UNIVERSITY OF DELHI

MASTER OF SCIENCE

(M.Sc. Computer Science)

CONTENTS

I. About the Department

Department of Computer Science was established at the University of Delhi, in the year 1981, with the objective of imparting quality education in the field of Computer Science. With rapidly evolving technology and continuous need for innovation, the department has been producing quality professionals, holding important positions in the Information Technology industry both in India and abroad.

The Department started Master of Computer Applications (MCA) programme in the year 1982, which was among the first such programmes in India. The MCA programme focuses on providing a sound theoretical background as well as good practical exposure to students in the relevant areas. It is intended to provide a modern, industry-oriented education in applied computer science. It aims at producing trained professionals who can successfully meet the demands of the IT industry. They obtain skills and experience in up-to-date approaches to analysis, design, implementation, validation, and documentation of computer software and hardware.

The Department started M.Sc. Computer Science course in the year 2004 with the aim to develop core competence in Computer Science and to prepare the students to take up challenges of research and development. The students develop the ability to apply a high level of theoretical expertise and innovation to complex problems and application of new technologies. M.Sc. programme has been designed to teach the mathematical principles of specification, design and efficient implementation of both software and hardware.

The Department also offers Doctor of Philosophy (Ph.D.) programme, aimed at producing quality researchers in several diverse branches of Computer Science. The Department also coordinates B.Sc. (H) Computer Science, B.Sc. Physical Science (Computer Science) and other courses taught at constituent colleges of University of Delhi.

II. Introduction to CBCS (Choice Based Credit System)

Choice Based Credit System:

The CBCS provides an opportunity for the students to choose courses from the prescribed courses comprising core, elective/minor or skill-based courses. The courses will be evaluated following the grading system, which is considered to be better than the conventional marks system. Grading system provides uniformity in the evaluation and computation of the Cumulative Grade Point Average (CGPA) based on student's performance in examinations which enables the student to move across institutions of higher learning. The uniformity in evaluation system also enables the potential employers in assessing the performance of the candidates.

Definitions:

(i) 'Academic Programme' means an entire course of study comprising its programme structure, course details, evaluation schemes etc. designed to be taught and evaluated in a teaching Department/Centre or jointly under more than one such Department/ Centre

(ii) 'Course' means a segment of a subject that is part of an Academic Programme

(iii) 'Programme Structure' means a list of courses (Core, Elective, Open Elective) that makes up an Academic Programme, specifying the syllabus, Credits, hours of teaching, evaluation and examination schemes, minimum number of credits required for successful completion of the programme etc. prepared in conformity to University Rules, eligibility criteria for admission

(iv) 'Core Course' means a course that a student admitted to a particular programme must successfully complete to receive the degree and which cannot be substituted by any other course

(v) 'Elective Course' means an optional course to be selected by a student out of such courses offered in the same or any other Department/Centre

(vi) 'Open Elective' means an elective course which is available for students of all programmes, including students of same department. Students of other Department will opt these courses subject to fulfilling of eligibility of criteria as laid down by the Department offering the course.

(vii) 'Credit' means the value assigned to a course which indicates the level of instruction; One-hour lecture per week equals 1 Credit, 2 hours practical class per week equals 1 credit. Credit for a practical could be proposed as part of a course or as a separate practical course

(viii) 'SGPA' means Semester Grade Point Average calculated for individual semester.

(ix) 'CGPA' is Cumulative Grade Points Average calculated for all courses completed by the students at any point of time. CGPA is calculated each year for both the semesters clubbed together.

(x) 'Grand CGPA' is calculated in the last year of the course by clubbing together of CGPA of two years, i.e., four semesters. Grand CGPA is being given in Transcript form. To benefit the student a formula for conversation of Grand CGPA into %age marks is given in the Transcript.

III. M.Sc. Programme Details:

Programme Objectives (POs):

The M.Sc. Computer Science programme was introduced in 2004. It is a four-semester course which aims at imparting quality education in core Computer Science, so that the students are prepared to face the challenges of the highly competitive IT industry as well as carry out research and development. The objective of the programme is to impart sound knowledge of theory and hands on practical skills in various areas of Computer Science. Considering, the Computer Science curriculum that the students have undertaken at the graduate level, it aims at imparting advanced courses in Computer Science. The course structure includes a minor project in the third semester followed by a major project in the final semester which helps in development of research skills in the areas of their interest.

Programme specific outcomes (PSOs):

The programme is designed to

PSO1: prepare the students to take up a career in the highly competitive IT industry with research and development skills acquired through minor and major projects.

PSO2: equip students with comprehensive knowledge and understanding of advanced theoretical fundamentals in computer science as well as state-of-the-art key research issues in specialized areas of computer science.

PSO3: provide freedom to choose subjects of interest from the list of specialized courses and to allow the students to follow the career path they have dreamt of.

PSO4: develop expertise in advanced areas of computer science-Algorithms, Artificial Intelligence, Data Mining, Information Security to name a few, while learning key skills in research and development.

Programme Structure:

The M.Sc. programme is a two-year course divided into four-semesters. A student is required to complete 103 credits for the completion of course and the award of degree.

		Semester	***Semester***
Part – I	First Year	Semester I	Semester II
Part – II	Second Year	Semester III	Semester IV

Course Credit Scheme

Semester	Core Courses			Elective Course			Open Elective Course			Total Credits
	No. of courses	Credits (Th+P+T)	**Total Credits**	No. of courses	Credits (Th+P+T)	**Total Credits**	No. of courses	Credits (Th+P+T)	**Total Credits**	
I	5	20+5+0	**25**	0	0	**0**	0	0	**0**	**25**
II	3	12+3+0	**15**	1	4+X+X	**5**	1	3+X+X	**4**	**24**
III	Minor Project	10	**10**	2	8+X+X	**10**	1	3+X+X	**4**	**24**
IV	Major Project	25	**25**	0	0	**0**	0	0	**0**	**25**
Total Credits for the Course			**75**			**15**			**8**	**98**

* For each Core and Elective Course there will be 4 lecture hours of teaching per week.
* Open Electives to the maximum total of 8 credits.
* Duration of examination of each course shall be 3 hours.
* Each course will be of 100 marks out of which 70 marks shall be allocated for end semester examination and 30 marks for internal assessment.
* Each student shall carry out a minor project in the third semester and a major project in the fourth semester. The projects will be carried out under the supervision of a teacher of the department. Depending on the requirement, a joint supervisor may also be appointed. The project will be evaluated jointly by the internal supervisor(s) and an examiner to be appointed by the Department in consultation with the internal supervisor(s). The minor and the major projects shall carry 200 and 500 marks respectively distributed as follows:
  a. Mid-semester evaluation: 30% weightage
     (to be carried out by the internal supervisor(s))
  b. End-semester evaluation
     (to be carried out jointly by the internal supervisor(s) and external examiner)
        (i) Dissertation: 30% weightage
        (ii) Viva-voce: 40% weightage
*To be eligible to pass a course and earn credits for it, a student must satisfy the criteria laid down by the University.
* Examination for courses specified in the odd (even) semesters shall be conducted only in the respective odd (even) semesters.
* Promotion Criteria: As laid down by the University.
* Award of degree: In order to be eligible for the award of the degree of M. Sc. (Computer Science) degree, a student must earn all the credits (103) as per the structure of the course, specified in the above table.

Semester wise Details of M.Sc. Course

Semester I

Semester I					
	Number of core courses				5
Course Code	**Course Title**	**Credits in each core course**			
		Theory	Practical	Tutorial	**Total**
MCSC101	Design and Analysis of Algorithms	4	1	0	**5**
MCSC102	Artificial Intelligence	4	1	0	**5**
MCSC103	Information Security	4	1	0	**5**
MCSC104	Mathematical Foundations of Computer Science	4	1	0	**5**
MCSC105	Data Mining	4	1	0	**5**
	Total credits in core course				25
	Number of elective courses				0
	Total credits in elective course				0
	Number of open electives				0
	Total credits in elective course				0
	Total credits in Semester I				**25**

Semester II

Semester II					
	Number of core courses				**4**
Course Code	**Course Title**	**Credits in each core course**			
		Theory	Practical	Tutorial	**Total**
MCSC201	Machine Learning	4	1	0	**5**
MCSC202	Advanced Operating Systems	4	1	0	**5**
MCSC203	Mobile and Satellite Communication Networks	4	1	0	**5**
	Total credits in core course				15
	Number of elective courses				**1**
		Theory	Practical	Tutorial	**Total**
	Elective 1	4	1/0	0/1	**5**
	Total credits in elective courses				5
	Number of open electives				**1**
	Credits in each open elective	Theory	Practical	Tutorial	**Total**
	Open Elective 1	3	1/0	0/1	**4**
	Total credits in open elective				4
	Total credits in Semester II				**24**

List of Electives for Semester II (One of the following)		
Course Code	**Course Title**	**Th-P-T**
MCSE201	Combinatorial Optimization	4-1-0
MCSE202	Digital Image Processing	4-1-0
MCSE203	Compiler Design	4-1-0
MCSE204	Database Applications	4-1-0
List of Open Electives for Semester II (One of the following)		
Course Code	**Course Title**	**Th-P-T**
MCSO201	Java Programming	3-1-0
MCSO202	GPU Programming	3-1-0
XXXXXXX	Open Elective floated by other department	3-X-X

Semester III

Semester III					
	Number of core courses				1
Course Code	**Course Title**	**Credits in each core course**			
		Theory	Practical	Tutorial	**Total**
MCSC301	Minor Project	10	0	0	**10**
	Total credits in core course				10
	Number of elective courses				2
	Credits in each open elective	Theory	Practical	Tutorial	**Total**
	Elective course 2	4	1/0	0/1	**5**
	Elective course 3	4	1/0	0/1	**5**
	Total credits in elective courses				10
	Number of open electives				1
	Credits in each open elective	Theory	Practical	Tutorial	**Total**
	Open Elective 2	3	1/0	0/1	**4**
	Total credits in open elective				4
	Total credits in Semester III				24

List of Elective Courses for Semester III (Two courses of the following)		
Course Code	**Course Title**	**Th-P-T***
MCSE301	Cyber Security	4-1-0
MCSE302	Graph Theory	4-0-1
MCSE303	Network Science	4-1-0
MCSE304	Deep Learning	4-1-0
MCSE305	Neural Networks	4-1-0
MCSE306	Modeling and Simulation	4-1-0
MCSE307	Computational Intelligence	4-1-0
MCSE308	Parallel and Distributed Computing	4-1-0
MCSE309	Software Quality Assurance and Testing	4-1-0
MCSE310	Text analytics	4-1-0
MCSE311	Multi-Agent Systems	4-1-0
MCSE312	Steganography and Digital Watermarking	4-1-0
MCSE313	NP Completeness and Approximation Algorithms	4-0-1
List of Open Courses for Semester III (One of the following)		
Course Code	**Course Title**	**Th-P-T***
MCSO301	Data Science**	3-1-0
MCSO302	E-Commerce	3-1-0
XXXXXXX	Open elective from other department	3-X-X

***Th-P- T: Theory -Practical- Tutorials**

** Only for students of other departments

Semester IV

Semester IV		
	Number of core courses	1
Course Code	**Course Title**	**Credits in each core course**
MCSC401	Project Work	25
	Total credits in core course	25
	Number of elective courses	0
	Total credits in elective courses	0
	Number of open electives	0
	Total credits in open elective	0
	Total credits in Semester IV	25

Total credits of the course = 25+24+24+25 = 98

Total credits required for degree = 98

LIST OF ELECTIVE COURSES:

Semester-II

Elective Courses, 4+1 credits each

Course Code	Course Title
MCSE201	Combinatorial Optimization
MCSE202	Digital Image Processing
MCSE203	Compiler Design
MCSE204	Database Applications

Open Elective Courses, 3+1 credits each

Course Code	Course Title
MCSO201	Java Programming
MCSO202	GPU Programming
XXXXXXX	Inter-Departmental Elective***

Semester-III

Elective Courses, 4+1 credits each

Course Code	Course Title
MCSE301	Cyber Security
MCSE302	Graph Theory
MCSE303	Network Science
MCSE304	Deep Learning
MCSE305	Neural Networks
MCSE306	Modeling and Simulation
MCSE307	Computational Intelligence
MCSE308	Parallel and Distributed Computing
MCSE309	Software Quality Assurance and Testing
MCSE310	Text analytics
MCSE311	Multi-Agent Systems
MCSE312	Steganography and Digital Watermarking
MCSE313	NP Completeness and Approximation Algorithms

Open Elective Courses, 3+1 credits each

Course Code	Course Title
MCSO301	Data Science**
MCSO302	E-Commerce
XXXXXXX	Inter-Departmental Elective***

**** Only for students of other departments**

*****As per the elective offered by the concerned Department.**

Selection of Elective Courses:

The students may select the elective courses out of the list of courses which are offered in a semester.

Teaching:

There shall be 90 instructional days excluding examination in a semester.

The faculty of the Department is primarily responsible for organizing teaching work for M.Sc. Computer Science programme. Based on the requirement, faculty from some other Departments and constituent colleges may also be associated with lecture/tutorial/practical work in the Department.

Eligibility for Admissions:

A. Mode of admission: To be decided by the University at the beginning of every academic year. Details for Academic year 2018-19 are as follows:

a) There are two modes of admission for MSc Computer Science programme.

i. Merit or Direct**:** 50% of the total intake shall be filled by direct admission of the students of the University of Delhi only, on the basis of merit. This mode is not available to students from other Universities.

ii. Entrance: Remaining 50% of the seats shall be filled through entrance examination and Interview/Group Discussion, if any.

b) It may be noted that all the applicants seeking admission to the M.Sc. Computer Science Programme are required to register online, <u>irrespective of the mode of admission</u>.

c) Applicants can choose to apply in these two modes under the following three options:

(i) Merit only, (ii) Entrance only, or (iii) Merit + Entrance.

d) The Department will release the Admission lists in both modes, separately for all categories.

e) The candidates who qualify the entrance test must appear for an Interview. For preparing the final merit list, 85% weightage will be given to the score in the Entrance Exam and 15% weightage will be given to the score in the Interview.

B. **Seats Distribution* and Eligibility criteria for M.Sc. Computer Science programme**

Total Seats: 46

Admission Mode	**General**	**SC**	**ST**	**OBC**	**Total**
Entrance	12	3	2	6	23
Merit	12	3	2	6	23

***University reserves the rights to revise, amend, or update the number of seats in various courses without giving any prior notice. Any change so made shall be updated on the PG admission portal.**

The reservation policy related to admissions will be followed as per the University Rule.**

C. Eligibility Criteria

Eligibility in Entrance Category		
Category Id	**Course Requirements**	**Marks Requirements**
1	B.Sc. (Honours) Computer Science (10+2+3 scheme) and B.Tech. Computer Science (10+2+4 scheme) from University of Delhi or any other University whose examination is recognized as equivalent to that of University of Delhi.	60% marks in aggregate or equivalent of CGPA as per University norms wherever it is applicable
2	B.Sc. Program/B.Sc. Applied Physical Sciences with Mathematics and Computer Science / B.Sc.(General) Mathematical Sciences, (10+2+3 scheme) with Mathematics and Computer Science from University of Delhi or any other University whose examination is recognized as equivalent to that of University of Delhi.	60% marks in aggregate with 60% marks in Computer Science and Mathematics separately or equivalent of CGPA as per University norms wherever it is applicable
3	Any Bachelor's Degree (10+2+3) of University of Delhi with at least six papers in Computer Science and two papers in Mathematics under Semester system / at least three papers in Computer Science and one paper in Mathematics under Annual Examination System or any other University whose examination is recognized as equivalent to that of University of Delhi.	60% marks in aggregate with 60% marks in Computer Science and Mathematics separately or equivalent of CGPA as per University norms wherever it is applicable
Eligibility in Merit or Direct Category		
Category Id	**Course Requirements**	**Marks Requirements**
4	B.Sc. (H) Computer Science (10+2+3 Scheme) from University of Delhi	60% marks in aggregate or equivalent of CGPA as per University norms wherever it is applicable

Note

1. ***The candidates who are appearing in the final year examinations of the Bachelor's degree on the basis of which admission is sought are eligible to apply in all categories mentioned above.***
2. ***Relaxation will be given to the candidates belonging to SC, ST and OBC category as per the University rules***
3. ***50% of the seats will be filled on the basis of merit in B.Sc.(H) Computer Science examination of University of Delhi. Students with gap year will be considered.***

4. ***The admission for the remaining 50% seats in the M.Sc. Computer Science course is based on Admission Test & Interview. For preparing the final merit list, 85% weightage will be given to the score in the Entrance Exam and 15% weightage will be given to the score in the Interview. The entrance examination shall be of two hours duration.***

5. ***For More information please visit <u>http://cs.du.ac.in/admission/mcs/</u>***

Assessment of Students' Performance and Scheme of Examinations:

A. English shall be the medium of instruction and examination.

B. Examinations shall be conducted at the end of each semester as per the academic calendar notified by the University of Delhi.

C. Assessment of students' performance and scheme of evaluation shall be as per University rules.

Pass Percentage & Promotion Criteria: As per University rules.

Conversion of Marks into Grades

A. Grade Points

   Grade point table as per University Examination rules.

B. CGPA Calculation

   As per University Examination rules.

C. SGPA Calculation

   As per University Examination rules.

D. Grand CGPA Calculation

   As per University Examination rules.

E. Conversion of Grand CGPA into Marks

   As per University Examination rules.

F. As notified by competent authority the formula for conversion of Grand CGPA into marks is: Final %age of marks = CGPA based on all four semesters × 9.5

Division of Degree into Classes

Post Graduate degree to be classified based on CGPA obtained into various classes as notified into Examination policy.

Attendance Requirement

As per University rule.

Span Period

As per University rule.

Guidelines for the Award of Internal Assessment Marks for the M. Sc. Computer Science Programme:

Performance of the students will be evaluated based on a comprehensive system of continuous evaluation. For each course, there shall be a minor test, assignments/ laboratory work. There shall be monitoring committee to be constituted at beginning of each semester to monitor the internal assessment.

IV: Content Details for M.Sc. Computer Science Programme

SEMESTER - I

MCSC101: DESIGN AND ANALYSIS OF ALGORITHMS [4-1-0]

Course Outcomes:

On 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, identify and apply suitable technique(s) for simple applications.
CO3: demonstrate correctness of algorithms and analyse time complexity - theoretically as well as practically.
CO4: analyze algorithms in the probabilistic framework.
CO5: apply string matching to application at hand.
CO6: understand what are parallel algorithms, their utility, and the notion of speedup.
CO7: be able to appreciate that certain problems are too hard to admit fast solutions and be able to prove their hardness.
CO8: understand what are approximation algorithms, their utility, and the notion of approximation ratio.

Syllabus:

Unit-I Review: Review of Basic Sorting and Searching Algorithms, Greedy Algorithms
Divide & Conquer and Dynamic Programming.

Unit-II Augmentation: Maximum Flow and Min Cut Problems, Matching in bipartite graphs, Minimum weight matching.

Unit-III String Processing: Finite Automata method, KMP.

Unit-IV Randomized algorithms: Introduction to Random numbers, randomized Qsort, randomized selection, randomly built BST, randomized min-cut.

Unit-V Parallel Algorithms: Shared Memory Model, Distributed Memory Model, Speedup. Searching, sorting, selection, matrix-vector multiplication, prefix-sum.

Unit-VI Linear Programming: Formulating an LP, Feasible region and Convex Polyhedron, Simplex Algorithm, LP-rounding to obtain integral solutions, Primal-Dual Algorithm.

Unit-VII Introduction to Complexity Classes: Classes P, NP - Verifiability, NP-Hard - Reducibility, NP Complete.

Unit-VIII 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 [4-1-0]

Course Outcomes:

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

CO1: differentiate between various approaches to Artificial Intelligence.
CO2: design intelligent agents and distinguish between Utility based agents and Goal based agents.
CO3: apply concepts, methods, and theories of search, heuristics, games, knowledge representation, planning.
CO4: apply Natural language processing techniques.
CO5: understand the limitations of Artificial Intelligence techniques.

Syllabus:

Unit-I Introduction: Introduction to Artificial Intelligence, various definitions of AI, AI Applications and Techniques, Turing Test and Reasoning - forward & backward chaining.

Unit-II Intelligent Agents: Introduction to Intelligent Agents, Rational Agent, their structure, reflex, model-based, goal-based, and utility-based agents, behavior and environment in which a particular agent operates.

Unit-III Problem Solving and Search Techniques: Problem Characteristics, Production Systems, Control Strategies, Breadth First Search, Depth First Search, iterative deepening, uniform cost search, Hill climbing and its Variations, simulated annealing, genetic algorithm search; Heuristics Search Techniques: Best First Search, A* algorithm, AO* algorithm, Minmax & game trees, refining minmax, Alpha – Beta pruning, Constraint Satisfaction Problem, Means-End Analysis.

Unit-IV Knowledge Representation: Introduction to First Order Predicate Calculus, Resolution Principle, Unification, Semantic Nets, Conceptual Dependencies, semantic networks, Frames system, Production Rules, Conceptual Graphs, Ontologies.

Unit-V Planning: Basic representation for planning, symbolic-centralized vs. reactive-distributed,partial order planning algorithm.

Unit-VI Reasoning with Uncertain Knowledge: Different types of uncertainty - degree of belief and degree of truth, various probability constructs - prior probability, conditional probability, probability axioms, probability distributions, and joint probability distributions, Bayes' rule, other approaches to modeling uncertainty such as Dempster-Shafer theory and fuzzy sets/logic.

Unit-VII Understanding Natural Languages: Components and steps of communication, contrast between formal and natural languages in the context of grammar, parsing, and semantics, Parsing Techniques, Context-Free and Transformational Grammars.

Readings:

1. S. Russell and P. Norvig, **Artificial Intelligence: A Modern Approach, 3rd edition**, Pearson Education, 2015.

2. Elaine Rich and Kelvin Knight, Artificial Intelligence, 3rd edition, Tata McGraw Hill , 2017.

3. DAN.W. Patterson, **Introduction to A.I. and Expert Systems** – PHI, 2007.

4. Michael Wooldridge, **An Introduction to MultiAgent Systems**, 2nd edition, John Wiley & Sons, 2009.

5. Fabio Luigi Bellifemine, Giovanni Caire, Dominic Greenwood, **Developing Multi-Agent Systems with JADE**, Wiley Series in Agent Technology, John Wiley & Sons, 2007.

6. W.F. Clocksin and C.S. Mellish, **Programming in PROLOG**, 5th edition, Springer, 2003.

7. Saroj Kaushik, **Logic and Prolog Programming**, New Age International Publisher, 2012.

8. Ivan Bratko, **Prolog Programming for Artificial Intelligence**, Addison-Wesley, Pearson Education, 4th edition, 2011.

MCSC103: INFORMATION SECURITY [4-1-0]

Course Outcomes:

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

CO1: describe various information security issues.

CO2: implement a symmetric and asymmetric cryptographic methods.

CO3: describe the role and implementation of digital signatures.

CO4: describe security mechanisms like intrusion detection, auditing and logging.

Syllabus:

Unit-I Overview of Security: Protection versus security; aspects of security– confidentiality, data integrity, availability, privacy; user authentication, access controls, Orange Book Standard.

Unit-II Security Threats: Program threats, worms, viruses, Trojan horse, trap door, stack and buffer overflow; system threats- intruders; communication threats- tapping and piracy.

Unit-III Computer Security Models: BLP Model, BIBA Model, HRU Model.

Unit-IV Cryptography: Substitution, transposition ciphers, symmetric-key algorithms: Data Encryption Standard, Advanced Encryption Standard, IDEA, Block cipher Operation, Stream Ciphers: RC-4. Public key encryption: RSA, ElGamal. Diffie-Hellman key exchange. Elliptic Curve, EC cryptography, Message Authentication code (MAC), Cryptographic hash function.

Unit-V Digital signatures: ElGamal digital signature scheme , Elliptic Curve digital signature scheme, NISTdigital signature scheme.

Unit-VI Key Management and Distribution : Symmetric Key Distribution, X.509 Certificate public key infrastructures.

Unit-VII Intrusion detection and prevention.

Readings:

1. W. Stalling, **Cryptography and Network Security Principles and Practices** (7^{th} ed.), Pearson of India, 2018.

2. A.J. Elbirt, **Understanding and Applying Cryptography and Data Security,** CRC Press, Taylor Francis Group, New York, 2015.

3. C. Pfleeger and SL Pfleeger, Jonathan Margulies, **Security in Computing** (5th ed.), Prentice-Hall of India, 2015

4. M. Merkow and J. Breithaupt, **Information Security: Principles and Practices**, Pearson Education, 2006.

MCSC104: MATHEMATICAL FOUNDATIONS OF COMPUTER SCIENCE [4-1-0]

Course Outcomes:

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

CO1: apply vector operations to solve problems in different areas of computer science.

CO2: perform operations on matrices and sparse matrices, compute determinant, rank and eigen values of a matrix

CO3: apply matrix algebra to solve problems in different areas of computer science.

CO4: perform data analysis in probabilistic framework.

CO5: visualize and model the given problem using mathematical concepts covered in the course.

Syllabus:

Unit-I Linear Algebra: Introduction to Vector space, Subspace, Linear Independence and Dependence, Basis and Dimensions, Convex set, Rank of a matrix, System of linear equations, Orthogonal bases, Projection, Gram-Schmidt orthogonality process, Linear Mappings, Kernel and

Image space of a linear map, Matrix associated with linear map, Eigen values and Eigen vectors, PCA, SVD, Applications in Data Reduction, Text Analysis and Image Processing.

Unit-II Probability and Statistics: Review of Probability Theory, Conditional Probability, Independent events, Bayes' theorem and it application in data analysis, Descriptive Statistics, Exploratory data analysis, Coefficient of variation, Skewness, Kurtosis, Data visualization, Scatter diagram, Grouped data, Histograms, Ogives, Percentiles, Box Plot, .

Unit-III Random variable: Introduction to random variable, Discrete random variables (Bernoulli, Binomial, Multinomial, Poisson, Geometric, Negative Binomial), Continuous random variables (Uniform, Exponential, Normal, Gamma), Expectation, variance, Conditional probability and conditional expectation, Central Limit Theorem, Markov and Chebyshev's inequality.

Readings:

1. Serge Lang, **Introduction to Linear Algebra,** 2nd Edition, Springer, 1986.
2. Gilbert Strang, **Introduction to Linear Algebra**, 4th Edition, Wellesley-Cambridge Press, 2009.
3. Sheldon M. Ross, **Probability Models for Computer Science**, Academic Press, 2002.
4. Ernest Davis, **Linear Algebra and Probability for Computer Science Applications**, CRC Press, 2012.
5. Kishor S. Trivedi, Probability and Statistics with Reliability, **Queuing and Computer Science Applications**, John Wiley, 2016.
6. Richard Cotton, **Learning R: a step by step function guide to data analysis,** O'reilly (SPD), Sixth edition reprint, 2017.
7. Mark Gardener, **Beginning R: The statistical programming language,** WILEY, 2017

MCSC105: DATA MINING [4-1-0]

Course Outcomes:

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

CO1: distinguish between the process of knowledge discovery and Data Mining.

CO2: follow formal notations and understand the mathematical concepts underlying data mining algorithms.

CO3: play with basic data exploration methods to develop understanding of given data and identify suitable pre-processing method for the given problem.

CO4: describe different data mining tasks and apply algorithms suitable for the given task.

CO5: use programming tools (e.g. Weka/Python/R etc.) for solving data mining tasks.

Syllabus:

Unit-I Overview: The process of knowledge discovery in databases, predictive and descriptive data mining techniques, supervised and unsupervised learning techniques.

Unit-II Data preprocessing : Data cleaning, Data transformation, Data reduction, Discretization.

Unit-III Classification: Supervised learning for predictive data mining, Basic issues in predictive data mining Decision trees, Decision rules, Statistical classification, Instance-based methods (nearest neighbor), Evaluation and Validation methods.

Unit-IV Clustering : Unsupervised learning for descriptive data mining, Basic issues in clustering, Partitioning methods, Hierarchical methods for clustering, Density-based methods, Cluster Validation methods and metrics.

Unit-V Association Rule Mining: Frequent item set, Maximal and Closed itemsets, Apriori property, Apriori algorithm.

Readings:

1. J Zaki Mohammed and Wagner Meira, **Data Mining and Analysis: Fundamental Concepts and Algorithms,** Cambridge University Press, 2014.

2. P. Tan, M. Steinbach, A Karpatne, and V. Kumar, **Introduction to Data Mining**, 2nd Ed., Pearson Education, 2018.

3. Jiawei Han and Micheline Kamber, **Data Mining: Concepts and Techniques**,3nd Ed., Morgan Kaufmann, 2011.

4. Charu C Agrawal, **Data Mining: The Textbook**, Springer, 2015.

SEMESTER - II

MCSC201: MACHINE LEARNING [4-1-0]

Course Outcomes:

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

CO1: appreciate machine learning approach to Artificial Intelligence, and understand fundamental issues and challenges of supervised and unsupervised learning techniques.

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

CO3: appreciate the underlying mathematical relationships within and across Machine Learning algorithms.

CO4: fine tune machine learning algorithms and evaluate models generated from data.

Syllabus:

Unit-I Introduction: Learning theory, Hypothesis and target class, Inductive bias and bias-variance tradeoff, Occam's razor, Limitations of inference machines, Approximation and estimation errors.

Unit-II Supervised learning: Linear separability and decision regions, Linear discriminants, Bayes optimal classifier, Linear regression, Standard and stochastic gradient descent, Lasso and Ridge Regression, Logistic regression, Support Vector Machines, Perceptron, Back propogation, Artificial Neural Networks, Decision Tree Induction, Overfitting, Pruning of decision trees, Bagging and Boosting, Dimensionality reduction and Feature selection.

Unit-III Support Vector Machines: Structural and empirical risk, Margin of a classifier, Support Vector Machines, Learning nonlinear hypothesis using kernel functions.

Unit IV Evaluation: Performance evaluation metrics, ROC Curves, Validation methods, Bias-variance decomposition, Model complexity.

Unit-V Unsupervised learning: Clustering, Mixture models, Expectation Maximization, Spectral Clustering, Non-parametric density estimation.

Unit-VI Introduction to Deep Networks: Introduction to deep feedforward networks, convolutional neural networks, stacking, striding and pooling.

Readings:

1. E. Alpaydin, **Introduction to Machine Learning**, 3rd Edition, Prentice Hall of India, 2014.

2. T Hastie, R Tibshirani and J Friedman, **The Elements of Statistical Learning Data Mining, Inference, and Prediction**, 2nd Edition, Springer, 2009.

3. C. M. Bishop, **Pattern Recognition and Machine Learning**, Springer, 2010.

4. R. O. Duda, P. E. Hart, and D.G. Stork, **Pattern Classification**, John Wiley and Sons, 2012.

5. Simon O. Haykin, **Neural Networks and Learning Machines**, Pearson Education, 2016

MCSC202: ADVANCED OPERATING SYSTEMS [4 -1- 0]

Course Outcomes:

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

CO1: develop skill to write new kernel routines or modify the existing ones for performance tuning.
CO2: write their own file systems by using the file management routines and port it in the open system as application process.
CO3: gather different run time statistics of the operating systems and adapt the system parameters for optimizing the performance.
CO4: derive benefit of application execution from the network or distributed operating over the stand alone operating system.

Syllabus:

Unit-I Process Management: System calls for process creation and termination, invoking other programs, changing size of a process, process scheduling schemes, time, clock, multi-threading.

Unit-II Interprocess Communication and Synchronization: Interprocesss communication mechanisms, signals handling, network communication mechanisms, process synchronization in multiprocessor environment.

Unit-III Memory Management: Swapping, demand paging, hybrid memory management with swapping and demand paging.

Unit-IV File Management: Internal representation of files, buffer cache allocation of disk blocks, mounting and unmounting of file systems, file systems maintenance.

Unit-V Network and distributed operating systems: Network operating systems and applications, client server applications, advantages of network operating over standalone PC operating systems, distributed operating systems and distributed applications, advantages of distributed operating systems over centralized operating systems, remote procedure call, distributed file system. distributed clock synchronization, mutual exclusions.

Readings:

1. Maurice J. Bach, **Design of the UNIX Operating System**, Prentice Hall, 1986.

2. Silberschatz, Galvin and Gagne, **Operating Systems concepts**, Wiley, 2013.

3. A S Tanenbaum, Maarten Van Steen, **Distributed Operating systems**, Pearson.2014.

MCSC203: MOBILE AND SATELLITE COMMUNICATION NETWORKS[4 -1- 0]

Course Outcomes:

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

CO1: describe various wireless communication techniques.
CO2: describe various multiple access schemes and tune the multiple access parameters to reduce call drops and enhance the quality of conversation.
CO3: describe different wireless networks and develop skills for improving spectrums reusability, interconnectivity and interoperability between different service providers.
CO4: develop skills to write new GPS software.

Syllabus:

Unit-I Wireless Communication Principles: Wireless propagation characteristics, multipath fading, intrusion handling, modulation techniques and bandwidth estimations, Direct Sequence and Frequency Hopping Spread Spectrum technologies.

Unit-II Multiple access and Duplexing techniques: Frequency Division Multiple Access, Time Division Multiple Access, Code Division Multiple Access, Space Division Multiple Access, Wavelength Division Multiple Access, duplexing techniques- Time Division Duplexing, Frequency Division Duplexing.

Unit-III Mobile cellular networks: Global Systems for Mobile combinations (GSM), General Packet Radio Services (GPRS), Enhanced Data rates for GSM Evolution (EDGE), Mobility and Hands-off in mobile cellular networks. 2G 3G, 4G mobile communications Networks.

Unit-IV Wireless Local Area Networks: Carrier Sense Multiple Access (CSMA/CA) protocol. Distributed Coordination Function, Point Coordination Function, Infrastructure based WLAN, ADHOC WLAN, IEEE 802.11 WLAN standards.

Unit-V Satellite Communication and Networks: Geosynchronous satellites, Medium Earth Orbit satellites, Global Positioning Systems, Low Earth Orbit Satellites, ALOHA, VSAT networks.

Readings:

1. A.S. Tanenbaum, David J Wetherall, **Computer Networks**, 5th Edition , Pearson, 2013.
2. Behrouz A. Forouzan, **Data Communications and Networking**, 5th Edition, McGraw Hill 2017.
3. C.N. Thurwachter, **Wireless Networking**, Prentice-Hall of India, 2002.
4. M. Richharia, **Mobile Satellite Communications: Principles & Trends,** Pearson Education, 2014.

MCSE201: COMBINATORIAL OPTIMIZATION[4-1-0]

Course Outcomes:

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

CO1: model problems using linear and integer programs.
CO2: differentiate between the computational complexities of LP and IP.
CO3: understand polyhedral analysis and apply it to develop algorithms.
CO4: understand the concept of duality and use it to design exact and approximate algorithms.
CO5: understand and explain the mathematical theory forming the basis of many algorithms for combinatorial optimization (particularly graph theoretic).

Syllabus:

Unit-I Introduction: Optimization problems, neighborhoods, local and global optima, convex sets and functions, simplex method, degeneracy; duality and dual simplex algorithm, computational considerations for the simplex and dual simplex algorithms-Dantzig-Wolfe algorithms.

Unit-II Integer Linear Programming: Cutting plane algorithms, branch and bound technique and approximation algorithms for traveling salesman problem.

Unit-III Graph Algorithms: Primal-Dual algorithm and its application to shortest path (Dijkstra's algorithm, Floyd-Warshall algorithms), max-flow problem (Ford and Fulkerson labeling algorithms), matching problems (bipartite matching algorithm, non-bipartite matching algorithms, bipartite weighted matching-hungarian method for the assignment problem, non-bipartite weighted matching problem), efficient spanning tree algorithms.

Unit-IV Matroids: Independence Systems and Matroids, Duality, Matroid Intersection.

Readings:

1. Bernhard Korte and Jens Vygen, "**Combinatorial Optimization: Theory and Algorithms (Algorithms and Combinatorics)**", 6th edition, 2018, Springer.
2. Matousek and Gartner, "**Understanding and Using Linear Programming**", 2007, Springer.
3. C.H. Papadimitriou and K.Steiglitz, "**Combinatorial Optimization: Algorithms and complexity**", 1998, Dover Publications.
4. Mokhtar S.Bazaraa, John J. Jarvis and Hanif D. Sherali, "**Linear Programming and Network Flows**", 4th Edition, 2010, Wiley-Blackwell.
5. H.A. Taha, "**Operations Research An Introduction**", 8th edition, 2014, Pearson Education India.

MCSE202: DIGITAL IMAGE PROCESSING[4-1-0]

Course Outcomes:

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

CO1: enhance the quality of an image using various transformations.
CO2: transform an image in spatial domain to frequency domain, and DWT domain.
CO3: apply various morphological operations to an image.
CO4: segment an image using various approaches.
CO5: compress an image.

Syllabus:

Unit-I Introduction: Applications of digital image processing, steps in digital image processing: image acquisition, image sampling and quantization, basic relationships between pixel.

Unit-II Image enhancement in the spatial domain and frequency domain: Gray level transformations, histogram processing, local enhancement, image subtraction, image averaging, spatial filtering: smoothing and sharpening filters, Discrete Fourier transformation, filtering in the frequency domain: smoothening and sharpening filters, image restoration in spatial and frequency domains.

Unit-III Morphological image processing: erosion and dilation, opening and closing, hit-or-miss transformation, some basic morphological algorithms.

Unit-IV Image segmentation: Point, line and edge detection, gradient operator, edge linking and boundary detection, thresholding, region-based segmentation, representation schemes like chain codes, polygonal approximations, boundary segments, skeleton of a region, boundary descriptor, regional descriptor.

Unit-V Introduction to Image Compression: Image compression models, Error free compression techniques, lossy compression techniques, JPEG, MPEG.

Readings:

1. Rafael C. Gonzalez and Richard E.Woods**, Digital Image Processing** (3rd edition), Prentice–Hall of India, 2016.

2. Bernd Jahne**, Digital Image Processing**, (6th edition), Springer, 2005.

3. S. Annadurai and R. Shanmugalakshmi, **Fundamentals of Digital Image Processing**, Pearson Education, 2007.

4. M.A. Joshi, **Digital Image Processing: An Algorithmic Approach** (2nd edition), Prentice-Hall of India.

5. B. Chandra and D.D. Majumder, **Digital Image Processing and Analysis**, Prentice-Hall of India, 2011.

MCSE203: COMPILER DESIGN[4-1-0]

Course Outcomes:

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

CO1: describe how different phases of a compiler work.
CO2: implement top down and bottom up parsing algorithms.
CO3: use compiler tools like lex and yacc for implementing syntax directed translator.

Syllabus:

Unit-I Lexical Analysis: Review of regular languages and finite automata, design of a lexical analyzer generator, context-free grammars.

Unit-II Syntax Analysis: Review of context free grammar, top-down parsing: recursive descent and predictive parsing, LL(k) parsing; bottom-up parsing: LR parsing, handling ambiguous in bottom-up parsers.

Unit-III Syntax directed translation: Top down and bottom up approaches, data types, mixed mode expression; subscripted variables, sequencing statement, subroutines and functions: parameters calling, subroutines with side effects.

Unit-IV Code generation: Machine code generation, machine dependent and machine independent optimization techniques.

Readings:

1. Alfred V. Aho, Ravi Sethi, D. Jeffrey Ulman, Monica S. Lam, **Principles, Techniques and Tools**, Pearson Education India, 2nd edition,, 2013

2. A.V. Aho, M. S. Lam, R. Sethi and J. D. Ullman, **Compilers,** Pearson, 2016.

3. Moder Dick Grune, Kees van Reeuwijk, Henri E. Bal, Ceriel J.H. Jacobs, Springer, 2016.

MCSE204: DATABASE APPLICATIONS [4-1-0]

Course Outcomes:

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

CO1: describe the architecture of a web application.
CO2: describe the issues in query optimization.
CO3: develop a web-based database application incorporating security issues.

Syllabus:

Unit-I Introduction: Review of database design methods: ER modeling and normalization.

Unit-II Database Programming: SQL user-defined data types, collection types; procedures and functions, exception handling, triggers, large objects, bulk loading of data.

Unit-III Authorizations in SQL: System and user privileges, granting and revoking privileges, roles, authorization on views, limitations of SQL authorizations, audit trails.

Unit-IV Web Application Design and Development: Web technologies, web interfaces to databases, digital signatures and digital certificates, performance issues, XML in Databases**.**

Readings:

1. A. Silberschatz, H. Korth and S. Sudarshan, **Database System Concepts** (6th ed.), McGraw Hill, 2010.

2. Loney and Koch, **Oracle 10g The Complete Reference**, Tata McGraw Hill, 2006.

3. J. Morrison, M. Morrison and R. Conrad, **Guide to Oracle 10g**, Thomson Learning, 2005.

4. David Flanagan, **JavaScript: The Definitive Guide**, O'Reilly Media, 6th edition 2011.

5. Marty Hall, Larry Brown, and Yaakov Chaikin, **Core Servlets and Javaserver Pages: Core Technologies, Vol. 2 (2nd ed.)**, Sun Microsystems Press, 2006.

MCSO201: JAVA PROGRAMMING[3-1-0]

Course Outcomes:

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

CO1: appreciate object-oriented concepts – Classes, Objects, Inheritance, Polymorphism – for problem solving.
CO2: handle program exceptions.
CO3: design, implement, document, test, and debug a Java application consisting of multiple classes.
CO4 : handle input/output through files.
CO5 : create Java applications with graphical user interface (GUI).

Syllabus:

Unit-I Review of Programming Concepts: program, identifiers, variables, constants, primitive data types, expressions, control statements, structured data types, arrays, functions.

Unit-II Object Oriented Concepts: Abstraction, encapsulation, objects, classes, methods, constructors, inheritance, polymorphism, static and dynamic binding, overloading, Abstract classes, Interfaces and Packages**.**

Unit-III File Handling: Byte Stream, Character Stream, File I/O Basics, File Operations, Serialization**.**

Unit-IV Exception handling: Throw and Exception, Throw, try and catch Blocks, Multiple Catch Blocks, Finally Clause, Throwable Class, Types of Exceptions, java.lang Exceptions, Built-In Exceptions**.**

Unit-V GUI Design: GUI based I/O, Input and Message Dialog boxes, Swng components, Displaying text and images in windows.

Readings:

1.James Gosling, Bill Joy, Guy L. Steele Jr, Gilad Bracha, Alex Buckley, **The Java Language Specification, Java SE 7 Edition**, Addison-Wesley, 2013.

2. Cay S. Horstmann, Core Java **-** Vol. I – Fundamentals, 10th Edition, Pearson, 2017.

3. Deitel & Deitel, **Java-How to Program** (9th ed.), Pearson Education, 2012.

4. **Richard Johnson,** An Introduction to Java Programming and Object-Oriented Application Development, **Thomson Learning,** 2006.

5. Herbert Schildt, **Java: The Complete Reference**, 10th Edition, McGraw-Hill Education, 2018.

MCSO202: GPU PROGRAMMING[3-1-0]

Course Outcomes:

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

CO1: describe GPU architecture and parallel programming models.
CO2: implement fundamental GPU Algorithms – reduce, scan, and histogram.
CO3: analyse, and figure out portion of programs being parallelizable.
CO4: write efficient parallel algorithm to solve a given problem.
CO5: optimize GPU programs.

Syllabus:

Unit-I Introduction: Introduction to heterogeneous computing, overview of CUDA C/Python, and kernel-based parallel programming.

Unit-II Performance Issues: Memory model for locality, tiling for conserving memory bandwidth, handling boundary conditions, and performance considerations, simple matrix-matrix multiplication in CUDA environment.

Unit-III Introduction to OpenCL: operations such as vector addition using streams.

Unit-IV Applications: Parallel convolution pattern, parallel scan pattern, parallel histogram pattern and atomic operations, data transfer and task parallelism.

Readings:

1. Shane Cook, **CUDA Programming: A Developer's Guide to Parallel Computing with GPUs**, Elsevier; 2014.
2. Norman Matloff, **Parallel Computing for Data Science: With Examples in R, C++ and CUDA**, Chapman & Hall/CRC, 2015.

SEMESTER – III

MCSE301: CYBER SECURITY [4-1-0]

Course Outcomes:

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

CO1 : state the need and scope for cyber laws.
CO2 : enumerate various network attacks, describe their sources, and mechanisms of prevention.
CO3: describe the genesis of SCADA policies and their implementation framework.
CO4: carry out malware analysis using simulations.

Syllabus:

Unit-I Introduction: Cyberspace, Internet, Internet of things, Cyber Crimes, cyber criminals, Cyber security, Cyber Security Threats, Cyber laws and legislation, Law Enforcement Roles and Responses.

Unit-II Network Attacks: Network Threat Vectors, MITM, OWAPS, ARP Spoofing, IP & MAC Spoofing, DNS Attacks, SYN Flooding attacks, UDP ping-pong and fraggle attacks, TCP port scanning and reflection attacks, DoS, DDOS. Network Penetration Testing Threat assessment, Penetration testing tools, Penetration testing, Vulnerability Analysis, Threat matrices, Firewall and IDS/IPS, Wireless networks, Wireless Fidelity (Wi-Fi), Wireless network security protocols, Nmap, Network fingerprinting, BackTrack, Metasploit.

Unit-III Introduction to SCADA (supervisory control and data acquisition): Understanding SCADA security policies, SCADA Physical and Logical Security, Understanding differences between physical and logical security, Define perimeter controls and terms, Define various security zones, Understand communication cyber threats, Understand firewall, architectures.

Unit-IV Introduction Malware Analysis: Static Analysis, Code Review, Dynamic Analysis, Behavioral analysis of malicious executable, Sandbox Technologies, Reverse-engineering malware, Defeat anti-reverse engineering technique, automated analysis, intercepting network connections, Network flow analysis, Malicious Code Analysis, Network analysis, Anti-disassembling techniques, Identifying assembly logic structures with a disassembler.Malware Handling: Malicious Documents and Memory Forensics - Reverse engineering of malicious executable using memory forensic techniques, Analyze malicious Microsoft Office (Word, Excel, PowerPoint) and Adobe PDF documents, Analyzing memory to assess malware characteristics and reconstruct infection artifacts. Using memory forensics to analyze rootkit infections, Legal & Ethical Issues - Reinforce understanding and the application of discipline specific legal and ethical issues, Reverse Engineering Malware (REM) Methodology.

Readings:

1. Peter W. Singer and Allan Friedman Cybersecurity and Cyberwar**, Oxford University Press, 2014.**

2. Jonathan Clough, Principles of Cybercrime**, Cambridge University Press, 24-Sep-2015.**

3. Jie Wang, Zachary A. Kissel, Introduction to Network Security: Theory and Practice**, Wiely 2016.**

4. Michael Bazzell, Open Source Intelligence Techniques: Resources for Searching and Analyzing Online Information**, 2nd edition , CreateSpace Independent Publishing Platform,2014.**

5. Robert Radvanovsky, Jacob Brodsky, Handbook of SCADA/Control Systems Security**, CRC Press, 2013.**

6. Ed Skoudis , Lenny Zeltser, Malware: Fighting Malicious Code, Prentice Hall Series in Computer Networking and Distributed**, 2003.**

7. Michael Sikorski , Andrew Honig, Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software 2012**, No Starch Press, San Francisco.**

MCSE302: GRAPH THEORY[4-0-1]

Course Outcomes:

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

CO1: model problems in different types of basic graphs like trees, bipartite graphs and planar graphs.
CO2: identify special graphs like Euler graphs and Hamiltonian graphs.
CO3: able to identify various forms of connectedness in a graph.
CO4: appreciate different graph-coloring problems and their solutions.
CO5: model simple problems from real life as graph-coloring problems.

Syllabus:

Unit-I Fundamental Concepts: Definitions, examples of problems in graph theory, adjacency and incidence matrices, isomorphism, paths, walks, cycles, components, cut-edges, cut-vertices, bipartite graphs, Eulerian graphs, vertex degrees, reconstruction conjecture, extremal problems, degree sequences, directed graphs, de Bruijn cycles, Orientations and tournaments.

Unit-II Trees: Trees and forests, characterizations of trees, spanning trees, radius and diameter, enumeration of trees, Cayley’s formula, Prüfer code, counting spanning trees, deletion-contraction, matrix tree theorem, graceful labelling, minimum spanning trees (Kruskal’s algorithm), shortest paths (Dijkstra’s algorithm).

Unit-III Matching and Covers: Matchings, maximal and maximum matchings, M-augmenting paths, Hall's theorem and consequences, Min-max theorems, maximum matchings and vertex covers, independent sets and edge covers,Connectivity, vertex cuts, Edge-connectivity.

Unit-IV Connectivity and Paths: Blocks, k-connected graphs, Menger's theorem, line graphs, network flow problems, flows and source/sink cuts, Ford-Fulkerson algorithm, Max-flow min-cut theorem.

Unit-V Graph Coloring: Vertex colorings, bounds on chromatic numbers, Chromatic numbers of graphs constructed from smaller graphs, chromatic polynomials, properties of the chromatic polynomial, the deletion-contraction recurrence.

Unit-VI Planar Graphs: Planar graphs, Euler's formula, Kuratowski's theorem, five and four color theorems.

Readings:

1. Douglas B West, **Introduction to Graph Theory**, II Edition, 2017, Pearson.

2. Gary Chartrand and Ping Zhang, **Introduction to Graph Theory**, 2017, Tata McGraw Hill.

3. Jonathan L. Gross and Jay Yellen, **Graph Theory and Its Applications**, 2nd Edition, 2005, Chapman Hall (CRC).

MCSE303: NETWORK SCIENCE[4-1-0]

Course Outcomes:

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

CO1: appreciate ubiquity of graph data model and importance of graph theoretic concepts in social network analysis.
CO2: compute structural features of a network and their impact on functions
CO3: generate networks using theoretical graph generation models.
CO4: identify community structures in networks.

Syllabus:

Unit-I Introduction: Introduction to complex systems and networks, Modelling of complex systems, Review of graph theory.

Unit-II Network properties: Local and Global properties, Clustering coefficient, All-pair-shortest-path based properties, Centrality measures for directed and undirected networks.

Unit-III Graph models: Random graph model, Small world network model, Barabasi-Albert (preferential attachment) network model.

Unit-IV Community structure in networks: Communities and community detection in networks, Hierarchical algorithms for community detection, Modularity based community detection algorithms, Label Propagation algorithm.

Readings:

1. Mohammed J. Zaki, Wagner Meira Jr.; **Data Mining and Analysis: Fundamental Concepts and Algorithms**, Cambridge University Press, 2014.

2. Albert Barabasi, **Network Science** , Cambridge University Press, 2016

3. M.E. J. Newman, **Networks: An Introduction**, Oxford University Press, 2010.

4. David Easley and Jon Kleinberg, **Networks, Crowds, and Markets: Reasoning About a Highly Connected World**, Cambridge University Press, 2010.

MCSE304: DEEP LEARNING[4-1-0]

Course Outcomes:

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

CO1: describe the feedforward and deep networks.
CO2: design single and multi-layer feed-forward deep networks and tune various hyper-parameters.
CO3: analyse performance of deep networks.

Syllabus:

Unit-I Introduction: Historical context and motivation for deep learning; basic supervised classification task, optimizing logistic classifier using gradient descent, stochastic gradient descent, momentum, and adaptive sub-gradient method.

Unit-II Neural Networks: Feedforward neural networks, deep networks, regularizing a deep network, model exploration, and hyperparameter tuning.

Unit-III Convolution Neural Networks: Introduction to convolution neural networks: stacking, striding and pooling, applications like image, and text classification.

Unit-IV Sequence Modeling: Recurrent Nets: Unfolding computational graphs, recurrent neural networks (RNNs), bidirectional RNNs, encoder-decoder sequence to sequence architectures, deep recurrent networks.

Unit-V Autoencoders: Undercomplete autoencoders, regularized autoencoders, sparse autoencoders, denoising autoencoders, representational power, layer, size, and depth of autoencoders, stochastic encoders and decoders.

Unit-VI Structuring Machine Learning Projects: Orthogonalization, evaluation metrics, train/dev/test distributions, size of the dev and test sets, cleaning up incorrectly labeled data, bias and variance with mismatched data distributions, transfer learning, multi-task learning.

Readings :

1. Ian Goodfellow, **Deep Learning**, MIT Press, 2016.

2. Jeff Heaton, **Deep Learning and Neural Networks**, Heaton Research Inc, 2015.

3. Mindy L Hall, Deep Learning, VDM Verlag, 2011.

4. Li Deng (Author), Dong Yu, **Deep Learning: Methods and Applications (Foundations and Trends in Signal Processing)**, Now Publishers Inc, 2009.

MCSE305: NEURAL NETWORKS[4-1-0]

Course Outcomes:

On completion of the course, students will be able to:

CO1: describe the role of neural networks in engineering, artificial intelligence, and cognitive modelling.
CO2: design single and multi-layer feed-forward neural networks for practical applications,
CO3: analyse performance of neural networks and tune various hyper-parameters.

Syllabus:

Unit-I Introduction: Neuron as basic unit of neurobiology, perceptron as a model of learning, perceptron convergence theorem, batch perceptron learning algorithm, relation between perceptron and Bayesian learner for a Gaussian environment; linear regression model, maximum a posteriori (MAP) estimation of the parameter vector, least mean squares algorithm.

Unit-II Multilayer Perceptrons: Backpropagation algorithm, batch learning and online learning, adaptive control of learning rate; estimating regularization parameter: Tikhonov's regularization theory, complexity regularization, and network pruning.

Unit-III Kernel Methods and Support Vector Machines: Separability of patterns, interpolation problem, radial basis function (RBF) networks, support vector machines.

Unit-IV Introduction to Hopefield networks, Boltzmann machines, restricted Boltzmann machines.

Readings:

1. Simon O. Haykin, **Neural Networks and Learning Machines**, Pearson Education, 2016

2. C. M. Bishop, **Pattern Recognition and Machine Learning**, Springer, 2010.

3. C. Bishop, **Neural Networks and Machine Learning**, Springer, 1998.

MCSE306: MODELING AND SIMULATION[4-1-0]

Course Outcomes:

On completion of the course, students will be able to:

CO1: acquire basic understanding of systems, models and simulation and familiarity with Modeling and simulation tools.
CO2: perform parameter estimation and test goodness of fit in a model.
CO3: check system's stability, observability and controllability.
CO4: apply statistical methods in modeling and simulation.
CO5: apply modeling and simulation to real world problems.

Syllabus:

Unit-I Systems, Models and Simulation study: Natural and Artificial Systems, Complex Systems, Definition and types of model, Mathematical models, Cyber-physical systems and its modeling, Network models, Steps in simulation study, Advantage and disadvantage of simulation**.**

Unit-II Random Numbers: True and pseudo random numbers, Properties of random numbers, Generation of pseudo random numbers, Tests for randomness, Random variate generation using inverse transformation, Direct transformation, Convolution method and Acceptance-rejection method**.**

Unit-III Design and Analysis of simulation experiments: Data collection, Identifying distributions with data, Parameter estimation, Goodness of fit tests, Selecting input models without data, Multivariate and time series input models, Verification and validation of models, Steady-state simulation, Terminating simulation, Confidence interval estimation, Output analysis for steady state simulation, Stochastic simulation**.**

Unit-IV Control Systems: Laplace transform, Transfer functions, State- space models, Order of systems, z-transform, Feedback systems, Stability, Observability, Controllability.

Unit-V Statistical Models in Simulation: Common discrete and continuous distributions, Poisson process, Markov chain, Empirical distributions, Queuing systems, Transient and steady-state behavior, performance, Network of queues**.**

Unit-VI Modeling and Simulation tools: Open Modelica, Netlogo, Python modules for modeling and simulation, GPSS**.**

Readings:

1. Ross, S., **Simulation**, 5th Edition, Academic Press, 2012.

2. Frank L. Severance, **System Modeling And Simulation: An Introduction**, Wiley, 2001**.**

3. Jerry Banks, John S. Carson II, Barry L. Nelson, Devid M. Nicol, P. Shahabudeen: **Discrete-Event system simulation**, 5th Edition, 2009**.**

4. Geoffrey Gordon: **System Simulation**, 2nd Edition, 2002.

5. A.M. Law and W.D. Kelton: **Simulation and Modeling and analysis**, 5th Edition, 2015.

MCSE307: COMPUTATIONAL INTELLIGENCE[4-1-0]

Course Outcomes:

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

CO1: understand, analyse, and solve computational problems.
CO2: select a suitable programming construct and inbuilt data structure for a situation.
CO3: develop, document, and debug modular python programs.
CO4: use classes, objects, and files in application programs.

Syllabus:

Unit-I Introduction: Computational intelligence as a computing paradigm.

Unit-II Rough Sets: Introduction, set approximation, decision tables.

Unit-III Fuzzy Logic Systems: Notion of fuzziness, operations on fuzzy sets, T-norms and other aggregation operators, basics of approximate reasoning, compositional rule of inference, fuzzy rule based systems, schemes of fuzzification, inferencing, defuzzification, fuzzy clustering, fuzzy rule based classifier.

Unit-IV Type-2 Fuzzy Sets: Notion of uncertainty of membership in a fuzzy set, foot print of uncertainty, embedded fuzzy sets, operations on type-2 fuzzy sets, type-2 fuzzy relations, type reduction, type-2 fuzzy inference system.

Unit-V Neural Networks: Review of basic concepts of a neuron, perceptron, back propagation, feedforward neural networks.
Multilayer perceptron – Batch learning and online learning, adaptive control of learning rate, estimating regularization parameter.

Unit-VI Evolutionary Computation: Genetic operators, building block hypothesis, evolution of structure, genetic algorithms based on tree and linear graphs, multi-objective optimization, applications in science and engineering.

Readings:

1. Russell Eberhart, **Computational Intelligence: Principles, Techniques and Applications**, Elsevier, 2012.

2. Leszek Rutkowski, **Computational Intelligence: Methods and Techniques**, Springer 2008.

MCSE308: PARALLEL AND DISTRIBUTED COMPUTING[4-1-0]

Course Outcomes:

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

CO1: describe architectures for parallel and distributed systems.
CO2: develop elementary parallel algorithms.
CO3: develop an application involving synchronization of communicating distributed process.

Unit-I Parallel Computing: Trends in microprocessor architectures, memory system performance, dichotomy of parallel computing platforms, physical organization of parallel platforms, communication costs in parallel machines, SIMD versus MIMD architectures, global versus distributed memory, the PRAM shared-memory model, distributed-memory or graph models, basic algorithms for some simple architectures: linear array, binary tree, 2D mesh with shared variables.

Unit II Distributed Computing Architectures: Characteristics and goals of distributed computing, architectural styles: centralized, decentralized, and hybrid architectures, layered, object-based and service oriented, resource-based, publish-subscribe architectures, middleware organization: wrappers, interceptors, and modifiable middleware, system architecture, example architectures: network file system and web.

Unit-III Distributed Processes and Communication in Distributed Systems: Threads in distributed systems, principle of virtualization, clients: network user interfaces and client-side software for distribution transparency; servers: design issues, object servers, server clusters; code migration; Layered protocols, remote procedure call: RPC operation, parameter passing; message-oriented communication: transient messaging with sockets, message-oriented persistent communication; multicast communication: application-level tree-based multicasting, flooding-based multicasting, gossip-based data dissemination.

Unit IV Naming and Coordination in Distributed Systems: Names, identifiers, and addresses, flat naming, Structured naming, and attribute-based naming; coordination: clock synchronization, logical clocks, mutual exclusion: centralized, distributed, token-ring, and decentralized algorithms; election algorithms, location systems, distributed event matching, gossip-based coordination.

Unit-V Consistency and replication, Fault Tolerance, and Security: Introduction to consistency models and protocols, fault tolerance, and security issues in distributed systems

Readings:

1. Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar, **Introduction to Parallel Computing**, 2nd edition, 2003, Addison-Wesley.

2. B. Parhami, **Introduction to Parallel Processing: Algorithms and Architectures**, Plenum, 1999, Springer.

3. M. van Steen, A. S. Tanenbaum, **Distributed Systems**, CreateSpace Independent Publishing Platform, 2017.

MCSE309: SOFTWARE QUALITY ASSURANCE AND TESTING[4-1-0]

Course Outcomes:

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

CO1: understand quality management processes.
CO2: understand the importance of standards in the quality management process and role of SQA function in an organization.
CO3: gain knowledge of statistical methods and process for software quality assurance.
CO4: understand the need and purpose of software testing.
CO5: model the quantitative quality evaluation of the software products.

Syllabus:

Unit-I Introduction: Concept of Software quality, product and process quality, software quality metrics, quality control and total quality management, quality tools and techniques, quality standards, defect management for quality and improvement.

Unit-II Designing software quality assurance system: Statistical methods in quality assurance, fundamentals of statistical process control, process capability, Six-sigma quality.

Unit-III Testing: Test strategies, test planning, functional testing, stability testing and debugging techniques.

Unit-IV Reliability: Basic concepts, reliability measurements, predictions and management.

Readings:

1. N.S. Godbole, **Software Quality Assurance: Principles and Practice for the New Paradigm (2nd Ed.**), Narosa Publishing, 2017.

2.G. Gordon Schulmeyer (4th eds.), **Handbook of Software Quality Assurance Artech House**,Inc, 2008.

3. G. O'Regan, **A Practical Approach to Software Quality**, Springer Verlag, 2002.

4. Daniel Galin, **Quality Assurance: From theory to implementation,** Pearson Education Ltd., 2004.

5. S.H. Kan, **Metrics and Models in Software Quality Engineering (2nd ed.)**, Pearson Education Inc., 2003.

6. J.D. McGregor and D.A. Sykes, **A Practical Guide to Testing**, Addison-Wesley, 2001.

7. Glenford J. Myers, **The Art of Software Testing (2nd ed.)**, John Wiley, 2004.

8. D. Graham, E.V. Veenendaal, I. Evans and R. Black, **Foundations of Software Testing**,Thomson Learning, 2007.

MCSE310: TEXT ANALYTICS [4-1-0]

Course Outcomes:

On completion of the course, the student will be able to

CO1: identify relevance of text to the information needs of diverse individuals, communities and organizations.
CO2: select appropriate technique to automatically process the text.
CO3: use open source text analytic tools to perform various text processing and analysis tasks.
CO4: develop simple text analysis tools.
CO5: design a project for text analysis suitable for a specific problem.

Syllabus:

Unit-I Introduction : Introduction to Natural Language Processing (NLP) and Text Analytics, Tokenization, Part-of-speech tagging, Stemming and lemmatization, NLP toolkits, Indexing text using inverted file index, Posting lists , Information retrieval, Authorship attribution, Sentiment analysis, Named-entity recognition, Document summarization.

Unit-II Document representation and Language models : Representation of the unstructured text documents, Heaps' and Zipf's Laws, Boolean vector representation, Bag-of-words model, tf-idf model, Term weighting , Scoring and ranking, Vector space model, Visualization techniques, Unigrams, bi-grams and n-grams, language models.

Unit-III Text categorization: Supervised text categorization algorithms, Naive Bayes, k Nearest Neighbor (kNN), Logistic Regression.

Unit-IV Text clustering: Clustering structure of a corpus of text documents, Hierarchical clustering, Centroid-based clustering. Topic Modeling- Latent Semantic Indexing (LSI).

Readings:

1. Ricardo Baeza – Yates, Berthier Ribeiro – Neto, **Modern Information Retrieval: The concepts and Technology behind Search,** (ACM Press Books), Second Edition, 2011.

2. Christopher D. Manning, Prabhakar Raghavan, Hinrich Schutze, **Introduction to Information Retrieval**, Cambridge University Press, First South Asian Edition, 2008.

3. Steven Struhl, **Practical Text Analytics: Interpreting Text and Unstructured Data for Business Intelligence**, Kogan Page, 2015.

4. Matthew A. Russell, **Mining the Social Web,** O'Reilly Media, 2013.

MCSE311: MULTI-AGENT SYSTEMS [4-1-0]

Course Outcomes:

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

CO1: define "Agent" with the description of intelligent and rational agents.
CO2: explain agent types, structure and their interactions.
CO3: identify problems that are amenable to solution by specific agents.
CO4: explain multiagent interactions with their reaching agreements.
CO5: demonstrate communication language used by the agents for their coordination and coherence.
CO6: appreciate the utility of different type of AI agents.
CO7: explain the cooperative distributed problem solving techniques used by agents with task and result sharing.

Syllabus:

Unit-I Intelligent Agents: Environments, Intelligent Agents, Agents and Objects, Agents and Expert Systems, Agents as Intentional Systems, Abstract Architectures for Intelligent Agents, Purely Reactive Agents, Perception, Agents with State, How to Tell an Agent What to Do, Utility Functions.

Unit-II Practical Reasoning Agents: Practical Reasoning, Means-Ends Reasoning, The BlocksWorld, Implementing a Practical Reasoning Agent, Commitment to Ends and Means, The Procedural Reasoning System.

Unit-III Reactive and Hybrid Agents: Brooks and the Subsumption Architecture, The Limitations of Reactive Agents, Hybrid Agents.

Unit-IV Multiagent Interactions: Utilities and Preferences, Multiagent Encounters, Dominant Strategies and Nash Equilibria, Competitive and Zero-Sum Interactions, The Prisoner'sDilemma.

Unit-V Reaching Agreements: Mechanism Design, Auctions, Negotiation, Task-Oriented Domains, Worth-Oriented Domains, Argumentation.

Unit-VI Communication: Speech Acts, Agent Communication Languages, KIF, KQML, The FIPA Agent Communication Languages, Ontologies for Agent Communication, Coordination Languages.

Unit-VII Working Together: Cooperative Distributed Problem Solving, Coherence and Coordination, Task Sharing and Result Sharing, Task Sharing in the Contract Net, Result Sharing, Handling Inconsistency, Coordination, Multiagent Planning and Synchronisation.

Readings:

1. Michael Wooldridge, **An Introduction to MultiAgent Systems**, 2nd edition, John Wiley & Sons, 2009.

2. G. Weiss**. Multiagent Systems--A Modern Approach to Distributed Artificial Intelligence,** 2nd edition, MIT Press, Cambridge, MA, 2013.

3. S. Russell and P. Norvig, **Artificial Intelligence: A Modern Approach, 3rd edition**, Pearson Education, 2015.

4. Rafael H. Bordini, Jomi Fred Hübner, Michael Wooldridge, **Programming Multi-Agent Systems in AgentSpeak using Jason,** Wiley Series in Agent Technology, John Wiley & Sons, 2007.

5. Fabio Luigi Bellifemine, Giovanni Caire, Dominic Greenwood, **Developing Multi-Agent Systems with JADE**, Wiley Series in Agent Technology, John Wiley & Sons, 2007.

MCSE312: STEGANOGRAPHY AND DIGITAL WATERMARKING [4-1-0]

Course Outcomes:

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

CO1: implement algorithms for steganographic and watermarking operations.
CO2: evaluate steganographic and watermarking algorithms for robustness**.**
CO3: describe various finger printing techniques.

Syllabus:

Unit-I Information hiding: Introduction, Background, and Applications of Information hiding: Data hiding, applications of data hiding.

Unit-II Steganography: Frameworks of secret communication, Security of steganography systems, Information hiding in noisy data, Adaptive and non-adaptive algorithms, Active and malicious attackers, Information hiding in written text, Invisible communication.

Unit-III Data hiding in still images : LSB encoding, BPCS steganography, Lossless data hiding, Data hiding by quantification, Patchwork, Transform domain methods, Robust data hiding in JPEG images, frequency domain watermarking Detecting malicious tempering, Robust wavelet based watermarking, Kundur-Hatzinakos watermarking, Data hiding in binary images, Zhao-koch method, Wu-Lee method, CPT method, TP method, Data hiding in fax images.

Unit-IV Watermarking: Introduction, Watermarking principals, Applications, Requirements and algorithmic design issues, Evaluation and standards of watermarking.

Unit-V Fingerprinting: Introduction, Terminology and requirements, Classifications, Research history, fingerprinting schemes, Statistical fingerprinting, and Collusion-secure fingerprinting.

Readings:

1. I.J.Cox, M.L.Miller, J.A.Bloom, J.Fridrich, T.Kalker, **Digital Watermarking and Stegonagraphy,** Morgan Kaufman 2008.

2. F.Y.Shih, **Digital Watermarking and Stegonagraphy Fundamentals and Techniques,** CRC press 2008.

3. Stefon Katzeubeisser, F.A.Petitolos, **Information Hiding Techniques for Stegonagraphy and digital watermarking**, Aatech House London 2008.

MCSE 313: NP COMPLETENESS AND APPROXIMATION ALGORITHMS [4-0-1]

Course Outcomes:

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

CO1: appreciate that certain problems are too hard to admit fast solutions and be able to prove their hardness.
CO2: explain what an approximation algorithm is, and the advantage of using approximation algorithms.
CO3: apply techniques to design approximation algorithms.
CO4: analyse the approximation factor of an algorithm.

Syllabus:

Unit-I Introduction to Classes P, NP, NP-Hard, NP Complete: Verifiability and Reduction**.**

Unit-II Proving NP-Completeness from first principle: Satisfiability Problem (SAT), 3SAT.

Unit-III Graph Problems: Clique, Vertex Cover, Independent Set, Hamiltonian Cycle Problem, Travelling Salesman Problem, Graph Partitioning, Subgraph problem, Graph Isomorphism, Graph Coloring.

Unit-IV Sets and Partitions: Set partition, Set Cover, Subset Sum and Knapsack Problem.

Unit-V Techniques to design approximation algorithms: LP-rounding, Primal-Dual, Dual Fitting, Greedy, Local Search.

Readings:

1. J. Kleinberg and E.Tardos, **Algorithm Design**, 1st Edition 2013., Pearson Education India,

2. 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.

3. Vijay V. Vazirani, **Approximation Algorithms**, 2013, Springer.

4. David P. Williamson and David B. Shmoys, **The Design of Approximation Algorithms**, 2011, Cambridge University Press.

5. Part of the course will be covered by research papers.

MCSO301: DATA SCIENCE[3-1-0]

Course Outcomes:

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

CO1: demonstrate proficiency with statistical analysis of data.
CO2: develop the ability to build and assess data-based models.
CO3: execute statistical analyses and interpret outcomes.
CO4: apply data science concepts and methods to solve problems in real-world contexts and will communicate these solutions effectively.

Syllabus:

Unit-I Introduction: Introduction data acquisition, data preprocessing techniques including data cleaning, selection, integration, transformation and reduction, data mining, interpretation;

Unit-II Statistical data modeling: Review of basic probability theory and distributions, correlation coefficient, linear regression, statistical inference, exploratory data analysis and visualization.

Unit-III Predictive modeling: Introduction to predictive modeling , decision tree, nearest neighbor classifier and naïve Baye's classifier, classification performance evaluation and model selection.

Unit-IV Descriptive Modeling: Introduction to clustering, partitional, hierarchical, and density-based clustering (k-means, agglomerative, and DBSCAN), outlier detection, clustering performance evaluation.

Unit-V Association Rule Mining: Introduction to frequent pattern mining and association rule mining, Apriori algorithm, measures for evaluating the association patterns.

Unit-VI Text Mining: Introduction of the vector space model for document representation, term frequency-inverse document frequency (tf-idf) approach for term weighting, proximity measures for document comparison, document clustering and text classification.

Readings:

1. W. McKinney, Python for Data Analysis: Data Wrangling with Pandas, NumPy and iPython, 2^{nd} Ed., O'Reilly, 2017.

2. P. Tan, M. Steinbach, A Karpatne, and V. Kumar, **Introduction to Data Mining**, 2^{nd} Ed., Pearson Education, 2018.

3. G James, D Witten, T Hastie and R Tibshirani **An Introduction to Statistical Learning with Applications in R,** Springer Texts in Statistics, Springer, 2013.

4. G. Grolemund, H. Wickham, **R for Data Science**, 1^{st} Ed., O'Reilly, 2017.

MCSO302: E-COMMERCE [3-1-0]

Course Outcomes:

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

CO1: understand the basic business management concepts and how E-commerce is affecting business enterprises, governments, consumers and people in general.
CO2: understand technical concepts, and privacy relating to E-commerce.
CO3: understand distributed environment, client-server architecture and middleware for the purpose of the development of E-commerce applications.
CO4: understand the infrastructure and components of E-commerce.
CO5: describe various electronic payment systems.
CO6: understand ethical and legal issues relating to E-commerce.

Syllabus:

Unit-I Introduction: Introduction to networking technologies, Network Protocols, Client Server architecture, Two-tier architecture, Three-tier architecture, MVC architecture.

Unit-II Building Blocks of E-Commerce: Software technologies for building E-Commerce applications, Distributed Objects, Remote Method Invocation (RMI), introduction to CORBA, Web services.

Unit-III Security of E-Commerce Transactions: Review of cryptographic tools, authentication, signatures, observers, anonymity, privacy, traceability, key certification, management and escrow.

Unit-IV Payment Protocols and Standards: Smart card, e-cash, e-wallet, electronic money and electronic payment systems, crypto-currency payments, business models for e-commerce, electronic marketplaces, auctions and other market mechanisms, design of auctions, content optimization algorithms for marketplaces, multi-agent systems.

Unit-V Global E-Commerce and Law: Cyber law in India, comparative evaluation of Cyber laws of certain countries.

Readings:

1. P.T. Joseph, S.J., **E-Commerce: An Indian Perspective** (5th ed.), Prentice-Hall of India, 2015.
2. Efraim Turban, Jae Kyu Lee, Dave Klng, Judy McKay, Peter Marshall, **Electronic Commerce: A Managerial Perspective** (5th ed.), Pearson, 2008.
3. M.L. Liu, **Distributed Computing: Principles and Applications**, Pearson, 2004.
4. Stuart Jacobs, **Engineering Information Security**, IEEE Press, Wiley, 2011.
5. R. Orfali and Dan Harkey, **Client/Server Programming with Java and CORBA** (2nd ed.), John Wiley & sons, 1998.
6. Michael Wooldridge, **An Introduction to MultiAgent Systems (**2nd ed.), John Wiley & Sons, 2009.