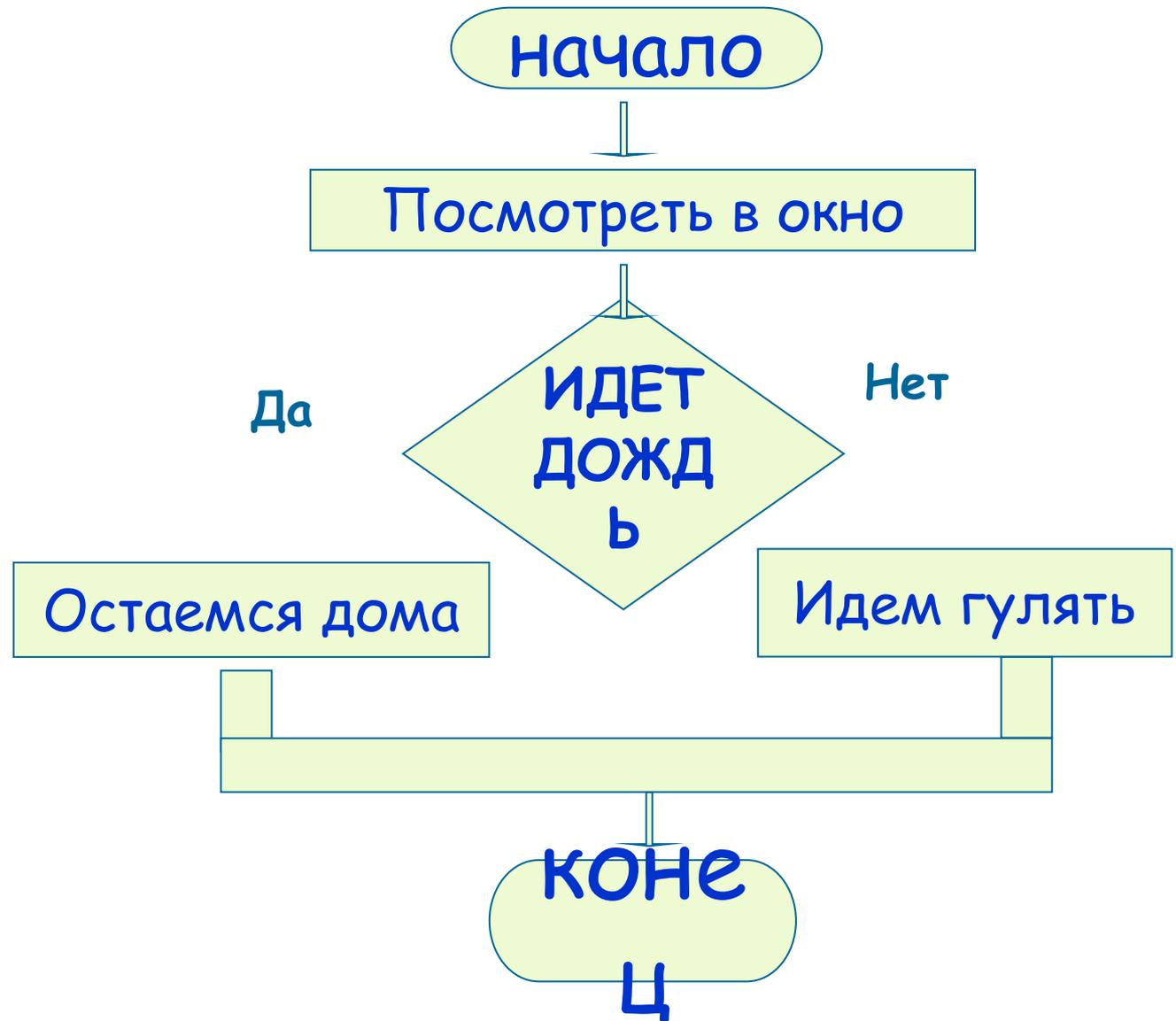
ЕСЛИ низко ласточки летают, ТО будет дождь

ИНАЧЕ дождя не будет

ЕСЛИ уроки выучены ТО иди гулять

ИНАЧЕ учи уроки

Например, алгоритм проведения выходного дня в зависимости от погоды. Если будет дождь – одни действия, если – нет, то планы будут другие.



Условие - ромб

Если **ИДЕТ ДОЖДЬ**, то **ОСТАЕМСЯ ДОМА**
иначе **ИДЕМ ГУЛЯТЬ**

Условный оператор if

Оператор **if** обеспечивает **выполнение** или **пропуск** инструкции в зависимости от указанного логического условия. Если условие истинно, то инструкция выполняется.

```
if (условие) {  
    Инструкция;  
}
```

```
// Пример 1
```

```
int a = 25;
```

```
if (a != 0) System.out.println( 100/a );
```

```
// Пример 2
```

```
int b = 25;
```

```
if (b != 0) {
```

```
    System.out.println( 100/b );
```

```
}
```

У оператора **if** существует формат с дополнительной частью **else**:

```
if (условие)  
инструкция1;  
else  
инструкция2;
```

В случае **истинности** условия выполняется простая или составная **инструкция1**, а в случае **ложности** простая или составная **инструкция2**.

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

*До наступления
ночи*

Задано число
повторений

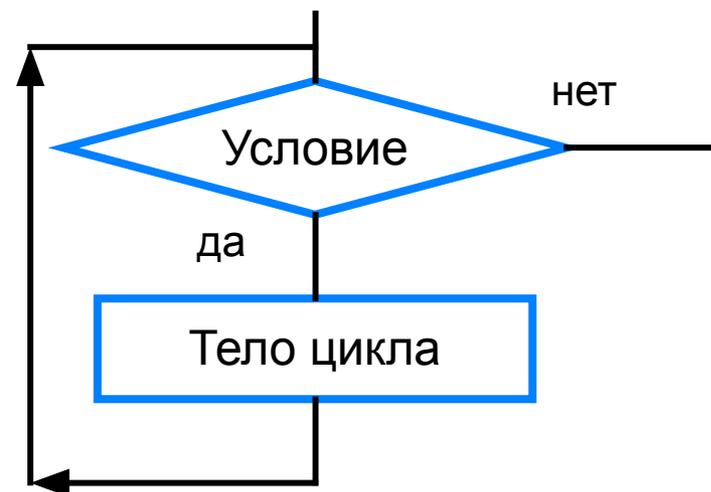
Ровно 100 кирпичей

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

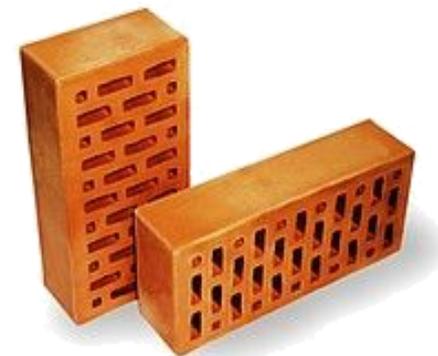
то положить кирпич в машину

иначе отложить кирпич в сторону

все

кц

кон



Цикл `while` в Java.

Конструкция `while` имеет следующий вид:

```
while(логическое_выражение) {  
    //тело цикла  
}
```

Пример: Вывод на экран значений от 1 до 10.

```
int i = 1;
while(i < 11){
    System.out.println("i= " + i);
    i++;
}
```

Конструкция `while` выполняет выражение в фигурных скобках до тех пор, пока параметр логическое_выражение имеет истинное значение (`true`). Этот параметр является условием выполнения цикла. В выше приведенном примере логическое

Для реализации
бесконечного цикла,
в качестве
параметра
достаточно указать
true

```
while(true) {  
    //тело цикла  
}
```

Досрочный выход из

<http://kostin.ws/java/java-if-else-logic.html>

<http://study-java.ru/uroki-java/java-operatory-tsikla-for-while-do-while-operator-break/>