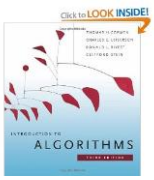
Data Structures and Algorithms CMPSC 465



LECTURES 1

Analysis of Algorithms

- Course information
- What are algorithms?
- Why study them?

Paul Medvedev

These slides are based on slides of S. Raskhodnikova, A. Smith, K. Wayne, E. Demaine and C. Leiserson

Course information

Instructor: Prof. Medvedev (pzm11@psu.edu), offices at Wartik 517 and IST 343J.

Lectures: MWF 4:40–5:30pm, 026 Hosler. Recitations are held on Mondays.

Webpage: `https://goo.gl/B75pUO`

Discussion: We will be using Piazza for class discussion. Please use this forum, instead of email or Angel, for asking questions about the course material. For questions related to your grades or other personal matters, please use Angel mail. Our Piazza page is found at: `https://piazza.com/class/icb646n9hlm7oz?cid=4#`

Communication: Course email from the instructors, including important announcements, will be sent through Angel and through Piazza. Please check Angel often or setup mail forwarding.

Prerequisites CMPSC 122 (basic data structures, recursion), CMPSC 360/CSE 260/MATH 311W (mathematical induction, graphs, trees, equivalence relations, basic probability) or equivalent.

Textbook: Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Introduction to Algorithms. 3rd ed. Cambridge, MA: MIT Press. ISBN: 0262032937.

Course information

Grading: The grade will be calculated as follows:

30%	Homework
15%	Midterm 1
15%	Midterm 2
15%	Midterm 3
25%	Final exam

Course information

Homework: There will be three different types of homeworks assigned in the class: programming assignments, Angel assessments, and open-ended questions. For programming assignments and Angel assessments, all work must be independent (see Collaboration policy below). For open-ended questions,

- You may work either alone or in teams of two. If working in pairs, you should hand-in a single assignment, with both of your names on it.
- All assignments must be typed (Word is recommended). No handwritten work will be accepted or graded, with the exception of drawn figures.

Late homework will not be accepted, but you are allowed to drop one lowest homework grade at the end of the semester. If you cannot hand in the homework due to extenuating circumstances, you must make arrangements with the instructor at least 24 hours in advance.

You should be as clear and concise as possible in your write-up of solutions. Understandability of your answer is as desirable as correctness. A simple, direct analysis is worth more points than a convoluted one. For example, points might be subtracted for solutions that are too long.

Course information

Collaboration Policy You are not allowed to discuss any aspects of the homework solutions problem with anyone. You may, however, discuss the question requirements and general approaches. Finding answers to problems on the Web or from other outside sources (these include anyone not enrolled in the class) is strictly forbidden. No collaboration whatsoever is permitted on exams.

It is a violation of this policy to submit a problem solution that you cannot orally explain to an instructor or TA. Violations of this policy will be dealt with according to University regulations.

Etymology of "Algorithm"

Abu Abdullah Muhammad ibn Musa al-Khwarizmi (c. 780 -- 850 AD)

- Persian astronomer and mathematician
- lived in Baghdad, father of algebra
- "On calculating with hindu numerals"
  a treatise in Arabic, 825
- "Agoritmi de numero Indorum"
  translation into Latin, 12th century
- author's name, mistaken for a plural noun, came to mean "calculation methods"

Algorithm Design and Analysis

Theoretical study of how to solve computational problems

- sorting a list of numbers
- finding a shortest route on a map
- scheduling the operations of an airport
- answering web search queries

(Generally: precisely defined set of **inputs** and, for each input, **acceptable outputs**)

Algorithms

- Definition: Finite set of unambiguous instructions for solving a problem.
  - An algorithm is **correct** if on all legitimate inputs, it outputs the right answer in a finite amount of time
- Can be expressed as
  - pseudocode
  - flow charts
  - text in a natural language (e.g. English)
  - computer code

Performance isn't everything

- Typical goal: Find most space- and time-efficient algorithm for given problem.
- What else is important?
  - modularity
  - correctness
  - maintainability
  - functionality
  - robustness
  - user-friendliness
  - programmer time
  - simplicity
  - extensibility
  - reliability

Performance isn't everything

- Typical goal: Find most space- and time-efficient algorithm for given problem.
- Even performance has many facets:
  - type of memory access
  - cache usage
  - network usage
  - parallelism
- This course: simple models, general skills

Data Structures

- **Data structures** are ways to store information for which there are **algorithms** for performing particular operations (retrieving/manipulating information), e.g.
  - linked lists
  - hash tables
  - arrays
  - trees
  - heaps

Course Objectives

- classical algorithms and data structures
- analysis of algorithms
- standard design techniques

Why study algorithms and DS?

- It will be on your interview
- standard set of algorithms and design techniques
- analyzing correctness and resource usage
- many successful companies are due to innovative algorithms (e.g. Google, Amazon)
- feasibility (what can and cannot be done)
  - halting problem, NP-completeness
- computation is fundamental to understanding the world
  - cells, brains, social networks, physical systems all can be viewed as computational devices

The problem of sorting

Input: sequence $\langle a_1, a_2, \ldots, a_n \rangle$ of numbers.

Output: permutation $\langle a'_1, a'_2, \ldots, a'_n \rangle$ such that $a'_1 \le a'_2 \le \cdots \le a'_n$.

Example:

Input: 8 2 4 9 3 6

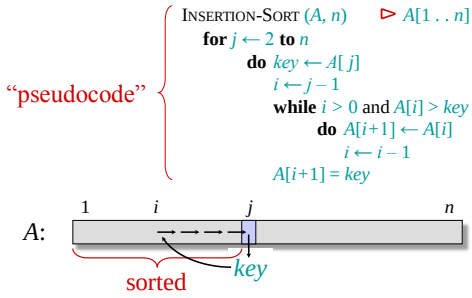
Output: 2 3 4 6 8 9

Insertion Sort

"pseudocode"

```
INSERTION-SORT (A, n)      ▷ A[1 . . n]
  for j ← 2 to n
       do key ← A[ j]
          i ← j – 1
          while i > 0 and A[i] > key
               do A[i+1] ← A[i]
                  i ← i – 1
          A[i+1] = key
```

Insertion Sort



Example of Insertion Sort