Московский государственный университет им. М. В. Ломоносова
Факультет вычислительной математики и кибернетики

Алгоритмы и алгоритмические языки
Лекция 1

4 сентября 2019 г.

Курс «Алгоритмы и алгоритмические языки»

Лектор: Андрей Андреевич Белеванцев,
ИСП РАН им. В. П. Иванникова / кафедра СП ВМК МГУ

Лекции 2 раза в неделю: среда/суббота, 8.45
Практические и лабораторные занятия 2 раза в неделю

Структура курса:

- Элементы теории алгоритмов
- Язык Си
- Алгоритмы и структуры данных

В конце курса зачет с оценкой и письменный экзамен

Курс «Алгоритмы и алгоритмические языки»

Сайт курса: `http://algcourse.cs.msu.su/`

- Новости и объявления
- Материалы лекций
- Рекомендуемая литература
- Вопросы к экзамену
- Среда разработки программ и опции компилятора
- Стиль кодирования

Практические и лабораторные занятия

Контрольные и коллоквиумы (по лекциям — 2, по семинарам — 3)

Критерии оценки: 10% – 15% – 75% (коллоквиумы/экзамен)

Рекомендуемая литература

По алгоритмам и машинам Тьюринга

1. Г. Эббинхауз, К. Якобс, Ф. Манн, Г. Хермес. Машины Тьюринга и рекурсивные функции. «Мир», М.– 1972
2. Л. С. Корухова, М. Р. Шура-Бура. Введение в алгоритмы (учебное пособие для студентов I курса). PDF на сайте курса

По языку Си

1. Б. Керниган, Д. Ритчи. Язык программирования Си. Издание 2-е, «Вильямс» – 2013
2. Стандарт языка Си C99 + TC1,2,3. `http://www.open-std.org/JTC1/SC22/WG14/www/docs/n1256.pdf`
3. Stephen Prata. C Primer Plus. Fifth Edition. Sams Publishing 2004. `http://www.9wy.net/onlinebook/CPrimerPlus5/main.html`

По алгоритмам и структурам данных

1. Т. Кормен, Ч. Лейзерсон, Р. Ривест, К. Штайн. Алгоритмы. Построение и анализ. Издание 2-е, «Вильямс» – 2011
2. Harry R. Lewis, Larry Denenberg. Data Structures and Their Algorithms. HarperCollins, 1991.

Методические пособия

1. А. А. Белеванцев, С. С. Гайсарян, Л. С. Корухова, Е. А. Кузьменкова, В. С. Махнычев. Семинары по курсу «Алгоритмы и алгоритмические языки» (учебно-методическое пособие для студентов 1 курса). М., 2012: Изд. отдел ф-та ВМК МГУ имени М.В. Ломоносова.
2. А. А. Белеванцев, С. С. Гайсарян, В. П. Иванников, Л. С. Корухова, В. А. Падарян. Задачи экзаменов по вводному курсу программирования (учебно-методическое пособие). М., 2012: Изд. отдел ф-та ВМК МГУ имени М.В. Ломоносова.
3. К. А. Батузов, А. А. Белеванцев, Р. А. Жуйков, А. О. Кудрявцев, В. А. Падарян, М. А. Соловьев. Практические задачи по вводному курсу программирования (методическое пособие). М., 2012: Изд. отдел ф-та ВМК МГУ имени М.В. Ломоносова.
4. О. В. Сенюкова. Сбалансированные деревья поиска: учебно-методическое пособие. М.: Издательский отдел факультета ВМиК МГУ имени М.В. Ломоносова; МАКС Пресс, 2014. Доступно на сайте курса

Планируется: А. А. Белеванцев, С. С. Гайсарян, Л. С. Корухова, Е. А. Кузьменкова. Элементы теории алгоритмов. Учебно-методическое пособие для студентов 1 курса. М.: Издательский отдел факультета ВМиК МГУ имени М.В. Ломоносова; МАКС Пресс, 2019.

Неформальное (интуитивное) определение алгоритма

Под *алгоритмом* (или эффективной процедурой) в математике понимают *точное предписание*, задающее *вычислительный процесс*, ведущий от *начальных данных*, которые могут варьироваться, к *искомому результату*. Алгоритм должен обладать следующими свойствами:

- **Конечность** (результативность). Алгоритм должен заканчиваться за конечное (хотя и не ограниченное сверху) число шагов.
- **Определенность** (детерминированность). Каждый шаг алгоритма и переход от шага к шагу должны быть точно определены и каждое применение алгоритма к одним и тем же исходным данным должно приводить к одинаковому результату.

Неформальное (интуитивное) определение алгоритма

Под *алгоритмом* (или эффективной процедурой) в математике понимают *точное предписание*, задающее *вычислительный процесс*, ведущий от *начальных данных*, которые могут варьироваться, к *искомому результату*. Алгоритм должен обладать следующими свойствами:

- **Простота и понятность**. Каждый шаг алгоритма должен быть четко и ясно определен, чтобы выполнение алгоритма можно было «поручить» *любому* исполнителю (человеку или механическому устройству).
- **Массовость**. Алгоритм задает процесс вычисления для множества исходных данных (чисел, строк букв и т.п.), он представляет общий метод решения класса задач.

Неформальное (интуитивное) определение алгоритма

Пример. *Алгоритм Евклида* нахождения наибольшего общего делителя двух целых положительных чисел a и b НОД(a, b).

Даны два целых числа a и b, найти НОД(a, b).

Выполнить следующие шаги:

1. Если $a < b$, то поменять их местами.
2. Разделить нацело a на b; получить остаток r.
3. Если $r = 0$, то НОД$(a, b) = b$.
4. Если $r \neq 0$, заменить: a на b, b на r и вернуться к шагу 2.

Почему необходимо формальное определение алгоритма

Не имея такого определения, **невозможно** доказать, что задача алгоритмически неразрешима, т.е. алгоритм ее решения никогда не удастся построить.

Тезис Тьюринга–Чёрча. Для любой интуитивно вычислимой функции существует вычисляющая её значения машина Тьюринга.

Тезис Тьюринга–Чёрча невозможно строго доказать или опровергнуть, так как он устанавливает эквивалентность между строго формализованным понятием частично вычислимой функции и неформальным понятием вычислимости.

Формализация понятия алгоритма. Алфавиты и отображения

Алфавит — это конечное множество A_p элементов a_i: $A_p = \{a_1, a_2, \ldots, a_p\}$.
Элементы алфавита A_p называются *символами*.

Последовательность из m символов алфавита A_p называется *словом* длины m над алфавитом A_p: $a_{i_1} a_{i_2} \ldots a_{i_m}$

Слово длины 0 называется *пустым* словом и обозначается ε.

Множество всех слов над алфавитом A_p:

$$A_p^* = \{\varepsilon\} \cup A_p \cup A_p^2 \cup \ldots \cup A_p^m \cup \ldots = \bigcup_{m=0}^{\infty} A_p^m.$$

Длину слова $w \in A_p^*$ будем обозначать $|w|$,
в частности, для пустого слова $|\varepsilon| = 0$.

Формализация понятия алгоритма. Кодирование

Утверждение. Для любой пары алфавитов A и B можно выполнить кодирование алфавита A с помощью алфавита B и обратно, возможно, с применением дополнительно служебного символа $\imath$ («конец кода символа»).

Следствие. Кодирование позволяет ограничиться одним алфавитом.
Обычно рассматриваются A_1 или A_2.

Формализация понятия алгоритма. Обработка информации.

Задача обработки информации — это задача построения частичного отображения (функции) $F : A^* \to A^*$.

Утверждение. Существует взаимно-однозначное отображение $\# : A^* \leftrightarrow \mathbb{N}_0$, где $\mathbb{N}_0$ — множество целых неотрицательных чисел, которое любому слову $w \in A^*$ ставит в соответствие его номер $n \in \mathbb{N}_0$. (Это отображение $\#$ и называется нумерацией.)

Формализация понятия алгоритма. Обработка информации

$$\begin{array}{ccc} A^* & \xrightarrow{F} & A^* \\ \Big\downarrow{\scriptstyle \#} & & \Big\uparrow{\scriptstyle \#^{-1}} \\ \mathbb{N}_0 & \xrightarrow{f} & \mathbb{N}_0 \end{array}$$

Таким образом:

1. каждый алгоритм $F : A^* \to A^*$ определяет частично вычислимую функцию $f : \mathbb{N}_0 \to \mathbb{N}_0$;
2. каждая частично вычислимая функция $f : \mathbb{N}_0 \to \mathbb{N}_0$ определяет алгоритм $F : A^* \to A^*$.

Машина Тьюринга (МТ). Вычислимость по Тьюрингу

Машина-автомат: предъявляется любое исходное слово $w \in A^*$, а в результате обработки получается слово $v = F(w)$.

Каждая частичная функция F, для которой можно построить МТ, называется *вычислимой по Тьюрингу*.

Машина Тьюринга. Обозначения

Алфавит состояний $Q = \{q_0, q_1, q_2, \ldots, q_n\}$

Рабочий алфавит $S = A \cup A^{'}$

- A — алфавит *входных* символов
- $A^{'}$ — алфавит *вспомогательных* символов (*маркеров*)

Лента, размеченная на ячейки (пустая ячейка — Λ)

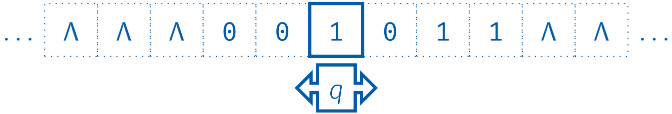
Управляющая головка (УГ)

Рабочая ячейка (РЯ)

Начальное состояние q_0, состояние останова q_s

Начальные данные — слова из A^*

Машина Тьюринга. Конфигурация



Конфигурация МТ: $\langle n, F, q \rangle$, где n — номер текущей рабочей ячейки, $F : \mathbb{Z} \to S$ — текущая запись на ленте, q — текущее состояние.

Позиция МТ: пара $\langle n, q \rangle$.

Такт работы МТ:
$\langle \textit{состояние}, \textit{символ} \rangle \to \langle \textit{состояние}, \textit{символ}, \textit{направление} \rangle$.