# CS- 403 ( Analysis of Algorithms )

## Course Description

| | | | |
|---|---|---|---|
| **Course code number** | CS-403 | **Course Title** | Analysis of Algorithms |
| **Semester hours** | 3 hours | **No. of Credit Units** | 3 |
| **Prerequisite** | CST-103, CST-203 | **Course Coordinator** | Dr. Kyar Nyo Aye |

## Course Aims

The aim of this course is to learn how to develop efficient algorithms for simple computational tasks and reasoning about the correctness of them. Algorithm design and analysis provide the theoretical backbone of computer science and are a must in the daily work of the successful programmer. Addition, this course is to provide a solid background in the design and analysis of the major classes of algorithms.

## Learning Outcomes

Students who have completed this course should:

- be able to demonstrate how the worst-case time complexity of an algorithm is defined
- be able to compare the efficiency of algorithms using asymptotic complexity
- be able to design efficient algorithms using standard algorithm design techniques
- be able to demonstrate a number of standard algorithms for problems in fundamental areas in computer science and engineering such as sorting, and searching.
- be able to solve problems which are algorithm based by using various design techniques.

## Course Contents

This course provides students with a broad foundation in computer science.

1. Models of Computation: Algorithm and Their Complexity, Random Access Machine, Computational Complexity of RAM programs, A Stored Program Model, Abstractions of the RAM, A primitive model of computation: the Turing machine
2. Design of Efficient Algorithms: Recursion, Divide-and-conquer, Balancing, Dynamic programming
3. Sorting and Order Statistics: The sorting problem, Radix sorting, Sorting by comparisons, Heap sort, Quick Sort, Order statistics

## Reference Materials

1. Design and Analysis of Computer Algorithms by Alfred V. Aho, John E. Hopcroft & Jeffery D. Ullman

**Course Organization**

The expected learning outcomes for the course will be assessed through six forms of activity:

1. Attending the lectures
2. Preparing for and participating in the recitations.
3. Assignments
4. Reading the text
5. Quiz
6. Exams

## Assessment

| | |
|---|---|
| Exam | 50% |
| Tutorials/Test | 10% |
| Class participation | 10% |
| Assignment | 10% |
| Quiz | 10% |
| Moodle Test | 10% |

**45 periods for 15weeks** (50 minutes for 1 period)

| No | Chapter | Page | Period | Detail Lecturer Plan |
|---|---|---|---|---|
| **I** | **Chapter (1)**<br>**Models of Computation** | 2 to 41 | **25** | Lectures + Tutorials + Exercise |
| 1. | 1.1 Algorithm and Their Complexity | 2-5 | 1 | Detail Explain why we need to analyze algorithm |
| 2. | 1.2 Random Access Machine RAM Instruction & Meaning | 5-8 | 2 | Detail Explain RAM model<br>Explain RAM instruction with Table 1.4 and Table 1.5 |
| 3. | 1.3 Teach writing ALGOL program<br>Teach how to write RAM program | 9-11 | 2 | Detail Explain about Fig 1.6 and 1.7<br>Detail Explain about Fig 1.8 and 1.9<br>Example 1.1, 1.2 |
| 4. | Exercise for RAM program | 39,40 | 2 | Ex 1.3, 1.5, 1.19<br>(Tutorial and Exercise) |
| 5. | 1.3 Computational Complexity of RAM programs | 12-14 | 2 | Explain how to express the uniform and logarithmic complexity of RAM program (Very Detail ) |
| 6. | Exercises for Complexity expression for RAM program | 39,40 | 2 | Ex. 1.3, 1.4, 1.5, 1.6, 1.9<br>(Tutorial and Exercise) |
| 7. | 1.4 A Stored Program Model (RASP) Different between RAM and RASP<br>Theorem 1.1 | 15-17 | 2 | Detail Explain RASP Model and Theorem 1.1 |
| 8. | Theorem 1.2<br>RASP Program Exercise | 18,19 | 2 | Detail Explain Theorem 1.2<br>Explain Complexity of RASP program |
| 9. | 1.5 Straight line programs<br>Bitwise Computations | 19-25 | 2 | Detail Explain for Bitwise Computational Model & Decision |

| | | | | |
|---|---|---|---|---|
| | Bit Vector Operations<br>Decision Tree | | | Tree |
| 10. | 1.6 Turing Machine Model | 25-31 | 2 | Explain how to construct a Turing Machine and its properties |
| 11. | Example 1.8 and 1.9 | 28,29 | 2 | Fig 1.20, 1.21, 1.22 |
| 12. | Exercise | 40 | 2 | 1.15, 1.16, 1.17 and Related Exercise |
| 13. | Revision | | 1 | Chapter Summary |
| 14. | Tutorial | | 1 | |
| **II.** | **Chapter (2)**<br>**Design of Efficient Algorithms** | 44 to 74 | **8** | Lectures + Tutorials + Exercise |
| 15. | 2.6Divide and Conquer<br>Definition, Two Examples of MAXMIN | 60-65 | 2 | Very important section<br>Detail Explain definition and some examples |
| 16. | 2.7Balancing | 65-67 | 1 | Briefly Explain |
| 17. | 2.8Dynamic Programming | 67-69 | 2 | Very important section<br>Detail Explain definition and some examples |
| 18. | Revision | | 1 | Chapter Summary |
| 19. | Exercises | | 1 | **Exercises from old questions** |
| 20. | Tutorial | | 1 | |
| **III** | **Chapter (3)**<br>**Sorting and Order Statistics** | 76 to 105 | **12** | Lectures + Tutorials + Exercise |
| 20. | 3.1 The Sorting Problem<br>3.2 Radix Sorting(Bucket Sort) | 76-78 | 1 | Explain about sorting problems<br>Briefly Explain |
| 21. | 3.2 Lexicographic Sort<br>(Algorithm 3.1) | 78-80 | 1 | Detail Explain |
| 22. | 3.2 Lexicographic Sort<br>(Algorithm 3.2) | 80-84 | 2 | Detail Explain |
| 23. | 3.4 Heap Sort<br>(Heapify, Build Heap Algorithms)<br>Analyze Complexity | 87-91 | 2 | Detail Explain |
| 24. | 3.5 Quick Sort<br>(Quick Sort, Partition Algorithms)<br>Analyze Complexity | 92-97 | 2 | Detail Explain |
| 25. | 3.6 Order Statistics<br>(Selection Algorithm)<br>Analyze Complexity | 97-99 | 1 | Detail Explain |
| 26. | Revision | | 1 | Chapter Summary |
| 27. | Exercises | | 1 | Ex. 3.1, 3.2, 3.4, 3.6, 3.7, 3.8, 3.11 |
| 28. | Tutorial | | 1 | |