

Алгоритмы

АлгориТм или алгоритФм?

- Обычно: алгориТм
- Марков: алгоритФм
- Названия эквивалентны

Понятие алгоритма

- Инструкция, какие действия нужно предпринять для достижения результата

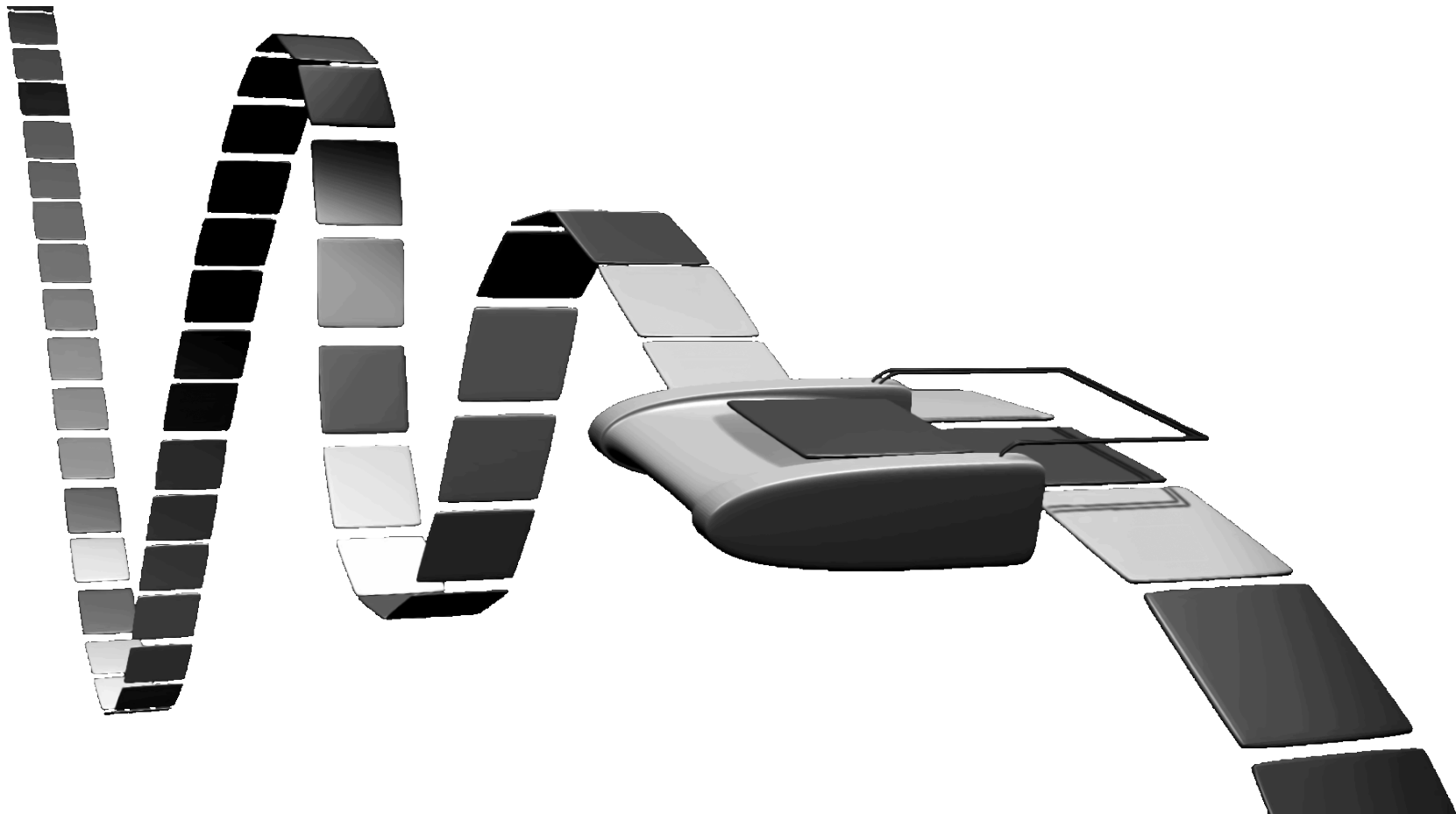
Классы алгоритмов

- Механические, детерминированные, жёсткие — определённые действия в чёткой последовательности
- Вероятностные - несколько путей, приводящих к вероятному достижению результата.
- Параллельные — для работы параллельно исполняющихся программ, процессов, нитей.
- Эвристические — на основании соображений без строгого обоснования

Алгоритмы в курсах ФУПМ

- Детерминированные - информатика (1 семестр)
- В приложении к олимпиадным задачам «Алгоритмы: построение и анализ»
- Теория формальных систем и алгоритмов (3 семестр).
- Детерминированные на графах «Алгоритмы и модели вычислений», 4 семестр
- Методы оптимизации (5-8 семестр)
- Параллельные алгоритмы (информатика, 3 семестр; параллельное программирование, 6-7 семестры; сетевые технологии, 7-8 семестры)
- Приближённые алгоритмы — «эффективные алгоритмы», 11 семестр.

Модели для описания алгоритмов: машины Тьюринга и Поста



Нормальный алгоритм Маркова

$$\left\{ \begin{array}{lcl} |b & \rightarrow & ba| \\ ab & \rightarrow & ba \\ b & \rightarrow & \\ *| & \rightarrow & b* \\ * & \rightarrow & c \\ |c & \rightarrow & c \\ ac & \rightarrow & c| \\ c & \rightarrow & . \end{array} \right.$$

Примитивно рекурсивные функции

Базовые:

- Нулевая функция O — функция без аргументов, всегда возвращающая 0.
- Функция следования S одного переменного, сопоставляющая любому натуральному числу x непосредственно следующее за ним натуральное число $x+1$.
- Функции I_n^m , где $0 < m \leq n$, от n переменных, сопоставляющие любому упорядоченному набору x_1, \dots, x_n натуральных чисел число x_m из этого набора.

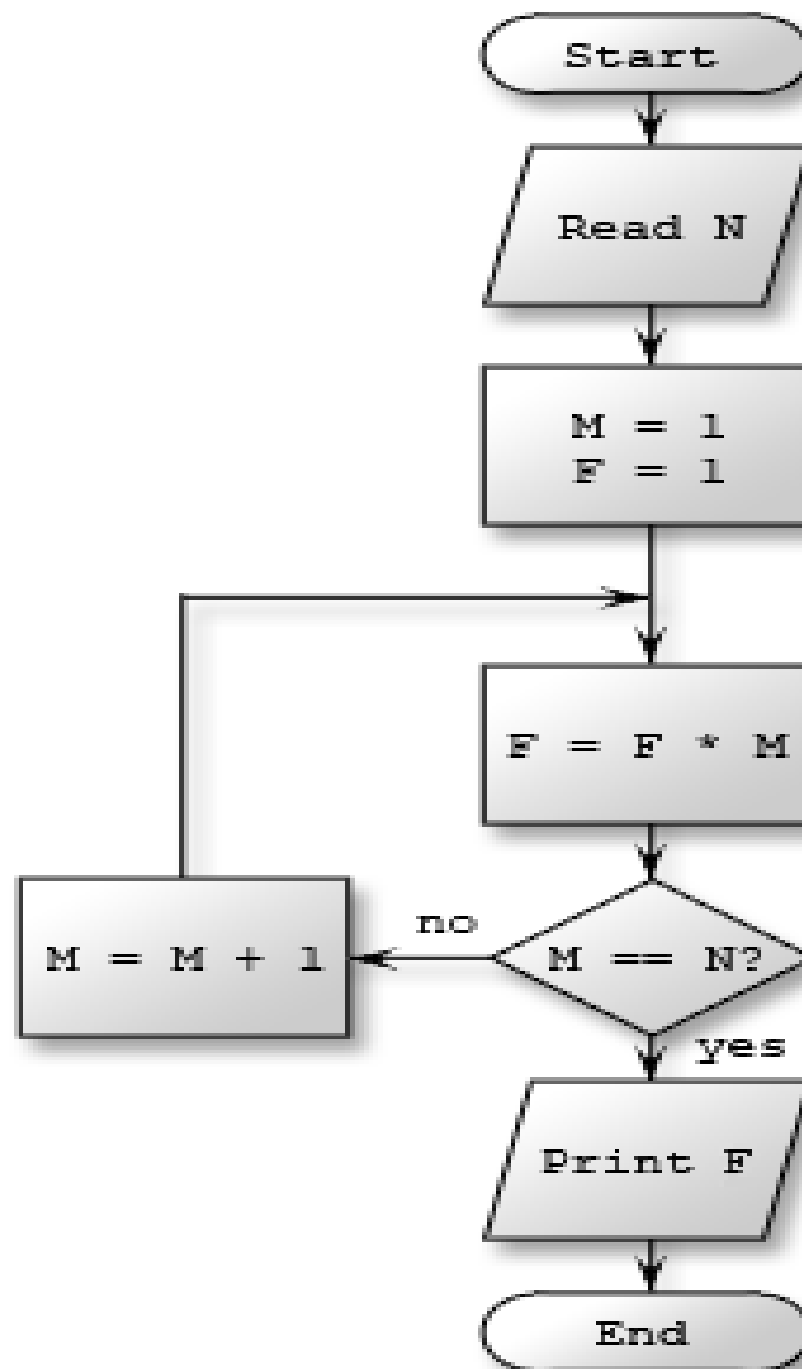
Оператор суперпозиции (иногда — оператор подстановки). Пусть f — функция от m переменных, а g_1, \dots, g_m — упорядоченный набор функций от n переменных каждая. Тогда результатом суперпозиции функций g_k в функцию f называется функция h от n переменных, сопоставляющая любому упорядоченному набору x_1, \dots, x_n натуральных чисел число

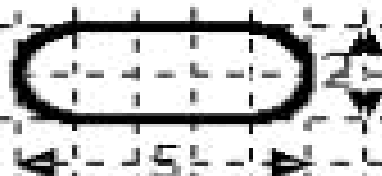

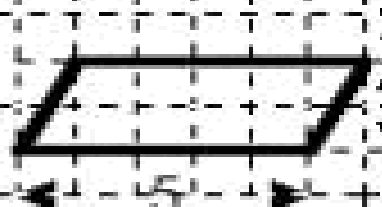
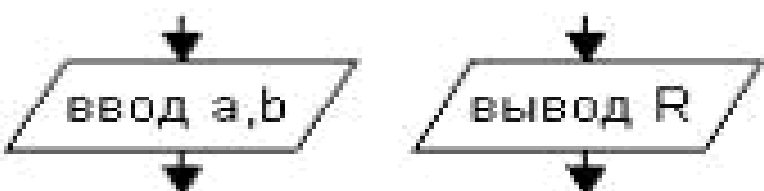
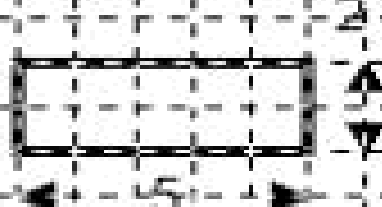
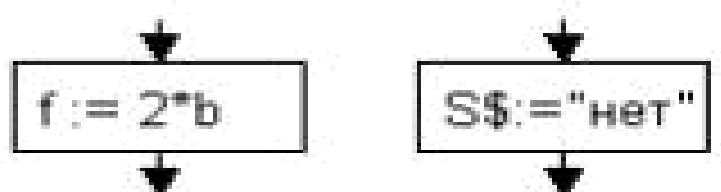
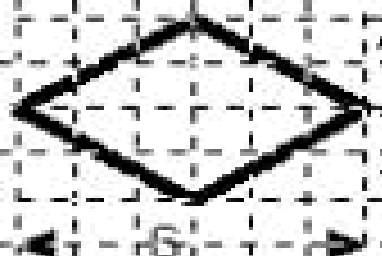
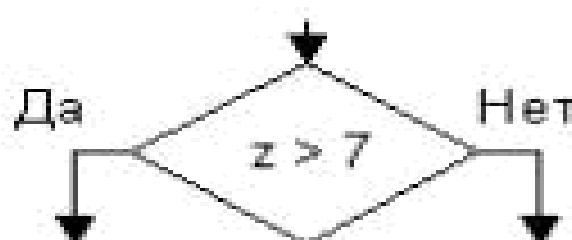
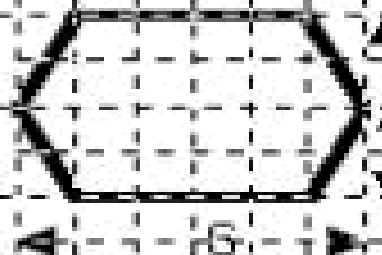
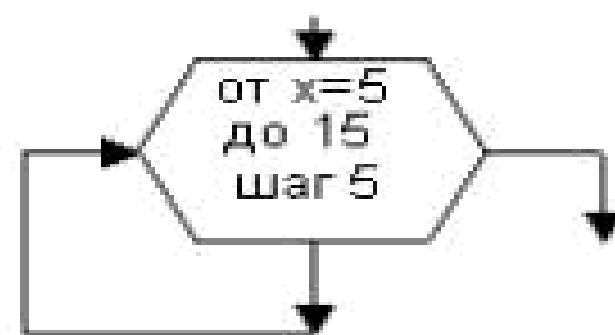
- $h(x_1, \dots, x_n) = f(g_1(x_1, \dots, x_n), \dots, g_m(x_1, \dots, x_n))$.

Оператор примитивной рекурсии. Пусть f — функция от n переменных, а g — функция от $n+2$ переменных. Тогда результатом применения оператора примитивной рекурсии к паре функций f и g называется функция h от $n+1$ переменной вида

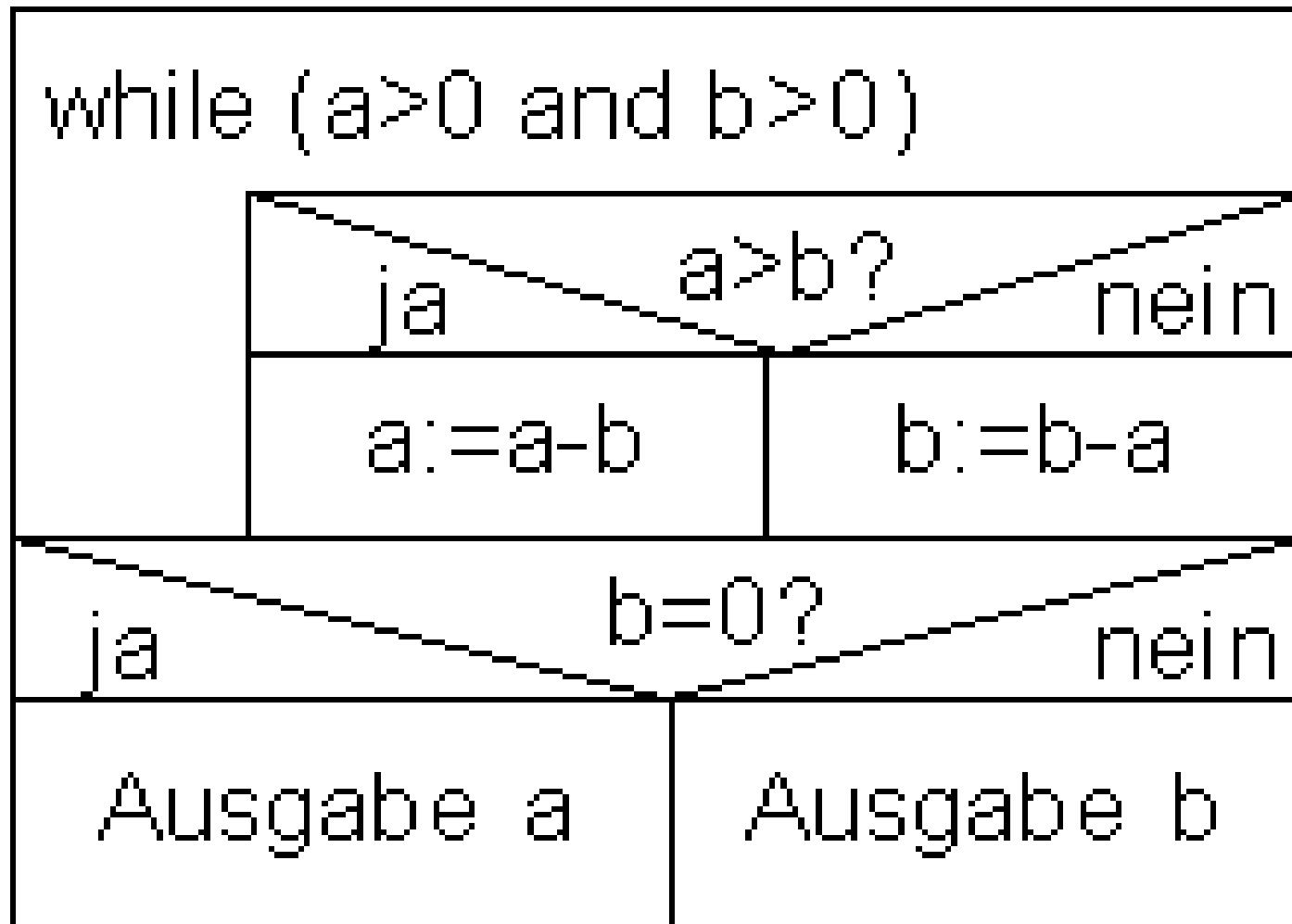
- $h(x_1, \dots, x_n, 0) = f(x_1, \dots, x_n)$;
- $h(x_1, \dots, x_n, y+1) = g(x_1, \dots, x_n, y, h(x_1, \dots, x_n, y))$.

Блок- схемы



вид блока	название / назначение	примеры записи
	блок начала / конца алгоритма обозначает начало или конец алгоритма	
	блок ввода / вывода служит для ввода исходных данных и вывода результатов	
	блок действия служит для записи команды присваивания	
	блок логического условия служит для организации ветвления в алгоритме	
	блок цикла служит для организации циклов в алгоритме	

Схемы Насси-Шнейдермана



$O(n)$

$O(n) \leq \text{const} * n$ при больших n

Временная сложность

- Зависимость времени исполнения (или количества простых операций) от размера входных данных

Пространственная сложность

- Зависимость потребляемой памяти от размера входных данных

Жадный алгоритм

- На каждом этапе двигаться в сторону наиболее выгодного решения.
- [Acм.mipt.ru/judge](http://acm.mipt.ru/judge) задача 26
- Каждый раз стремимся к числу, делящемуся на 4 (1-3 обрабатываем отдельно)

Динамическое программирование

- Решение вычисляется из решения подзадач

Факториал

- $a[i] = i * a[i-1]$
- Временная сложность $O(n)$
- Пространственная сложность $O(n)$

1-мерная динамика

- Числа Фибоначчи

$\text{fib}[n] = \text{fib}[n-1] + \text{fib}[n-2];$

- 009 задача с EJudge

Можно уменьшить пространственную сложность до $O(1)$

1-мерная динамика

- 065 задача с EJudge
- Покупка билетов
- $queue[i] = \min(\begin{aligned} &a[i] + queue[i-1], \\ &b[i-1] + queue[i-2], \\ &c[i-2] + queue[i-3]) \end{aligned}$

Дискретная задача о рюкзаке

- EJudge, задача 059
- Есть список целочисленных длин композиций. Какую максимальную их сумму мы можем получить при условии, что она не превышает X ?
- Можем ли мы получить сумму длин Z ?
- Какие длины мы можем получить, добавив композицию длины N ?

2-мерная динамика

- LCS — наибольшая общая подпоследовательность(BDCABA, ABCBD)=BCB

	B	D	C	A	B	A
A	0	0	0	1	1	1
B	1	1	1	1	2	2
C	1	1	2	2	2	2
B	1	1	2	2	3	3
D	1	2	2	2	3	3

Сложность

- Пространственная $O(n*m)$
- Временная $O(n*m)$