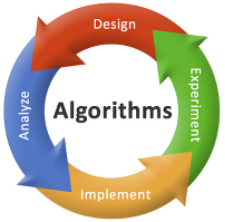
Introduction

Course: Algorithms



Faculty: Dr. Rajendra Prasath

Autumn 2018

About this Course

This course covers the essential aspects that every serious programmer needs to know about algorithms, **design principles and their analysis**, with emphasis on **real-time implementations** and **scalable application development**

What do we learn?

- Problem – Solving
- How to approach the given problem?
- Designing Solutions
  - To Solve Interesting Problems !!
  - Devise Efficient Methods !!
  - Logical and Precise steps
  - Computable Representations

  - Simple Algorithms to Advanced algorithms
  - Scalable problem solving approaches

Two Steps to Remember

- Data Structures
  - The choice of Data Structures
  - Built-in Data Structures (Primitive)
  - User Defined Data Structures (Abstract)

- Computational Efficiency
  - Time Complexity
  - Space Complexity

  - Problem / Solution Specific Constraints
  - Best Practices / Efficient Approaches

Look at the Table

- Problems?
- Ways of looking at different problems
- How to solve them?
- Naïve approach or Efficient approach?

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

A Simple Problem

- Consider Swapping of two integers
  - Input: x = 5, y = 7
  - Output: x = 7, y = 5

  - A simple Solution:
    - Use of temporary variable to hold the intermediate value
    - Any Other Solution?
      - Bitwise
      - Add / Subtract with no intermediate value

- Explore Complexity Complexity
  - Time and Space needed

Time Complexity

- **Computation of TIME**
  - Consider a unique list of 10 numbers (unsorted):
    5, 8, 21, 43, 35, 17, 94, 68, 54, 81
    (Traversal must be left to right)
- **Task:**
  - Find any one (??) number in the list

- **Best Case**
  - The minimum (at least) time taken for solving the problem
  - Finding the number: 5 in the list
- **Average Case**
  - Average time taken to solve the problem
- **Worst Case**
  - The maximum (at most) time taken for any input size
  - Finding the number 81 in the list

Space Complexity

- A measure of the amount of working storage an algorithm needs.
- This means how much memory, in the worst case, is needed at any point by the underlying algorithm?
- We always focus on how the space needs grow, in big-Oh terms, as the size N of the input problem grows.

- Explore different aspects of time and space complexities

Course Content

- Course is divided into several modules:

Module: M1 – M7

- Covers Basic Algorithms to Advanced Algorithms (at least one example problem with detailed analysis)

- Course is supposed to be an interactive course and class performance bonus would be given to students who solve the given set of problems efficiently

➔ Course Content follows ..

M1: Fundamentals

- Introduction
- The Model of Computation
  - Mathematics of Algorithms
  - Computational Complexity of algorithms
  - Algorithms Pattern / design
  - Algorithmic thinking
  - Floating point computations
  - Effects on the Choice of Data Structures
  - Handling Recurrence Relations
  - Best practices in Problem – Solving approaches
- Max: 3 classes to cover the above

M2: Sorting Algorithms

- Sorting Algorithms
  - Overview and Essence of Sorting Algorithms
  - Insertion Sort
  - Merge Sort
  - Quick Sort
  - Selection Sort
  - Heap Sort
  - Bucket Sort
  - Criteria for Choosing Sorting Algorithms
  - Take Home Assignments
- Max: 3 - 4 classes

M3: Searching Algorithms

- Searching algorithms
  - Sequential Search
  - Binary Search
  - Hash based Search
  - Binary Tree Search
  - Scalable Searching Algorithms
  - Efficient approaches in Searching

  - Class room assignments
  - Take Home assignments
- Max: 4 Classes

M4: Graph Algorithms

- Overview of Graph Algorithms
  - Graph Construction
  - Depth-First Search
  - Breadth First Search
  - Single Source Shortest Path Algorithm
  - All Pairs Shortest Path Algorithm
  - Minimum Spanning Tree Algorithms
  - Overlay Graphs
  - Practical Examples
  - Scalable Graph Examples: Small World Networks
- Max: 6 Classes

M5: Network Flows Algorithms

- Overview of Network Flow Algorithms
  - Maximum Flow Algorithm
  - Bipartite Matching
  - Reflections on Augmenting Paths
  - Minimum Cost Flow
  - Transportation Problems
  - Assignment Problems
  - Linear Programming Problems
  - Take Home Assignments

- Max: 4 Classes

M6: Geometric Algorithms

- Overview
- Geometric Algorithms
  - Convex Hull Scan
  - Line Sweep
  - Nearest Neighbor Queries
  - Range Queries
  - Applications in various domains
  - Practice Problem solving
  - Take Home Assignments

- Max: 3 – 4 classes

M7: Advances in Algorithms

- Overview
- Advances in Algorithms
  - Approximation Algorithms
  - Offline and Online Algorithms
  - AnyTime Algorithms
  - Parallel Algorithms
  - Randomized Algorithms
  - One solved problem in each category
  - Take Home Assignments
  - Real-Time Applications
- Max: 4 – 5 Classes

Take Home Assignments

- Solve a set of problems every week
- Must be solved by individuals
- Must be finished before Every Monday or the deadline specified for that set of problems
- All Assignments are COMPULSARY

- Total Weightage: 20%;

- **NOTE:** if you fail to your solution, you will get "0"

- Solutions would be cross checked !!
- Solutions submitted after the deadline will not be considered for evaluation
- Submission Procedure would be given.

Examinations

- Mid Semester – 1: 10 Marks
- Mid Semester – 2: 15 Marks
- End Semester : 25 Marks

- Total Weightage (100) = Take Home Assignments (20) + Exams (50) + Best Solutions (10) + Class Performance (20)

- Academic Code of Conduct
  - Explore PENALTIES

Penalties

- Every Student is expected to strictly follow a fair Academic Code of Conduct to avoid severe penalties

- Penalties would be heavy for those who involve in:
  - **Copy and Pasting** the code
  - **Plagiarism** (copied from your neighbor or friend – in this case, both will get "0" marks for that specific take home assignments)
  - If the candidate is **unable to explain his own solution**, it would be considered as a "copied case" !!
  - **Any other unfair means** of completing the assignments

Help among Yourselves?

- **Perspective Students** (having CGPA above 8.5 and above)
- **Promising Students** (having CGPA above 6.5 and less than 8.5)

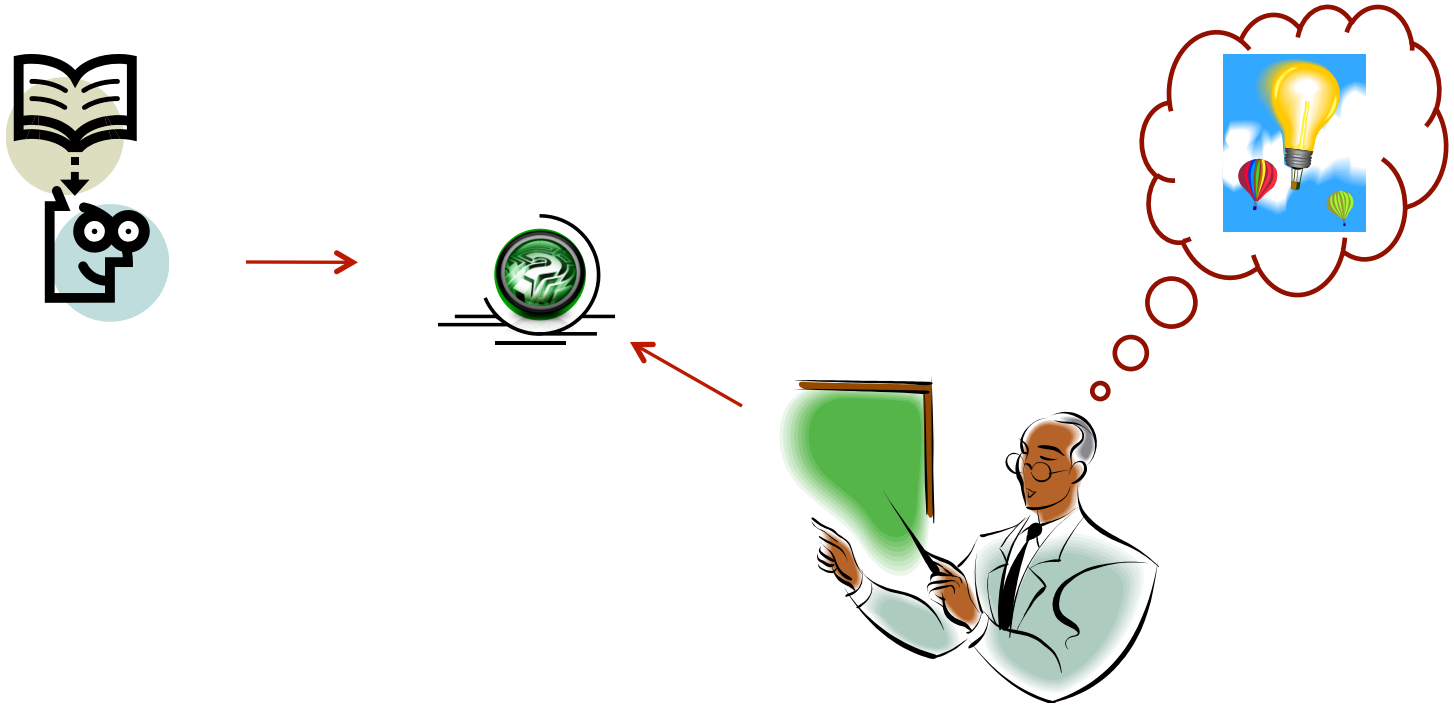
- **Needy Students** (having CGPA less than 6.5)
  - Can the above group help these students? (Your work will also be rewarded)

- You may grow a culture of **collaborative learning** by helping the needy students

Assistance

- You may post your questions to me at any time
- You may meet me in person on available time or with an appointment
- TA s would assist you to clear your doubts.
- You may leave me an email any time (email is the best way to reach me faster)

Thanks …



… Questions ???