CSC3202 Design and Analysis of Algorithm

COURSE INFORMATION SHEET

PROGRAMME: **Computer Science**	DEGREE: **BS**
COURSE: **Design and Analysis of Algorithm**	SEMESTER**: IV** CREDITS: **3**
COURSE CODE: **CSC3202**	COURSE TYPE: **CORE**
COURSE AREA/DOMAIN: **Computer Science**	CONTACT HOURS: **3 hours/Week**
CORRESPONDING LAB COURSE CODE (IF ANY): **Nil**	LAB COURSE NAME: **Nil**

COURSE OBJECTIVES:

The main objective of this course is to make the student familiar with subjects concerning algorithm complexity. By the end of the semester, the student should be able to: (1) recognize the use of several design techniques (greedy, divide-and-conquer, dynamic programming) and use these methods to solve simple Problems. (2) write and solve recurrence relations for recursive algorithms. (3) Determine asymptotic growth rates for algorithms. (4) Prove correctness of simple algorithm.

COURSE DESCRIPTION:

This course applies design and analysis techniques to numeric and nonnumeric algorithms which act on data structures. Design is emphasized so that the student will be able to develop new algorithms. Analysis of algorithms is concerned with the resources an algorithm must use to reach a solution. Topics include introduction to algorithm, asymptotic complexity, sorting and searching, divide and conquer, greedy graph algorithms, dynamic programming, data compression, backtracking, branch and bound.

SYLLABUS:

Week	Topics
1.	What are Algorithms, Why do you need to study Algorithm, What kinds of problems are solved by Algorithms, Algorithms as Technology, correctness and generality of Algorithms, Major Factors in Designing an Algorithm
2.	Pre-Condition, Post Condition, Partial Correctness of an Algorithm, Total Correctness of An Algorithm Loop Invariant, Correctness of an Iterative Algorithm
3.	Correctness of Recursive Algorithm, Mathematical Induction, Growth Rate of functions, Asymptotic Notations
4.	Major Assumptions in analyzing an algorithm, RAM Model Mathematical analysis of non recursive algorithms, Complexity Classes

5.	Time and Space Tradeoffs in algorithm, Brute force Algorithms and their Analysis
6.	Mathematical Analysis of recursive algorithms, Substitution Method, Tree Methods
7.	Master Theorem for solving recurrences, Divide and conquer Algorithm,
8.	Mid-Exam+ Merge Sort and its Worst case analysis,
9.	Best and Average Case Analysis of merge sort, Quick Sort and its Worst, Best and Average Case Analysis, Heap Sort
10.	Dynamic Programming, elements of Dynamic Programming, Comparison of Dynamic Programming with divide and Conquer , Knapsack Problem solving using Dynamic Programming
11.	Longest Common Subsequent problem solving through Dynamic Programming, Shortest Path finding through Dynamic Programming
12.	Graphs, and Graph Terminology, Graph Greedy Algorithms MST, Prims Algorithm,
13.	Kruskal Algorithm, Dijkstra Algorithm
14.	Optimization/Aproximation Algorithms
15.	Presentations

TEXT/REFERENCE BOOKS:

Introduction to Algorithms (3rd edition) by Thomas H. Corman, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein.
Algorithm Design, (1st edition, 2013/2014), Jon Kleinberg, Eva Tardos.
Algorithms, (4th edition, 2011), Robert Sedgewick, Kevin Wayne.

COURSE LEARNING OUTCOMES (CLOs):

CLO	DESCRIPTION	Domain	BT Level
1	Prove the correctness and analyze the running time of the basic algorithms for those classic problems in various domains.	C	1
2	Apply the algorithms and design techniques to solve problems.	C	2
3	Analyze the complexities of various problems in different domains.	C	3

MAPPING COURSE LEARNING OUTCOMES (CLOs) AND PROGRAM LEARNING OUTCOMES (PLOs):

	PROGRAM LEARNING OUTCOMES (PLOs)										
	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**
CLO.1			X								
CLO.2	X										
CLO.3		X									

PROGRAM LEARNING OUTCOMES (PLOs):

PLO	DESCRIPTION
a.	The ability to utilize logic, mathematics, and physical sciences to model and solve Computer Science problems.
b.	Proficiency in software design and development, design and analysis of algorithms theory of programming languages, operating systems, theory of computation and computer architecture.
c.	The ability to think critically, perform scientific analysis and develop solutions for typical Computer Science problems.