No student shall be admitted as a candidate for the examination for any of the Parts/Semesters after the lapse of four years from the date of admission to the Part-I/Semester-I of the programme.

12. ATTENDANCE REQUIREMENTS

No student shall be considered to have pursued a regular course of study unless he/she is certified by the Head of the Department of Computer Science, University of Delhi, to have attended 66.67% of the total number of lectures, tutorials, practicals, and seminars conducted in each semester, during his/her course of study. Provided that he/she fulfils other conditions, the Head, Department of Computer Science, may permit a student to attend the next semester who falls short of the required percentage of attendance by not more than 10 percent of the lectures, tutorials, and seminars conducted during the semester.

13. COURSE CONTENT FOR EACH COURSE

PART - I (SEMESTER - I)

MCSC101: DESIGN AND ANALYSIS OF ALGORITHMS [3-0-1]

Course Objectives

This course is designed to introduce advanced techniques of designing and analyzing algorithms. The course also familiarizes the students with some problems that are too hard to admit fast solutions. Some of the advanced algorithm design techniques provide good solutions to these problems.

Course Learning Outcomes

Upon successful completion of this course, the student will be able to:
CO1: understand advanced techniques to design algorithms like augmentation, randomization, parallelization and use of linear programming.
CO2: Analyse the strengths and weaknesses of each technique.
CO3: Identify and apply technique(s) suitable for simple applications.
CO4: Demonstrate correctness of algorithms and analyse their time complexity theoretically as well as practically.
CO5: Analyze algorithms in the probabilistic framework.
CO6: Understand and apply string matching to application at hand.
CO7: Understand what are parallel algorithms, their utility, and the notion of speedup.
CO8: be able to appreciate that certain problems are too hard to admit fast solutions and be able to prove their hardness.
CO9: understand what are approximation algorithms, their utility, and the notion of approximation ratio.

Syllabus:

Review: Review of Basic Sorting and Searching Algorithms, Greedy Algorithms Divide & Conquer and Dynamic Programming.
Augmentation: Maximum Flow and Min Cut Problems, Matching in bipartite graphs, Minimum weight matching.
String Processing: Finite Automata method, KMP.
Randomized algorithms: Introduction to Random numbers, randomized Qsort, randomized selection, randomly built BST, randomized min-cut.
Parallel Algorithms: Shared Memory Model, Distributed Memory Model, Speedup. Searching, sorting, selection, matrix-vector multiplication, prefix-sum.
Linear Programming: Formulating an LP, Feasible region and Convex Polyhedron, Simplex Algorithm, LP-rounding to obtain integral solutions, Primal-Dual Algorithm.

Introduction to Complexity Classes: Classes P, NP - Verifiability, NP-Hard - Reducibility, NP Complete.
Introduction to Approximation Algorithms.

Readings

1. J. Kleinberg and E.Tardos, "Algorithm Design", 1st Edition 2013., Pearson Education India,
2. Sanjoy Dasgupta, Christos Papadimitriou and Umesh Vazirani, "Algorithms", 1st Edition, 2017, Tata McGraw Hill.
3. T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein, "Introduction to Algorithms", 3rd Edition, 2010, Prentice-Hall of India Learning Pvt. Ltd.
4. Vijay V. Vazirani, "Approximation Algorithms", 2013, Springer.
5. Bernhard Korte and Jens Vygen, "Combinatorial Optimization: Theory and Algorithms (Algorithms and Combinatorics)", 6th edition, 2018, Springer.
6. Rajeev Motvani and Prabhat Raghavan, 2004, Cambridge University Press.

MCSC102: ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING [3-0-1]

Course Objectives: Beginning with a comprehensive overview of the AI techniques, the course introduces the supervised and unsupervised ML techniques, alongwith their applications in solving real-world problems. The course also covers evaluation and validation methods for ML models.

Course Learning Outcomes:

On successful completion of this course, the student will be able to:

CO1: discuss Turing Test, and various methods of knowledge representation as applicable to a given context.

CO2: design and implement supervised and unsupervised machine learning algorithms for real-world applications while understanding the strengths and weaknesses.

CO3: analyse the computational complexity of various machine learning algorithms.