**Алгоритм** – набор шагов или инструкций, необходимых для выполнения задачи. Это набор конечного числа правил, задающих последовательность выполнения операций компьютерной программы для решения задачи определённого типа.

**Зачем изучать алгоритмы?**

* Существует устоявшийся массив знаний о том, как хорошо решать конкретные проблемы, и важно знать, какие решения есть сейчас.
* Необходимо знать не только о существовании подходящего алгоритма, но и понимать, когда его применить, определять типа задачи.

**Свойства алгоритма:**

* *Конечность*. Алгоритм всегда должен заканчиваться после выполнения конечного числа шагов.
* *Определённость*. Действия, которые нужно выполнить, должны быть строго и недвусмысленно определены для каждого возможного случая.
* *Ввод*. Алгоритм имеет некоторое (возможно, равное нулю) число входных данных.
* *Вывод*. У алгоритма есть одно или несколько выходных данных, т.е. величин, имеющих вполне определённую связь с входными данными.
* *Эффективность*. Алгоритм обычно считается эффективным, если все его операторы достаточно просты для того, чтобы их можно было точно выполнить в течение конечного промежутка времени с помощью карандаша и бумаги.

**Алгоритм Евклида**

Даны два целых положительных числа m и n. Требуется найти их наибольший общий делитель, т.е. наибольшее целое положительное число, которое нацело делит оба числа m и n.

*Е1*. [Нахождение остатка] Разделим m на n, и пусть отсаток от деления будет равен r (где 0 <= r <= n).

*Е2*. [Сравнение с нулём] Если r = 0, то выполнение алгоритма прекращается; n – искомое значение.

*Е3*. [Замещение] Присвоить m <- n, n <- r и вернуться к шагу Е1.

**Память**

Использование основной памяти (обычно ОЗУ) нужной алгоритму.

Для анализа алгоритма обычно используется анализ пространственной сложности алгоритма, чтобы оценить необходимую память времени исполнения как функцию от размера входа. Результат обычно выражается в терминах «О» большое.

**Аспекты использования памяти:**

* Количество памяти, необходимой для хранения кода алгоритма.
* Количество памяти, необходимой для входных данных.
* Количество памяти, необходимое для любых входных данных

**Время**

Для анализа алгоритма обычно используется анализ временной сложности алгоритма, чтобы оценить время работы как функцию от размера входа. Результат обычно выражается в терминах «О» большое.

**О-символика как инструмент оценки ресурсов, асимптотики**

«О» большое – математическое обозначение для сравнения асимптотического поведения функций.

Если f(n) является функцией от положительного целого n, можно использовать запись O(f(n)); она обозначает величину, точное значение которой неизвестно, и известно только, что её значение не слишком велико.

Запись O(f(n)) всегда обозначает следующее: существуют положительное константы M и n0, такие, что величина xn, представленная в виде O(f(n)), удовлетворяет условию   
|xn| <= M |f(n)| для всех целых n >= n0. [Кнут. Искусство программирования]

Фраза «сложность алгоритма есть O(f(n))» означает, что с увеличением параметра n, характеризующего количество входной информации алгоритма, время работы алгоритма будет возрастать не быстрее, чем f(n), умноженная на некоторую константу.

*Операции над символом О [дописать]*

**Метод математической индукции**

1. Доказать, что Р(1) верно.
2. Доказать, что «если Р(1), Р(2)…., Р(n) справедливы, то Р(n + 1) также справедливо»; это доказательство должно иметь силу для любого целого положительного n.

1 + 3 + … + (2n – 1) = n2

P(n + 1): 1 + 3 + … + (2n – 1) + (2n + 1) = n2 + 2n + 1 = (n+1)2