Алгоритмы. Модели алгоритмов

**Алгори́тм** ([лат.](https://ru.wikipedia.org/wiki/%D0%9B%D0%B0%D1%82%D0%B8%D0%BD%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *algorithmi* — от имени среднеазиатского математика [Аль-Хорезми](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D1%8C-%D0%A5%D0%BE%D1%80%D0%B5%D0%B7%D0%BC%D0%B8)[[1]](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC#cite_note-1)) — конечная совокупность точно заданных правил решения некоторого класса задач или набор [инструкций](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), описывающих порядок действий исполнителя для решения определённой задачи. В старой трактовке вместо слова «порядок» использовалось слово «последовательность», но по мере развития [параллельности в работе компьютеров](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC) слово «последовательность» стали заменять более общим словом «порядок». Независимые инструкции могут выполняться в произвольном порядке, параллельно, если это позволяют используемые исполнители.

**Свойства алгоритмов**

Различные определения алгоритма в явной или неявной форме содержат следующий ряд общих требований:

* [Дискретность](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D1%81%D0%BA%D1%80%D0%B5%D1%82%D0%BD%D0%BE%D1%81%D1%82%D1%8C) — алгоритм должен представлять процесс решения задачи как упорядоченное выполнение некоторых простых шагов. При этом для выполнения каждого шага алгоритма требуется конечный отрезок времени, то есть преобразование исходных данных в результат осуществляется во времени дискретно.
* [Детерминированность](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%82%D0%B5%D1%80%D0%BC%D0%B8%D0%BD%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D1%81%D1%82%D1%8C) (определённость). В каждый момент времени следующий шаг работы однозначно определяется состоянием системы. Таким образом, алгоритм выдаёт один и тот же результат (ответ) для одних и тех же исходных данных. В современной трактовке у разных реализаций одного и того же алгоритма должен быть изоморфный [граф](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%B0). С другой стороны, существуют вероятностные алгоритмы, в которых следующий шаг работы зависит от текущего состояния системы и генерируемого случайного числа. Однако при включении метода генерации случайных чисел в список «исходных данных» вероятностный алгоритм становится подвидом обычного.
* Понятность — алгоритм должен включать только те команды, которые доступны исполнителю и входят в его систему команд.
* Завершаемость (конечность) — в более узком понимании алгоритма как математической функции, при правильно заданных начальных данных алгоритм должен завершать работу и выдавать результат за определённое число шагов. [Дональд Кнут](https://ru.wikipedia.org/wiki/%D0%94%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%B4_%D0%9A%D0%BD%D1%83%D1%82) процедуру, которая удовлетворяет всем свойствам алгоритма, кроме, возможно, конечности, называет *методом вычисления* ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *computational method*). Однако довольно часто определение алгоритма не включает завершаемость за конечное время. В этом случае алгоритм (метод вычисления) определяет [частичную функцию](https://ru.wikipedia.org/w/index.php?title=%D0%A7%D0%B0%D1%81%D1%82%D0%B8%D1%87%D0%BD%D0%B0%D1%8F_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F&action=edit&redlink=1). Для [вероятностных алгоритмов](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D1%80%D0%BE%D1%8F%D1%82%D0%BD%D0%BE%D1%81%D1%82%D0%BD%D1%8B%D0%B9_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC) завершаемость как правило означает, что алгоритм выдаёт результат с вероятностью 1 для любых правильно заданных начальных данных (то есть может в некоторых случаях не завершиться, но вероятность этого должна быть равна 0).
* Массовость (универсальность). Алгоритм должен быть применим к разным наборам начальных данных.
* Результативность — завершение алгоритма определёнными результатами.

Формальное определение

Разнообразные теоретические проблемы математики и ускорение развития физики и техники поставили на повестку дня точное определение понятия алгоритма.

Первые попытки уточнения понятия алгоритма и его исследования осуществляли в первой половине XX века [Алан Тьюринг](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B0%D0%BD_%D0%A2%D1%8C%D1%8E%D1%80%D0%B8%D0%BD%D0%B3), [Эмиль Пост](https://ru.wikipedia.org/wiki/%D0%AD%D0%BC%D0%B8%D0%BB%D1%8C_%D0%9F%D0%BE%D1%81%D1%82), [Жак Эрбран](https://ru.wikipedia.org/wiki/%D0%96%D0%B0%D0%BA_%D0%AD%D1%80%D0%B1%D1%80%D0%B0%D0%BD), [Курт Гедель](https://ru.wikipedia.org/wiki/%D0%9A%D1%83%D1%80%D1%82_%D0%93%D0%B5%D0%B4%D0%B5%D0%BB%D1%8C), [А. А. Марков](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%80%D0%BA%D0%BE%D0%B2,_%D0%90%D0%BD%D0%B4%D1%80%D0%B5%D0%B9_%D0%90%D0%BD%D0%B4%D1%80%D0%B5%D0%B5%D0%B2%D0%B8%D1%87_(%D0%BC%D0%BB%D0%B0%D0%B4%D1%88%D0%B8%D0%B9)), [Алонзо Чёрч](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%BE%D0%BD%D0%B7%D0%BE_%D0%A7%D1%91%D1%80%D1%87" \o "Алонзо Чёрч). Было разработано несколько определений понятия алгоритма, но впоследствии было выяснено, что все они определяют одно и то же понятие (см. [Тезис Чёрча — Тьюринга](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%B7%D0%B8%D1%81_%D0%A7%D1%91%D1%80%D1%87%D0%B0_%E2%80%94_%D0%A2%D1%8C%D1%8E%D1%80%D0%B8%D0%BD%D0%B3%D0%B0))[[6]](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC#cite_note-6)

Российский математик, основоположник [структурной лингвистики](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%BD%D0%B0%D1%8F_%D0%BB%D0%B8%D0%BD%D0%B3%D0%B2%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B0) в Советском Союзе [В. А. Успенский](https://ru.wikipedia.org/wiki/%D0%A3%D1%81%D0%BF%D0%B5%D0%BD%D1%81%D0%BA%D0%B8%D0%B9,_%D0%92%D0%BB%D0%B0%D0%B4%D0%B8%D0%BC%D0%B8%D1%80_%D0%90%D0%BD%D0%B4%D1%80%D0%B5%D0%B5%D0%B2%D0%B8%D1%87) считал, что понятие алгоритма впервые появилось у Эмиля Бореля в 1912 году, в статье об определённом интеграле. Там он написал о «вычислениях, которые можно реально осуществить», подчеркивая при этом: «Я намеренно оставляю в стороне большую или меньшую практическую деятельность; суть здесь та, что каждая из этих операций осуществима в конечное время при помощи достоверного и недвусмысленного метода».

#### Машина Тьюринга

Основная идея, лежащая в основе машины Тьюринга, очень проста. Машина Тьюринга — это абстрактная машина (автомат), работающая с лентой отдельных ячеек, в которых записаны символы. Машина также имеет головку для записи и чтения символов из ячеек, которая может двигаться вдоль ленты. На каждом шаге машина считывает символ из ячейки, на которую указывает головка, и, на основе считанного символа и внутреннего состояния, делает следующий шаг. При этом машина может изменить своё состояние, записать другой символ в ячейку или передвинуть головку на одну ячейку вправо или влево.[[8]](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC#cite_note-gmbm-8)

На основе исследования этих машин был выдвинут тезис Тьюринга (основная гипотеза алгоритмов):

*Некоторый алгоритм для нахождения значений функции, заданной в некотором алфавите, существует тогда и только тогда, когда функция исчисляется по Тьюрингу, то есть когда её можно вычислить на машине Тьюринга.*

Этот тезис является аксиомой, постулатом, и не может быть доказан математическими методами, поскольку алгоритм не является точным математическим понятием.

#### Рекурсивные функции

С каждым алгоритмом можно сопоставить функцию, которую он вычисляет. Однако возникает вопрос, можно ли произвольной функции сопоставить машину Тьюринга, а если нет, то для каких функций существует алгоритм? Исследования этих вопросов привели к созданию в 1930-х годах теории рекурсивных функций[[9]](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC#cite_note-9).

Класс вычислимых функций был записан в образ, напоминающий построение некоторой аксиоматической теории на базе системы аксиом. Сначала были выбраны простейшие функции, вычисление которых очевидно. Затем были сформулированы правила (операторы) построения новых функций на основе уже существующих. Необходимый класс функций состоит из всех функций, которые можно получить из простейших применением операторов.

Подобно тезису Тьюринга в теории вычислимых функций была выдвинута гипотеза, которая называется [*тезис Чёрча*](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%B7%D0%B8%D1%81_%D0%A7%D1%91%D1%80%D1%87%D0%B0_%E2%80%94_%D0%A2%D1%8C%D1%8E%D1%80%D0%B8%D0%BD%D0%B3%D0%B0):

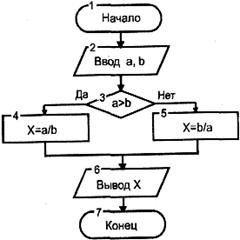
*Числовая функция тогда и только тогда алгоритмически исчисляется, когда она частично рекурсивна.*

Доказательство того, что класс вычислимых функций совпадает с исчисляемыми по Тьюрингу, происходит в два шага: сначала доказывают вычисление простейших функций на машине Тьюринга, а затем — вычисление функций, полученных в результате применения операторов.

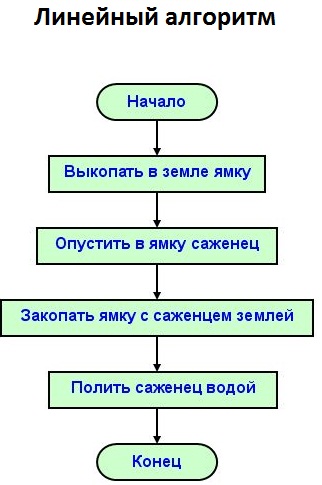
Таким образом, неформально алгоритм можно определить как четкую систему инструкций, определяющих дискретный детерминированный процесс, который ведёт от начальных данных (на входе) к искомому результату (на выходе), если он существует, за конечное число шагов; если искомого результата не существует, алгоритм или никогда не завершает работу, либо заходит в тупик.

***Примеры***

1. Задача:http://koi.tspu.ru/vav/vav_umk_inf/algoritmiz/image007.gif

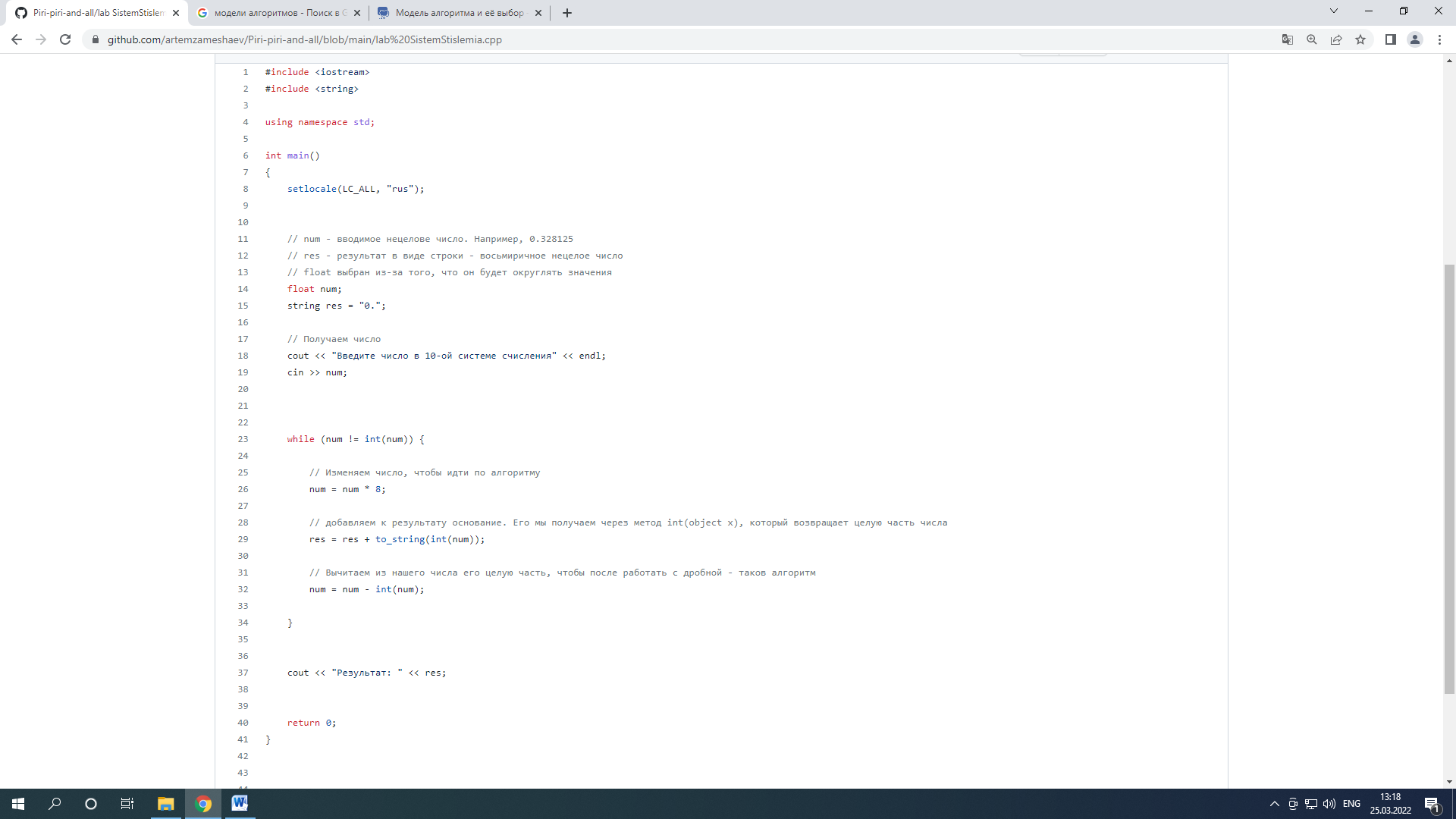
Её алгоритм: 

1. Задача: Посадить семечко

Алгоритм: 

1. Задача: Перевести 1-тичное дробное число в 8-ричную СС

Алгоритм, реализованный на языке программирования C++



Источники информации

* [Wiki](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC)
* [Habr](https://habr.com/ru/hub/algorithms/)
* [Tadviser](https://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC)
* [Neerc](http://neerc.ifmo.ru/wiki/index.php?title=%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%B0_%D0%B8_%D0%B5%D1%91_%D0%B2%D1%8B%D0%B1%D0%BE%D1%80)

Примеры алгоритмов:

* [Google изображения](https://www.google.com/search?q=%D0%BF%D1%80%D0%B8%D0%BC%D0%B5%D1%80%D1%8B+%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D1%8B&source=lnms&tbm=isch&sa=X&ved=2ahUKEwiZwujosuD2AhVJAxAIHSbfCH0Q_AUoAXoECAEQAw&biw=1920&bih=969&dpr=1#imgrc=7u942SUgHP2u6M)
* [Github](https://github.com/artemzameshaev/Piri-piri-and-all/blob/main/lab%20SistemStislemia.cpp)