C110 Algorithms

Credits. 2

Prerequisites. None

Dependent Courses. C303, E005, E006, E008

Attributions. Core; C, L

Rationale, Outlook, Purpose, Objectives, and Goals. This is intended to be an introduction to algorithms for a non-computer-science graduate student. In principle, any programming language (`C`, `Python`, `Haskel`, `LISP`, etc.) can be used for illustrating algorithms, at the discretion of the instructor. However, `C` is highly recommended so as to give student a closer feel of computer system organization. Please see the sister course C109. This course is intended for

- familiarizing the student with the computer system organization and its use for problem solving;
- making student understand the need for formal algorithm development; and
- introducing basic types of algorithms, design techniques, data structures used for problem solving.

Syllabus.

1. Introduction to Algorithms. What is an algorithm, why do we need it? Introduction to fundamental algorithms like counting, sorting; algorithms for problem solving using digital computers, flow chart and pseudocode techniques. [5-6 hrs.]
2. Algorithms. Fundamental algorithms and techniques, data structures required (queue, FIFO, FILO, LIFO, LILO terminologies, stacks, link-lists, trees and graphs), logic, set theory, functions, basics of number theory and combinatorics (sequences-series, Sigma and PI notations for termwise summation, multiplication, probability, permutations, combinations), mathematical reasoning–including induction. [10-12 hrs.]
3. Recursion. Need, advantages, disadvantages. Recurrence analysis. Introduction to recurrence equations and their solution techniques (substitution method, tree recursion method, master method). Proof of the master method for solving recurrences. Demonstration of the applicability of master theorem to a few algorithms and their analysis using recurrence equations. Example algorithms: binary search, powering a number, Strassen'S matrix multiplication, etc. [10-12 hrs.]
4. Types of Algorithms and Their Analysis. Theta and big-theta notation, θ and Θ notations; comparison of algorithms, notions of space and time efficiency; as an illustrative example, comparison of quick-sort algorithm with other sorting algorithms can be demonstrated. [5 hrs.]

Suggested Texts/References.

- V. Rajaraman, T. Radhakrishnan, *An Introduction to Digital Computer Design.* PHI, 2007.
- T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, *Introduction to Algorithms.* PHI Learning, 2009.
- D. E. Knuth, *The Art of Computer Programming, Vol. 1.* Addison Wesley, 2011.

- A. V. Aho, J. E. Hopcroft, J. D. Ullman, *Design and Analysis of Algorithms.* Pearson Education, 2011.
- E. Horowitz, S. Sahni, *Fundamentals of Computer Algorithms.* Universities Press, 2008.

Notes on Pedagogy.

Contributor/s. Bhalchandra Gore (`http://cms.unipune.ac.in/~bwgore`)