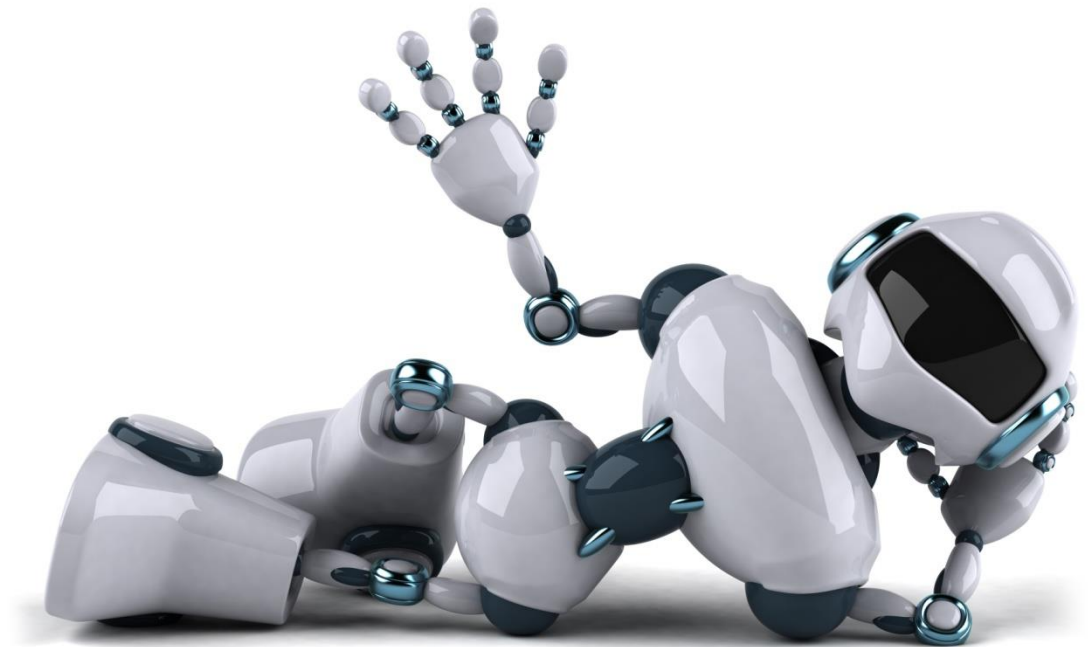
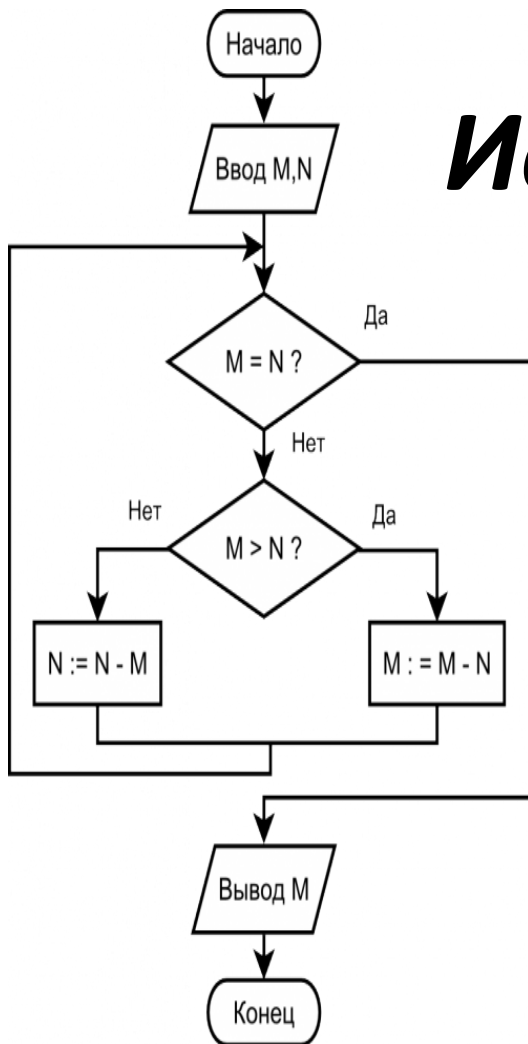
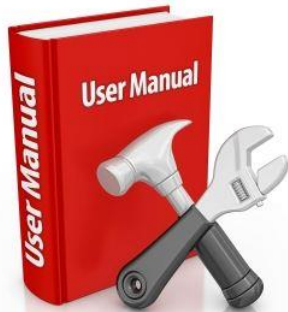


Понятие алгоритма и его свойства. Исполнитель алгоритмов



Что такое алгоритм?

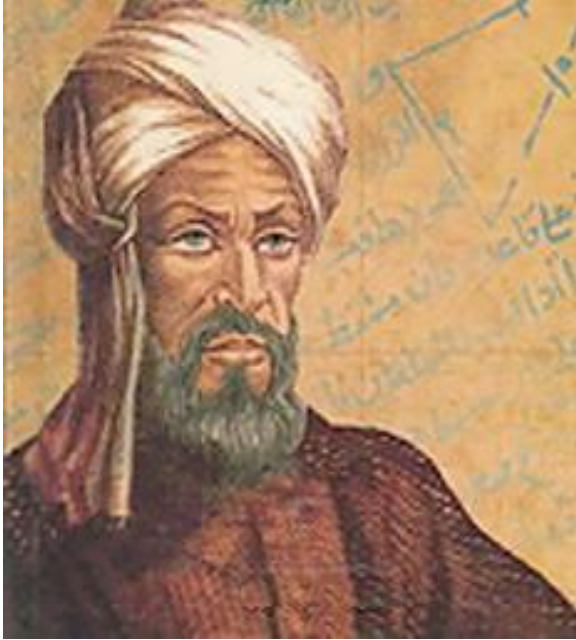
- Каждый из нас ежедневно использует различные алгоритмы: инструкции, правила, рецепты и т.п. Обычно мы это делаем не задумываясь.



- Например, открывая дверь ключом, никто не размышляет над тем, в какой последовательности выполнять действия. Однако чтобы научить кого-нибудь открывать дверь, придется четко указать и сами действия, и порядок их выполнения.
- То же потребуется и при указании маршрута поездки.



Происхождение слова "алгоритм"



Мухаммед ибн
Муса аль-Хорезми
(787-850)

- Слово "алгоритм" было введено в обращение выдающимся арабским математиком основателем алгебры в IX веке Мухаммед ибн Муса аль-Хорезми
- аль-Хорезми написал сочинение, в котором впервые дал описание позиционной десятичной системы счисления, а так же
- вывел правила арифметических действий над целыми и дробными числами. В переводе любое правило начиналось словами: «Алгоризми сказал...»

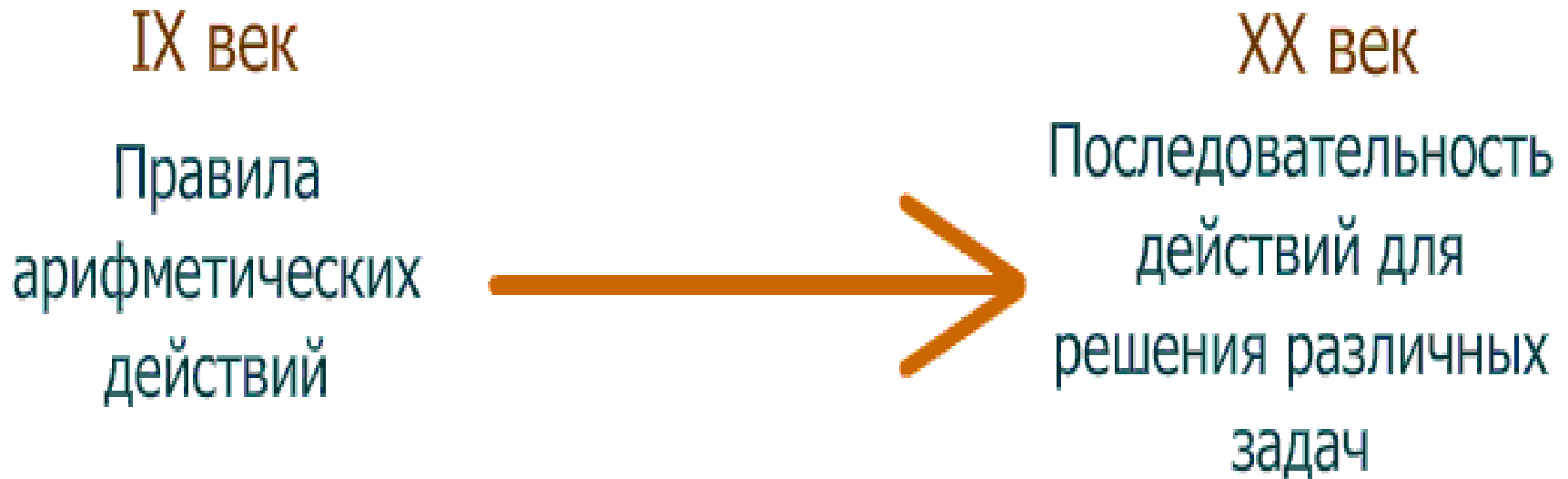
Использование понятия "алгоритм"



Готфрид Вильгельм
Лейбниц
(1646 — 1716)

- В науке использовать понятия "алгоритм" начал немецкий ученый **филосов, математик Лейбниц** в 17-м веке.
- **Первые известные алгоритмы** — это правила выполнения арифметических действий с числами. В них четко определены объекты (числа в десятичной записи) и элементарные шаги (сложить, вычесть, перемножить два однозначных числа).

Эволюция значения "алгоритм"



- Сначала определение понятия алгоритма было проблемой математики, однако с течением времени теория алгоритмов стала развиваться за счет влияния открытий не только в математике, но и в информатике. **В настоящее время алгоритм является одним из главных понятий информатики.**

Значительный вклад в развитие теории алгоритмов внесли:



Алан Мэ́тисон Тью́ринг
(1912 — 1954)

- Английский математик **Алан Тьюринг** в **1936** году предложил абстрактную **вычислительную «Машину Тьюринга»**, которая позволила формализовать понятие алгоритма и до сих пор используется во множестве теоретических и практических исследованиях

Значительный вклад в развитие теории алгоритмов внесли:



Эмиль Леон Пост
(1897 — 1954)

- Американский математик **Эмиль Пост** предложил абстрактную вычислительную машину – «**Машину Поста**», которая отличается от машины Тьюринга большей простотой. Обе машины «эквивалентны» и были созданы для уточнения понятия «алгоритм».

Значительный вклад в развитие теории алгоритмов внесли:



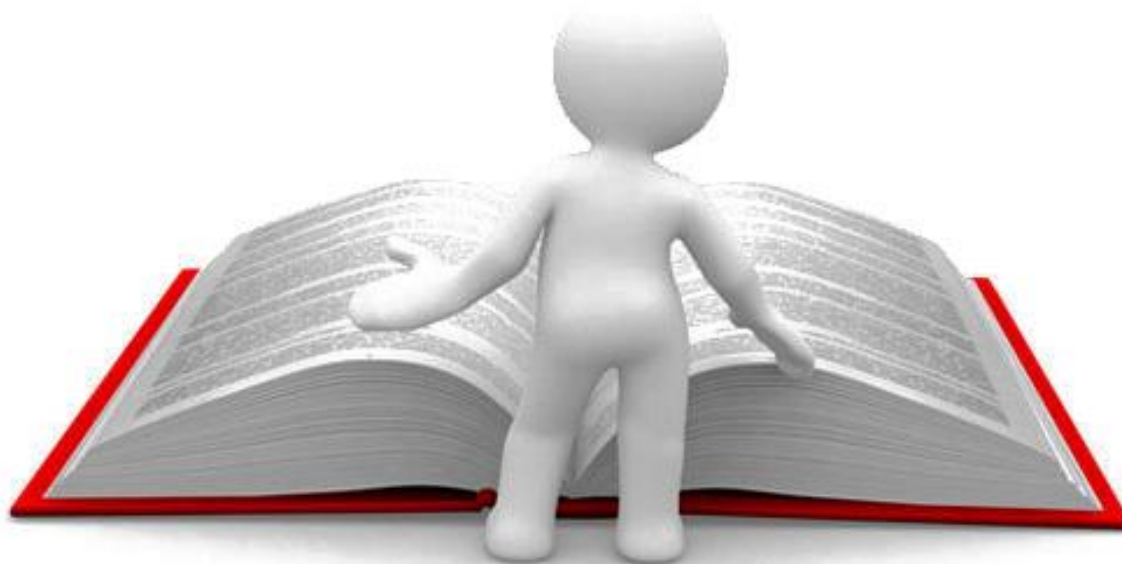
Alonzo Church
Алонзо Чёрч
(1903— 1995)

- Выдающийся американский математик **Алонзо Чёрч** разработал теорию лямбда-исчисление, последовавшей за его знаменитой статьёй 1936 года, в которой он показал существование «неразрешимых задач» (**теорема Чёрча — Тьюринга**)
- Впоследствии Чёрч и Тьюринг показали, что лямбда-исчисления и машина Тьюринга имели одинаковые свойства, таким образом доказывая, что различные «механические процессы вычислений» имеют одинаковые возможности.

Алгоритм

от лат. Algorithm (написание имени аль-Хорезми)

- набор инструкций, описывающих строгий и четкий порядок действий исполнителя, выполнение которых приводит к достижению результата решения задачи за конечное число действий



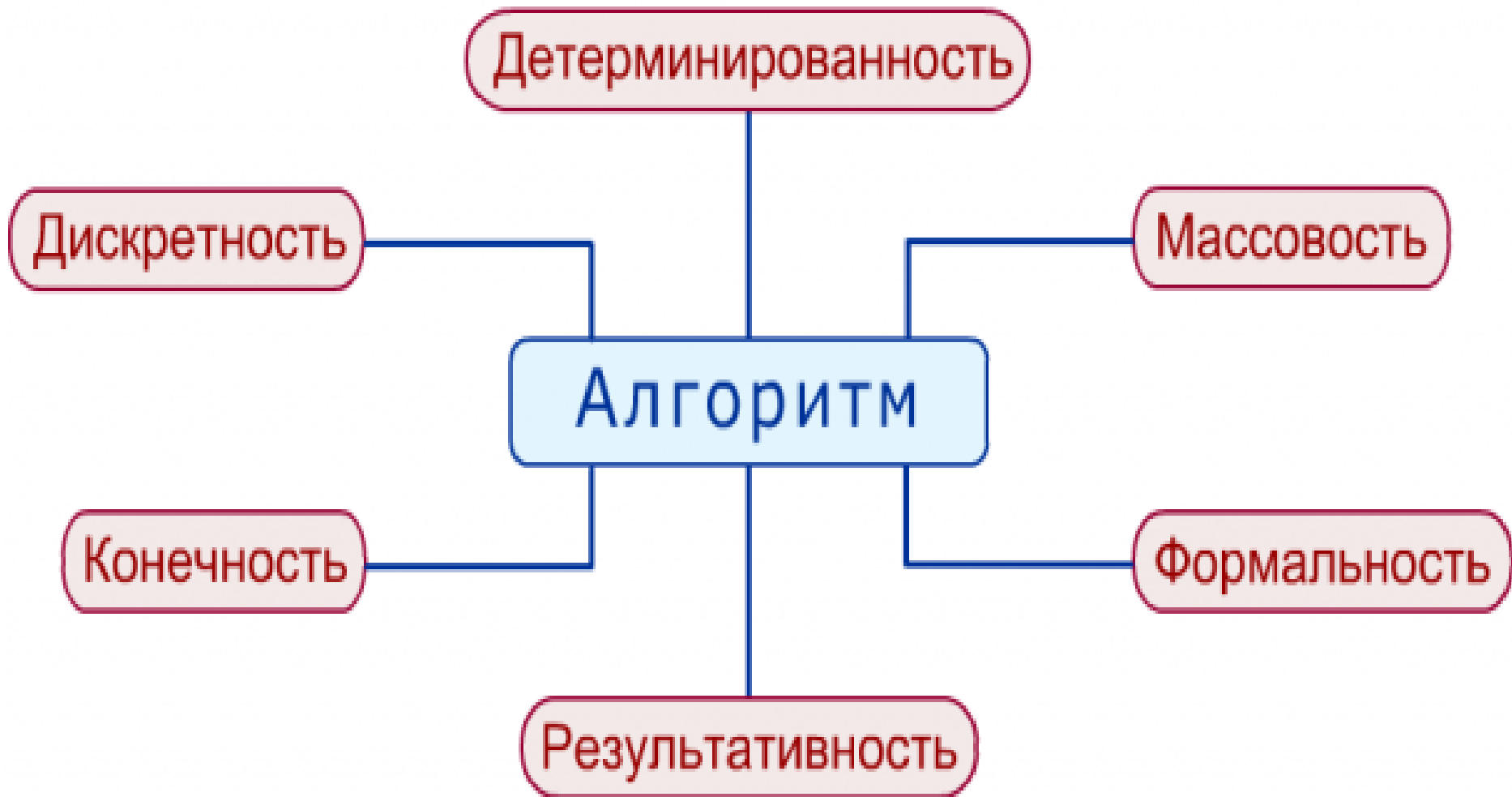
Верно ли, записан алгоритм ...

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



Свойства алгоритмов

- Для любого алгоритма справедливы общие закономерности
- свойства алгоритма

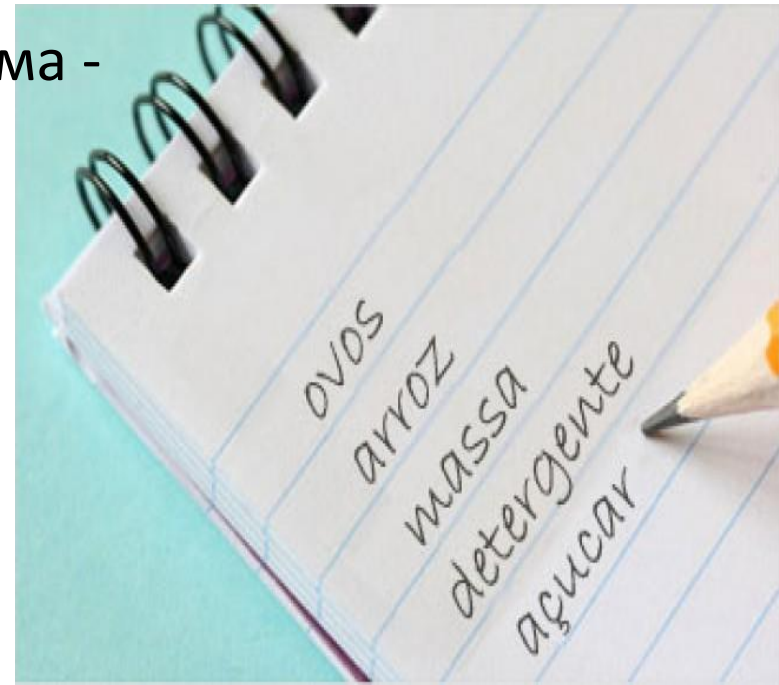


Детерминированность алгоритма

- Это определенность (однозначность, единственность) толкования правил выполнения действий. То есть при задании одних и тех же исходных данных несколько раз алгоритм будет выполняться абсолютно одинаково и всегда будет получен один и тот же результат.

Пример детерминированности алгоритма - список товаров для покупки.

- Как указание купить все эти товары в любом порядке.
Это недетерминированный алгоритм.
- Как **указание купить все эти товары в данном порядке.**
Это детерминированный алгоритм.



Массовость алгоритма

- Возможность применения алгоритма к множеству однотипных задач (один и тот же алгоритм можно использовать с разными исходными данными)

Десятичная:

$$15_{10} + 6_{10}$$

$$\begin{array}{r} 1 \\ 15 \\ + 6 \\ \hline 21 \end{array}$$

5+6=11=10+1
1+1=2

Двоичная:

$$11111_2 + 110_2$$

$$\begin{array}{r} 111 \\ + 1111 \\ 0110 \\ \hline 10101 \end{array}$$

1+0=1
1+1=2=10
1+1+1=3=11
1+1=2=10

Восьмеричная:

$$17_8 + 6_8$$

$$\begin{array}{r} 1 \\ 17 \\ + 6 \\ \hline 25 \end{array}$$

7+6=13=8+5
1+1=2

Пример **массовости** алгоритма - сложение (вычитание, умножение и деление) могут быть применены для любых чисел, причем не только в десятичной, но и в других позиционных системах счисления (двоичной, восьмеричной, шестнадцатеричной и др.)

Результативность и конечность алгоритма

- Возможность получения из исходных данных нужного результата по окончании алгоритма за конечное число шагов

A grid-in subtraction problem. The top row contains the number 27, and the bottom row contains the number 22. A minus sign is to the left of the 27, and a horizontal line is drawn between the two numbers.

A grid-in addition problem. The top row contains the number 52, and the bottom row contains the number 58. A plus sign is to the left of the 52, and a horizontal line is drawn between the two numbers.

A grid-in multiplication problem. The top row contains the number 925, and the bottom row contains the number 38. A multiplication sign (x) is to the left of the 38, and a horizontal line is drawn between the two numbers. The result 35150 is written in the bottom row.

Пример **результативности** алгоритма - правила сложения (вычитания, умножения и деления столбиком). Применение этих алгоритмов всегда приводит к результату.

Формальность алгоритма

- Это понятность алгоритма, каждая команда должна определять однозначное действие исполнителя, не допуская разных толкований

Пример формальности алгоритма – алгоритм мытья рук. Ребенок в детском саду, действует формально, то есть не задумываясь строго выполняет картинки-инструкции



Дискретность алгоритма

- Алгоритм должен содержать четкие описания отдельных шагов (действий), следующих в определенном порядке. Нельзя перейти к следующему действию, не закончив предыдущего.

Пример дискретности алгоритма - кулинарные рецепты, состоят из отдельных действий приготовления пищи, которые обычно нумеруются.

Готовим бисквит:



1. Разбиваем яйца и смешиваем с сахаром.

2. Ставим на водяную баню и помешиваем венчиком до полного растворения сахара.



3. Снимаем с водяной бани и взбиваем миксером около 10 минут. Взбиваем до тех пор, чтобы на белках оставался след от проведенного по ним пальца



Пример алгоритма

1. Подойти к реке.
2. Войти в реку.
3. Идти по дну, пока не выйдешь на другой берег.



Выполним ли этот алгоритм,
если человек подошёл к реке Волге?

Пример алгоритма



Ответ однозначен - нет.

А в каком случае этот алгоритм будет выполнен, и кто может пройти по дну реки Волги?

- Оказывается, алгоритм выполнится, если по дну пойдёт человек со специальным снаряжением или робот-“подводник”.
- Следовательно, один и тот же алгоритм может быть верным и неверным в зависимости от того, каковы исходные данные и кто **исполнитель алгоритма**.

Исполнитель алгоритма

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



Рассмотрим пример:

- Имеется **исполнитель - старик**. Он должен переправить на лодке через реку волка, козу и капусту.
- Лодка может выдержать только старика и одного пассажира. В каком порядке старик перевезет пассажиров? Не забудь, что волк может съесть козу, а коза — капусту.
- *Перед нами “гипотетический” человек, который, строго руководствуясь алгоритмом, решает задачу.*
- *Составим для данного исполнителя алгоритм решения задачи*

Алгоритм решения задачи



Алгоритм решения задачи

Шаг	Левый берег	Действие	Правый берег
1	Волк, капуста	Перевези козу	
2	Волк, капуста	Переправься обратно один	Коза
3	капуста	Перевези волка	Коза
4	капуста	Перевези козу	Волк
5	Коза	Перевези капусту	Волк
6	Коза	Переправься обратно один	Волк, капуста
7		Перевези козу	Волк, капуста, Коза

- Исполнителем указанных действий является человек - перевозчик, решающий задачу по алгоритму машинально.
- **Исполнителем** может быть и какое-то **устройство**

Представим...

- Небольшое устройство, похожее на электронную игру



- Нажимая на кнопки, мы можем передвигать существа по экрану. Других способов управления ими в данном устройстве нет. Последовательность действий для решения задачи будет состоять из нажатий на кнопки

Система
команд
исполнителя
(СКИ)

Среда
исполнителя

Система
отказов
(ошибок)
исполнителя

Исполнитель

```
graph TD; SKI[Система команд исполнителя (СКИ)] --> Исполнитель((Исполнитель)); Среда[Среда исполнителя] --> Исполнитель; СистемаОтказов[Система отказов (ошибок) исполнителя] --> Исполнитель;
```

Система команд исполнителя (СКИ)

- **Команда** – это указание исполнителю совершить некоторое действие.
- После вызова команды исполнитель совершает соответствующее **элементарное действие**.
- Совокупность команд из некоторого строго заданного списка, которые данный исполнитель может выполнять, называется **системой команд исполнителя (СКИ)**.

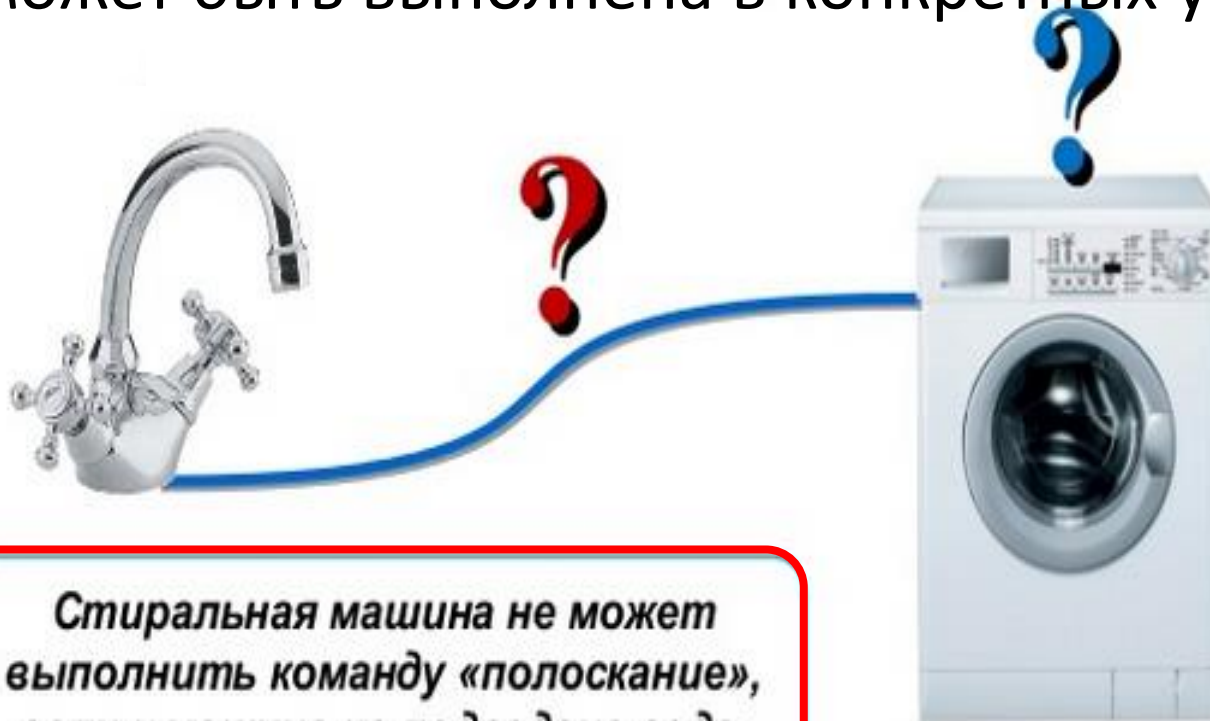
Система команд исполнителя (СКИ) стиральной машинки

- Замачивание
- Стирка
- Полоскание
- Отжим
- Сушка



Система отказов исполнителя

- Отказ **«Не понимаю»** возникает, если подается команда, не входящая в СКИ.
- Отказ **«Не могу»** возникает, если команда из СКИ не может быть выполнена в конкретных условиях среды.



Стиральная машина не может выполнить команду «полоскание», если к машине не подведена вода.

Стиральная машина не может выполнить команду «гладить» так как ее нет в системе команд

Среда исполнителя

- область, обстановка, условия и объекты (данные), над которыми исполнитель может выполнять действия, формируют **среду исполнителя**



Задача

- Опишите для робота - повара среду исполнителя
- Напишите для робота - повара СКИ и алгоритм приготовления чая



Решение Задачи

СКИ:

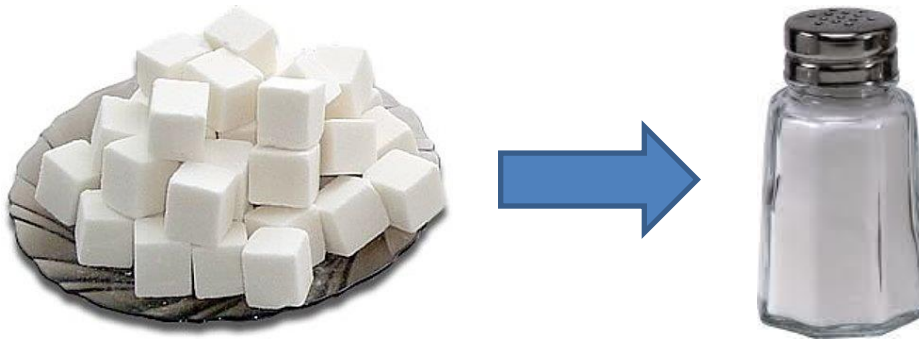
- налить кипяток
- помешать
- налить молоко
- насыпать сахар
- насыпать заварку

Алгоритм :

- насыпать заварку
- налить кипяток
- насыпать сахар
- налить молоко
- помешать

Вопросы:

1. Будет ли выполняться алгоритм, если исполнителю вместо сахара подсунуть соль?



2. Какие команды нужно поменять местами, чтобы результат выполнения алгоритма изменился?

Типы исполнителей

Формальные



Не формальные



Выполните предложенные действия:

1.Загадай число.

6

2.Умножь на 5.

$$6*5=30$$

3.Прибавь 8.

$$30+8=38$$

4.Умножь на 2.

$$38*2=76$$

5.Отними 16.

$$76-16=60$$

6.Отбрось крайнюю правую
цифру и получишь
загаданное число.

$$60 \rightarrow 6$$

**Вы выступили в роли
формального исполнителя**

Исполнители

Формальные

Не формальные

В роли
неформального
исполнителя чаще
всего выступает
человек

В роли формального
исполнителя чаще
всего выступает
**техническое
устройство**

Неформальный
исполнитель **сам
отвечает** за свои
действия

За действия
формального
исполнителя
**отвечает
управляющий им
объект**



Не формальный исполнитель

- Неформальный исполнитель (экскурсовод) **не может** выполнять одни и те же команды (действия) совершенно одинаково.



Формальный исполнитель

- Формальный исполнитель всегда **одинаково** выполняет одну и ту же команду.

Для каждого формального исполнителя можно указать:

- круг решаемых задач;
- среду;
- систему команд;
- систему отказов;
- режимы работы.



Автоматический фасовочно-упаковочный аппарат

Режимы работы исполнителя



Непосредственный



Программный



***СВЧ-печь может
выполнять
отдельную команду
– разогревать пищу***

***СВЧ-печь способна
приготовить
сложное блюдо по
заданной программе***

Домашнее задание

Найдите второй вариант решения задачи:

Старик должен переправить на лодке через реку волка, козу и капусту.

Лодка может выдержать только старика и одного пассажира.

В каком порядке старик перевезет пассажиров?

Не забудь, что волк может съесть козу, а коза – капусту.