COURSE DESCRIPTION

Dept., Number	CS430	Course Title	Introduction to Algorithms
Semester hours	3	Course Coordinator	Dr. Sanjiv Kapoor, Professor

Current Catalog Description

Introduction to the design, behavior, and analysis of computer algorithms. Searching, sorting, and combinatorial algorithms are emphasized. Worst case and average bounds on time and space usage. 3-0-3 (T) (C)

Textbook

Cormen, Leiserson and Rivest, *Introduction to Algorithms*, MIT Press/McGraw Hill, 2001

References

A. Aho, J. Hopcroft and J.D. Ullman, *Design and Analysis of Algorithms*, Addison-Wesley.

Course Outcomes

Students should be able to:

- Use big O, omega, and theta notation to give asymptotic upper, lower, and tight bounds on time and space complexity of algorithms.
- Determine the time complexity of simple algorithms, deduce the recurrence relations that describe the time complexity of recursively defined algorithms, and solve simple recurrence relations.
- Design algorithms using the brute-force, greedy, dynamic programming, divide-and-conquer, branch and bound strategies.
- Design algorithms using at least one other algorithmic strategy from the list of topics for this unit.
- Use and implement the fundamental abstract data types -- specifically including hash tables, binary search trees, and graphs -- necessary to solve algorithmic problems efficiently.
- Solve problems using techniques learned in the design of sequential search, binary search, O(N log N) sorting algorithms, and fundamental graph algorithms, including depth-first and breadth-first search, single-source and all-pairs shortest paths, and at least one minimum spanning tree algorithm.
- Demonstrate the following abilities: to evaluate algorithms, to select from a range of possible options, to provide justification for that selection, and to implement the algorithm in simple programming contexts.
- Communicate theoretical and experimental analyses of a set of algorithms (i.e. sorting) in

a lab report format.

Relationship between Course Outcomes and Program Outcomes

The following Program Outcomes are supported by the above Course Outcomes:

- a. An ability to apply knowledge of computing and mathematics appropriate to the discipline
- b. An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution
- c. An ability to design, implement and evaluate a computer-based system, process, component, or program to meet desired needs
- f. An ability to communicate effectively with a range of audiences
- h. Recognition of the need for, and an ability to engage in, continuing professional development
- i. An ability to use current techniques, skills, and tools necessary for computing practices.
- j. An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices
- l. Be prepared to enter a top-ranked graduate program in Computer Science.

Prerequisites by Topic

CS115/CS116 - Object-Oriented Programming: functions, pointers, recursion, classes

CS330 - Discrete Mathematics: sets, functions, counting, proofs

CS331 - Data Structures: abstract data types, lists, stacks, queues, trees

Major Topics Covered in the Course

1. Introduction to Algorithm Design, Complexity analysis including elementary tools like O-Notations, Recurrence Relations	4.5 hours
2. Introduction to Backtracking and Branch and Bound	3 hours
3. Introduction to Dynamic Programming	4.5 hours
4. Divide and Conquer and Greedy Methods (using Traveling Salesman Problem, Knapsack Problem and Optimum Triangulation of Convex Polygons)	3 hours
5. Sorting Methods - Quicksort, Mergesort, Heaps and Heapsort, Lower bound on sorting	4.5 hours
6. Searching I - Hash Functions and Hashing, Union Find	3 hours
7. Searching II-- Binary Search Trees, Balanced Binary Search Trees (AVL Trees, 2-3 trees/ Red-Black trees)	6 hours
8. Graph Algorithms I - Depth First Search, Breadth First search, Bi-connectivity, Topological Sort	4.5 hours
9. Graph Algorithms II - Minimum Spanning Trees, Shortest Paths	4.5 hours

10. String Matching	1.5 hours
11. NP-Complete Problems	3 hours
12. Parallel Model of Computing - Example Sorting (*)	1.5 hours
Midterm Exam	1.5 hours
Final Exam	-
(*) Optional Topics	45 hours

Assessment Plan for the Course

End of every semester Course Objective Assessments by CS department. End of semester Course Evaluations by IIT. Reviewed every Spring semester by CS Undergraduate Studies Committee for possible updates in the following Fall. Once every 4-5 years a detailed review of all materials for the course is made by the CS Undergraduate Studies Committee.

How Data in the Course is Used to Assess Program Outcomes (unless adequately covered already in the assessment discussion under Criterion 4)

See the assessment discussion under Criterion 4

For a computer science program

Estimate Curriculum Category Content (Semester hours)

Area	Core	Advanced	Area	Core	Advanced
Algorithms		2	Software design		
Data structures		1	Concepts of programming languages		