

Алгоритмы

Лекция 1

ПРЕПОДАВАТЕЛЬ



Roman Sukhanov

Java / Python Developer in Machine Learning / Deep Learning

- 10+ years (2012 - today)
- Oil and gas industry, hotel industry



ВАЖНО:

- Камера должна быть включена на протяжении всего занятия.
- Если у Вас возник вопрос в процессе занятия, пожалуйста, поднимите руку и дождитесь, пока преподаватель закончит мысль и спросит Вас, также можно задать вопрос в чате или когда преподаватель скажет, что начался блок вопросов.
- Организационные вопросы по обучению решаются с кураторами, а не на тематических занятиях.
- Вести себя уважительно и этично по отношению к остальным участникам занятия.
- Во время занятия будут интерактивные задания, будьте готовы включить камеру или демонстрацию экрана по просьбе преподавателя.

ПЛАН ЗАНЯТИЯ

1. Повторение изученного
2. Вопросы по повторению
3. Основной блок
4. Вопросы по основному блоку
5. Задание для закрепления основного блока
6. Практическая работа
7. Оставшиеся вопросы

АЛГОРИТМЫ, ЗАЧЕМ?

Знание алгоритмов позволяет:

- Быть квалифицированным специалистом в своей профессии
- Иметь карьерный рост
- Уважение коллег
- Эффективно решать поставленные задачи

АЛГОРИТМЫ, ЗАЧЕМ?

Знание алгоритмов и тесно связанной с ними организации структуры данных необходимо для серьезной работы в любой отрасли информационных технологий.

Примеры:

- Протоколы маршрутизации в коммуникационных сетях используют классические алгоритмы поиска кратчайшего пути.
- Шифрование с открытым ключом опирается на эффективные теоретико-числовые алгоритмы.
- Компьютерная графика задействует вычислительные примитивы, которые предоставляют геометрические алгоритмы.
- Индексация в базах данных опирается на структуры данных сбалансированных деревьев поиска.
- Алгоритмы применяют для парсинга данных, фильтрации дубликатов, отрисовки динамических списков, хранения и вывода оповещений для пользователя.



TEL-RAN
by Starta Institute

1

ПОВТОРЕНИЕ ИЗУЧЕННОГО

Вспомним:

- Массивы
- Одномерный массив
- Максимальный и минимальный индекс в массиве

Номера (индексы) элементов массива

0 1 2 3 4 5 6 7

Arr

5	8	10	12	-9	10	11	-3
---	---	----	----	----	----	----	----

Название массива

Элементы массива

Экспресс-опрос

- **Вопрос 1.**

Как в памяти хранятся массивы?

- **Вопрос 2.**

Какого типа данные, могут храниться в массивах ?



Вспомним:

Алгоритм: Сортировка пузырьком (Bubble sorting)

Описание: последовательно сравнивать значения соседних элементов и менять местами элементы, если предыдущий больше последующего.

Таким образом элементы с большим значением оказываются в конце списка, а с меньшим в начале.



Экспресс-опрос

- **Вопрос 1.**

Если искомый элемент всегда лежит в конце списка, будет ли эффективным алгоритм линейного поиска, для нахождения элемента?

Объясните почему “да” или “нет”.

- **Вопрос 2.**

Как вы думаете из сортировки “пузырьком” можно ли сделать сортировку “камнем”?



2

ВОПРОСЫ ПО ПОВТОРЕНИЮ

Введение

- Что такое алгоритмы
- Характеристики алгоритмов
- Свойства алгоритмов
- Типы алгоритмов
- Как создавать алгоритмы
- Способы описания алгоритмов
- Элементы блок-схем
- Синтаксис псевдо-кода
- Эффективность алгоритма
- Преимущества и недостатки алгоритмов
- Примеры



3

ОСНОВНОЙ БЛОК

Что такое алгоритмы

- Алгоритм означает набор правил, которым необходимо следовать при вычислениях или других операциях по решению задач.
- Алгоритм относится к последовательности конечных шагов для решения конкретной проблемы.
- Алгоритмы могут быть простыми и сложными в зависимости от того, чего вы хотите достичь.
- Алгоритмизация – процесс разработки алгоритма для решения какой-либо задачи



Характеристики алгоритмов

Ясный и не двусмысленный	Четко определенные входные данные	Четко определенные результаты	Конечный	Выполнимый	Независимый от языка
Каждый его шаг должен быть ясен во всех аспектах и должен вести только к одному смыслу.	Если алгоритм говорит принимать входные данные, это должны быть четко определенные входные данные.	Алгоритм должен четко определять, какой результат будет получен, и он также должен быть четко определен.	Алгоритм должен быть конечным, т.е. он должен завершаться через конечное время.	Алгоритм должен быть простым, универсальным и практичным, чтобы его можно было выполнить с доступными ресурсами.	Алгоритм должен быть независимым от языка, т. е. это должны быть простые инструкции, которые могут быть реализованы на любом языке, и при этом вывод будет таким же, как и ожидалось.

Свойства алгоритмов

- Алгоритм должен завершиться через конечное время.
- Алгоритм должен принимать ноль или более входных данных.
- Алгоритм должен давать один и тот же результат для одного и того же входного случая.
- Каждый шаг в алгоритме должен быть эффективным.



Экспресс-опрос

- **Вопрос 1.**

Как вы поняли, что такое алгоритм?

- **Вопрос 2.**

Какие характеристики алгоритмов вы запомнили?

- **Вопрос 3.**

Может ли результат одного и того же входного случая быть разным?



Типы алгоритмов

- Алгоритм грубой силы.
- Рекурсивный алгоритм.
- Алгоритм поиска с возвратом.
- Алгоритм сортировки.
- Алгоритм хеширования.
- Алгоритм «разделяй и властвуй».
- Жадный алгоритм.
- Алгоритм динамического программирования.
- Рандомизированный алгоритм.



Как создавать алгоритмы

- Проблема, которая должна быть решена с помощью этого алгоритма, т.е. четкое определение проблемы.
- При решении проблемы необходимо учитывать все ограничения.
- Входные данные, которые необходимо принять для решения этой проблемы.
- Ожидаемый результат после решения проблемы.
- Решение этой проблемы находится в рамках заданных ограничений.



Способы описания алгоритмов

Существует три основных способа описания алгоритма:

Текстовый

Графический

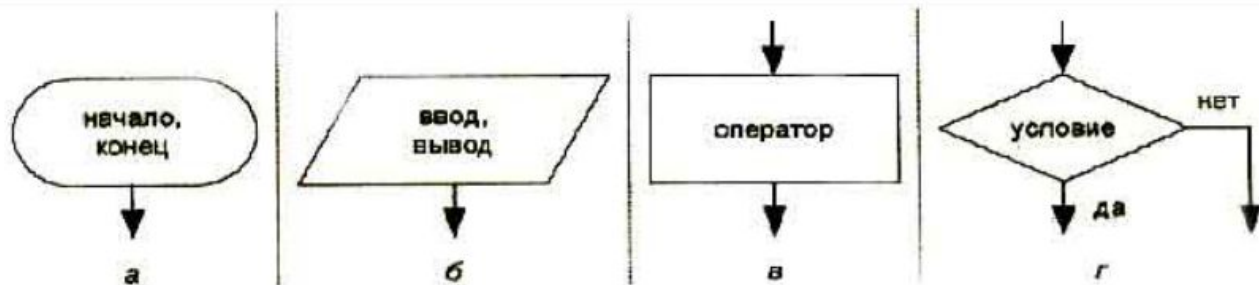
Алгоритмический

Расписать шаги
алгоритма
последовательно в
тексте

Изобразить
графически в виде
блок-схем

Написать псевдокод

Основные элементы блок-схем



а — начало (конец) алгоритма

б — блок ввода/вывода

в — операционный блок

г — логический (условный) блок

д — цикл с параметром (для параметра цикла указывается его начальное и конечное значение, шаг равен единице)



Синтаксис псевдокода

START / END - начало / конец алгоритма

WRITE / READ - ввод / вывод данных

IF THEN ELSE - выбор

FOR / WHILE / REPEAT - итерация(Циклы)

Экспресс-опрос

- **Вопрос 1.**

За что отвечает этот элемент блок-схемы?



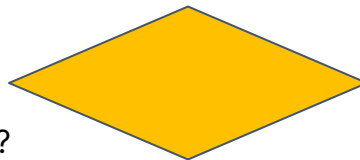
- **Вопрос 2.**

Для чего используется данный элемент блок-схемы?



- **Вопрос 3.**

Когда необходимо использовать следующий элемент блок-схемы?



Эффективность алгоритма

Чтобы стандартный алгоритм был хорошим, он должен быть эффективным. Следовательно, эффективность алгоритма должна проверяться и поддерживаться.

- Фактор времени : время измеряется путем подсчета количества ключевых операций.
- Фактор пространства : пространство измеряется путем подсчета максимального объема памяти, требуемого алгоритмом.



Преимущества алгоритмов

- Алгоритм легко понять.
- Алгоритм — это пошаговое представление решения данной задачи.
- В алгоритме проблема разбивается на более мелкие части или шаги, поэтому программисту легче преобразовать ее в настоящую программу.



Недостатки алгоритмов

- Написание алгоритма занимает много времени.
- Понимание сложной логики с помощью алгоритмов может быть очень трудным.
- Операторы ветвления и цикла трудно показать в алгоритме.



ПРИМЕРЫ

Словесный алгоритм

Алгоритм сравнения переменных:

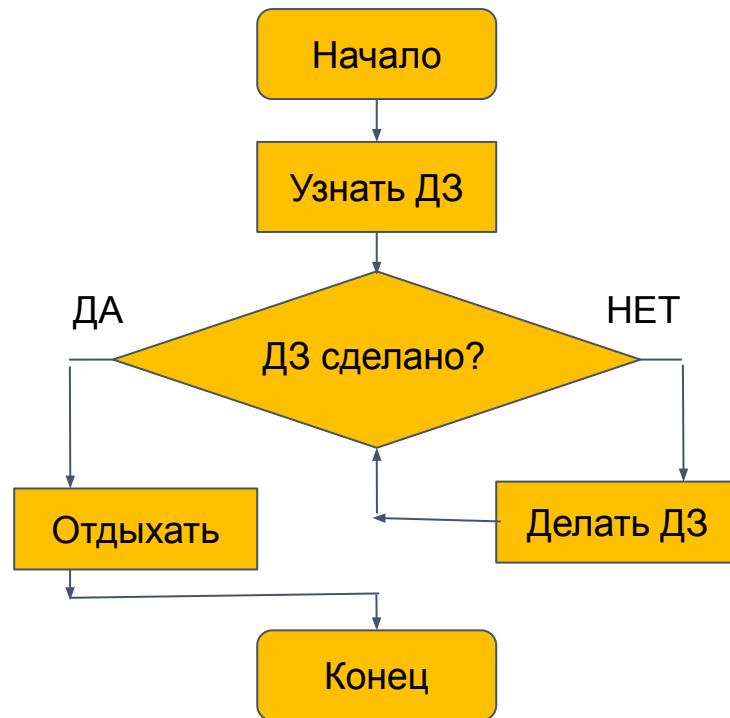
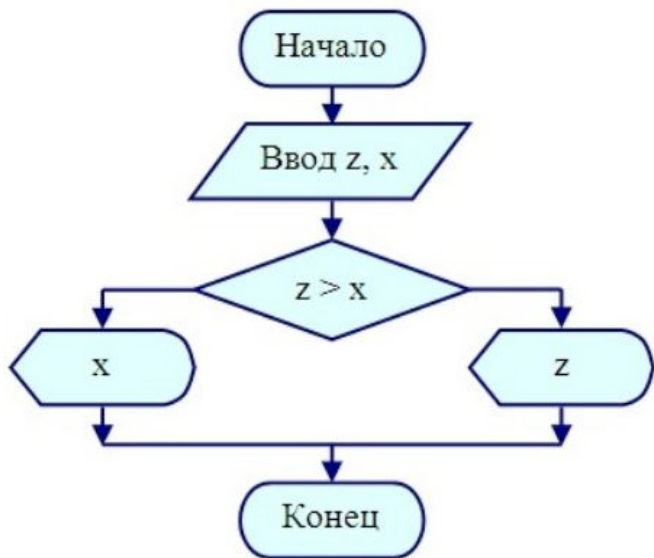
- Ввести z, x
- Если $z > x$, то выводим z
- Если $x > z$, то выводим x

Алгоритм выполнения домашнего задания:

- Узнать домашнее задание
- Выполнить домашнее задание
- Если домашнее задание выполнено, то отдыхай
- Если домашнее задание не выполнено, то выполняй домашнее задание



Графический алгоритм



Алгоритмический язык

Алгоритм сравнения переменных:

START

Number input: Z, X

IF $Z > X$ THEN output Z

ELSE output X

END

Алгоритм выполнения домашнего задания:

START

READ determine the task

WHILE (task is done)

doing task

END





TEL-RAN
by Starta Institute

4

ВОПРОСЫ ПО ПОДБЛОКУ



TEL-RAN
by Starta Institute

5

ЗАДАНИЕ ДЛЯ ЗАКРЕПЛЕНИЯ