B.Tech.

IN

COMPUTER SCIENCE AND ENGINEERING

CURRICULUM
2023

तमसो मा ज्योतिर्गमय

Department of Computer Science and Engineering
NATIONAL INSTITUTE OF TECHNOLOGY CALICUT
Kozhikode - 673601, KERALA, INDIA

Program Educational Objectives (PEOs) of B.Tech. in Computer Science and Engineering

PEO1	Graduates shall have sound knowledge regarding the fundamental principles and techniques in the discipline of Computer Science and Engineering.
PEO2	Graduates shall have the ability to specify, design, develop and maintain reliable and efficient software.
PEO3	Graduates shall have the necessary communication and management skills and ethical values to become competent professionals.

Programme Outcomes (POs) and Programme Specific Outcomes (PSOs)
of
B.Tech. in Computer Science and Engineering

PO1	**Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2	**Problem analysis**: Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	**Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	**Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	**Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.
PO6	**The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO7	**Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO8	**Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO9	**Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	**Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO11	**Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	**Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PSO1	Analyze computational problems and design effective and efficient algorithmic solutions.
PSO2	Write effective code for implementing algorithmic solutions and use available software tools for the design of efficient software solutions.

CURRICULUM

Total credits for completing B.Tech. in Computer Science and Engineering is 150

COURSE CATEGORIES AND CREDIT REQUIREMENTS:

The structure of B.Tech. programmes shall have the following Course Categories:

Sl. No.	Course Category	Number of Courses	Minimum Credits
1.	Institute Core (IC)	8	22
2.	Program Core (PC) and Program Electives (PE)	24-25	82
3.	Open Electives (OE)	8	24
4.	Institute Electives (IE) (Entrepreneurship Innovation (EI) + Digital / Automation Technologies (DA) + Humanities, Social Science, Management (HM))	6	18
5.	Activity Credits (AC)	--	4

COURSE REQUIREMENTS

The effort to be put in by the student is indicated in the tables below:
L: Lecture (One unit is of 50 minute duration)
T: Tutorial (One unit is of 50 minute duration)
P: Practical (One unit is of one hour duration)
O: Outside the class effort / self-study (One unit is of one hour duration)

1. INSTITUTE CORE (IC)

a) Mathematics

Sl. No.	Course Code	Course Title	L	T	P	O	Credits
1.	MA1002E	Mathematics I	3	1*	0	5	3
2.	MA1012E	Mathematics II	3	1*	0	5	3
3.	MA2002E	Mathematics III	3	1*	0	5	3
4.	MA2012E	Mathematics IV	3	1*	0	5	3
		Total	**12**	**4***	**0**	**20**	**12**

*Optional for Students (can be replaced by self-study)

b) Basic Sciences

Sl. No.	Course Code	Course Title	L	T	P	O	Credits
1.	PH1001E	Physics of Materials	3	0	0	6	3
2.	BT1001E	Biology for Engineers	3	0	0	6	3
		Total					**6**

c) Professional Communication and Professional Ethics

Sl. No.	Course Code	Course Title	L	T	P	O	Credits
1.	MS1001E	Professional Communication	3	0	0	6	3
2.	CS2019E	Professional Ethics	1	0	0	2	1
		Total	**4**	**0**	**0**	**8**	**4**

2A. PROGRAMME CORE (PC)

Sl. No.	Course Code	Course Title	L	T	P	O	Credits
1	CS1001E	Computer Programming	3	0	0	6	3
2	CS1002E	Introduction to Computing Science	3	0	0	6	3
3	CS1003E	Discrete Structures I	3	0	0	6	3
4	CS1091E	Programming Laboratory	0	0	3	3	2
5	CS1011E	Program Design	3	0	0	6	3
6	CS1012E	Logic Design	3	0	0	6	3
7	CS1013E	Discrete Structures II	3	0	0	6	3
8	CS1092E	Program Design Laboratory	1	0	3	5	3
9	CS2001E	Data Structures and Algorithms	3	1	2	6	4
10	CS2002E	Computer Organization	3	1	2	6	4
11	CS2091E	Data Structures and Algorithms Laboratory	1	0	3	5	3
12	CS2092E	Hardware Laboratory	1	0	3	5	3
13	CS2011E	Database Management System	3	1	2	6	4
14	CS2012E	Operating Systems	3	1	2	6	4
15	CS2013E	Theory of Computation	3	0	0	6	3
16	CS3001E	Computer Networks	3	1	2	6	4
17	CS3002E	Compiler Design	3	1	2	6	4
18	CS3003E	Design and Analysis of Algorithms	3	1	2	6	4
19	CS3011E	Software Engineering	3	1	2	6	4
20	CS3012E	Artificial Intelligence	3	1	2	6	4
21	CS3099E	Project	0	0	0	9	3
22	CS4097E	Summer Internship	-	-	-	*	2
		Total	**-**	**-**	**-**	**-**	**73**

* *Decided by the organisation in which the internship is done*

2B. LIST OF ELECTIVES

Following courses may be credited under the categories mentioned in the table below.

Sl. No.	Course Code	Course Title	L	T	P	O	Credits	Categories			
								PE	EI	DA	HM
1.	CS4021E	Logic for Computer Science	3	0	0	6	3	Y	N	Y	N
2.	CS4022E	Program Analysis	3	0	0	6	3	Y	N	Y	N
3.	CS4023E	Formal Semantics	3	0	0	6	3	Y	N	Y	N
4.	CS4024E	Computational Complexity	3	0	0	6	3	Y	N	Y	N
5	CS4025E	Formal Verification	3	0	0	6	3	Y	N	Y	N
6	CS4026E	Vehicular Networks: Theory to Practice	3	0	0	6	3	Y	N	Y	N
7	CS4027E	Computational Geometry	3	0	0	6	3	Y	N	Y	N
8	CS4029E	Principles of Programming Languages	3	0	0	6	3	Y	N	Y	N
9	CS4030E	Foundations of Programming	3	0	0	6	3	Y	N	Y	N
10	CS4031E	Network Security	3	0	0	6	3	Y	N	Y	N
11	CS4032E	Computer Security	3	0	0	6	3	Y	N	Y	N
12	CS4033E	Quantum Computation	3	0	0	6	3	Y	N	Y	N
13	CS4034E	Advanced Computer Networks	3	0	0	6	3	Y	N	Y	N
14	CS4035E	Probabilistic Methods in Combinatorics	3	0	0	6	3	Y	N	Y	N
15	CS4036E	Algorithms in Optimization	3	0	0	6	3	Y	N	Y	N
16	CS4037E	Algorithmic Graph Theory	3	0	0	6	3	Y	N	Y	N
17	CS4038E	Computational Algebra	3	0	0	6	3	Y	N	Y	N
18	CS4039E	Computer Architecture	3	0	0	6	3	Y	N	Y	N
19	CS4040E	Mathematical Foundations of Machine Learning	3	0	0	6	3	Y	N	Y	N
20	CS4041E	Introduction to Machine Learning	3	0	0	6	3	Y	N	Y	N
21	CS4042E	Randomized Algorithms	3	0	0	6	3	Y	N	Y	N
22	CS4043E	Introduction to Parameterized Algorithms	3	0	0	6	3	Y	N	Y	N

23	CS4044E	Introduction to Parameterized Complexity Theory	3	0	0	6	3	Y	N	Y	N
24	CS4045E	Image Processing	3	0	0	6	3	Y	N	Y	N
25	CS4046E	Deep Learning for Computer Vision	3	0	0	6	3	Y	N	Y	N
26	CS4047E	Advanced Computer Architecture and Security	3	0	0	6	3	Y	N	Y	N
27	CS4048E	Cloud Computing	3	0	0	6	3	Y	N	Y	N
28	CS4049E	Distributed Computing	3	0	0	6	3	Y	N	Y	N
29	CS4050E	Natural Language Processing	3	0	0	6	3	Y	N	Y	N
30	CS4051E	Introduction to Bioinformatics	3	0	0	6	3	Y	N	Y	N
31	CS4052E	Number Theory and Cryptography	3	0	0	6	3	Y	N	Y	N
32	CS4053E	Data Mining	3	0	0	6	3	Y	N	Y	N
33	CS4054E	Embedded Systems	3	0	0	6	3	Y	N	Y	N
34	CS4055E	Object Oriented Systems	3	0	0	6	3	Y	N	Y	N
35	CS4056E	Approximation Algorithms	3	0	0	6	3	Y	N	Y	N
36	CS4057E	Data Privacy	3	0	0	6	3	Y	N	Y	N
37	CS4058E	Coding Theory	3	0	0	6	3	Y	N	Y	N
38	CS4059E	Term Paper	3	0	0	6	3	Y	N	Y	N
39	CS4080E	Operating Systems Laboratory	1	0	3	5	3	Y	N	Y	N
40	CS4081E	Compiler Laboratory	1	0	3	5	3	Y	N	Y	N
41	CS4082E	Digital Design Laboratory	1	0	3	5	3	Y	N	Y	N
42	CS4083E	Database System Design Laboratory	1	0	3	5	3	Y	N	Y	N
43	CS4084E	Networks Laboratory	1	0	3	5	3	Y	N	Y	N
44	CS4085E	Software Engineering Laboratory	1	0	3	5	3	Y	N	Y	N
45	CS4086E	Systems Programming Laboratory	1	0	3	5	3	Y	N	Y	N
46	CS4087E	Computer Security Laboratory	1	0	3	5	3	Y	N	Y	N
47	CS4088E	Object Oriented Systems Laboratory	1	0	3	5	3	Y	N	Y	N
48	CS4089E	Machine Learning Laboratory	1	0	3	5	3	Y	N	Y	N

49	CS4090E	Image Processing Laboratory	1	0	3	5	3	Y	N	Y	N
50	CS4091E	Processor Design Lab	1	0	3	5	3	Y	N	Y	N
51	CS4098E	Project	0	0	0	9	3	Y	N	N	N
52	CS4099E	Project	0	0	0	18*	6	Y	N	N	N
53	CS4101E[#]	Deep Learning: Algorithms and Architecture	3	0	0	6	3	Y	N	Y	N
54	CS4102E[#]	AI for Healthcare	3	0	0	6	3	Y	N	Y	N
55	CS4201E[##]	Cyber-Physical Systems: Modelling and Simulation	3	0	0	6	3	Y	N	Y	N
56	CS4202E[##]	Foundations of IoT and M2M Communications	3	0	0	6	3	Y	N	Y	N

** Decided by the organisation in which the project is done, if carried out outside the Institute.*
# Introduced as part of the Minor program on Artificial Intelligence and Machine Learning
## Introduced as part of the Minor program on Cyber-Physical Systems

3. OPEN ELECTIVES (OE)

Courses offered by other Departments / Schools / Centres or Approved Online Platforms, with a limit on the maximum number of courses from such platforms specified as per B.Tech. Ordinances and Regulations. In addition, all PE courses except Projects offered by CSE shall be permitted to be included in this category for students of B.Tech. CSE.

4. INSTITUTE ELECTIVES (IE)

In case of the Institute Electives, courses in the appropriate categories offered by other departments/schools/centres also can be credited instead of the courses offered by the Computer Science & Engineering Department, subject to the approval from the Course Faculty and Faculty Advisor.

a) Entrepreneurship / Innovation Basket (EI):

Courses proposed by the Departments/Schools/Centres and approved by Institute Innovation Council. Total credits required is 3.

b) Digital Automation Technologies (DA):

Courses related to programming / automation tools & techniques / Industry 4.0. These courses may be proposed by Departments/Schools/Centres and approved by the Senate to be included in the DA category. Total credits required is 6. All of CSE's PEs can be credited as DA electives too.

c) Humanities, Social Science, Management (HM):

Courses such as Indian and Foreign languages, Economics, Engineering Management, Financial Management and Design Thinking. These courses may be proposed by Departments/Schools/Centres and approved by the Senate to be included in HM category. Total credits required is 9.

5. ACTIVITY CREDITS (AC)

A minimum of 80 Activity Points are to be acquired for obtaining the 4 Activity Credits required in the curriculum.
Activity points acquired should be a minimum of 20 at the end of S4.
Activity points acquired should be a minimum of 40 at the end of S6.

PROGRAMME STRUCTURE

Semester I

Sl. No.	Course Code	Course Title	L	T	P	O	Credits	Category
1.	MA1002E	Mathematics I	3	1*	0	5	3	IC
2.	MS1001E	Professional Communication	3	0	0	6	3	IC
3.	CS1001E	Computer Programming	3	0	0	6	3	PC
4.	CS1002E	Introduction to Computing Science	3	0	0	6	3	PC
5.	CS1003E	Discrete Structures I	3	0	0	6	3	PC
6.	CS1091E	Programming Laboratory	0	0	3	3	2	PC
		Total					**17**	**--**

Semester II

Sl. No.	Course Code	Course Title	L	T	P	O	Credits	Category
1.	MA1012E	Mathematics II	3	1*	0	5	3	IC
2.	PH1001E	Physics of Materials	3	0	0	6	3	IC
3.	BT1001E	Biology for Engineers	3	0	0	6	3	IC
4.	CS1011E	Program Design	3	0	0	6	3	PC
5.	CS1012E	Logic Design	3	0	0	6	3	PC
6.	CS1013E	Discrete Structures II	3	0	0	6	3	PC
7.	CS1092E	Program Design Laboratory	1	0	3	5	3	PC
		Total					**21**	**--**

Semester III

Sl. No.	Course Code	Course Title	L	T	P	O	Credits	Category
1.	MA2002E	Mathematics III	3	1*	0	5	3	IC
2.	CS2001E	Data Structures and Algorithms	3	1	2	6	4	PC
3.	CS2002E	Computer Organization	3	1	2	6	4	PC
4.		EI Elective	3	0	0	6	3	EI
5.	CS2091E	Data Structures and Algorithms Laboratory	1	0	3	5	3	PC
6.	CS2092E	Hardware Laboratory	1	0	3	5	3	PC
		Total					**20**	--

Semester IV

Sl. No.	Course Code	Course Title	L	T	P	O	Credits	Category
1.	MA2012E	Mathematics IV	3	1*	0	5	3	IC
2.	CS2011E	Database Management Systems	3	1	2	6	4	PC
3.	CS2012E	Operating Systems	3	1	2	6	4	PC
4.	CS2013E	Theory of Computation	3	0	0	6	3	PC
5.	CS2019E	Professional Ethics	1	0	0	2	1	IC
6.		DA Elective - 1	3	0	0	6	3	DA
7.		Minor Course - 1	3	0	0	6	$3^{\#}$	MC
		Total (Excluding the Minor Courses)					**18**	--

Semester V

Sl. No	Course Code	Course Title	L	T	P	O	Credits	Category
1.	CS3001E	Computer Networks	3	1	2	6	4	PC
2.	CS3002E	Compiler Design	3	1	2	6	4	PC
3.	CS3003E	Design and Analysis of Algorithms	3	1	2	6	4	PC
4.		Humanities Elective - 1	3	0	0	6	3	HM
5.		DA Elective - 2	3	0	0	6	3	DA
6.		Minor Course - 2	3	0	0	6	$3^{\#}$	MC
		Total (Excluding the Minor Courses)					**18**	--

Semester VI

Sl. No.	Course Code	Course Title	L	T	P	O	Credits	Category
1.	CS3011E	Software Engineering	3	1	2	6	4	PC
2.	CS3012E	Artificial Intelligence	3	1	2	6	4	PC
3.		Humanities Elective - 2	3	0	0	6	3	HM
4.		Open Elective - 1	3	0	0	6	3	OE
5.		Open Elective - 2	3	0	0	6	3	OE
6.	CS3099E	Project	0	0	0	9	3	PC
7.		Minor Course - 3	3	0	0	6	3#	MC
		Total (Excluding the Minor Courses)					**20**	**--**

Semester VII

Sl. No.	Course Code	Course Title	L	T	P	O	Credits	Category
1.		Humanities Elective - 3	3	0	0	6	3	HM
2.		Open Elective - 3	3	0	0	6	3	OE
3.		Open Elective - 4	3	0	0	6	3	OE
4.		Open Elective - 5	3	0	0	6	3	OE
5.		Open Elective - 6	3	0	0	6	3	OE
6.	CS4097E	Summer Internship	-	-	-	*	2	PC
7.	CS4098E /	Project / Programme Elective - 1	-	-	-	-	3	PE
8.		Minor Course - 4	3	0	0	6	3#	MC
		Total (Excluding the Minor Courses)					**20**	**--**

***** CS4097E Summer Internship (including the academic internship) is to be completed during the vacation after S6, and the evaluation will be done in S7. Working hours will be decided by the organisation in which the internship is done.

Semester VIII

Sl. No.	Course Code	Course Title	L	T	P	O	Credits	Category
1.	CS4099E /	Project / Programme Elective - 2, Programme Elective - 3	-	-	-	-	6	PE
2.		Open Elective - 7	3	0	0	6	3	OE
3.		Open Elective - 8	3	0	0	6	3	OE
4.	CS4096E	Activity Credits (minimum of 80 points)	-	-	-	-	4	AC
		Total					**16**	**--**

DETAILED SYLLABUS

(2023 Admission onwards)

MA1002E MATHEMATICS I

L	T	P	O	C
3	1	0	5	3

Total: 39 Lecture sessions

Course Outcomes:

CO1: Find the limits, check for continuity and differentiability of real valued functions of one variable.
CO2: Find the limits, check for continuity and differentiability of real valued functions of two variables.
CO3: Find the maxima and minima of real valued functions of one variable or two variables.
CO4: Find the parametric representation of curves and surfaces in space and evaluate integrals over curves and surfaces.

Functions of one variable: limit, continuity, differentiability, local maxima and local minima, mean value theorems, Taylor's theorem, L'hôpital's rule, integration, fundamental theorem of calculus, volume, area, improper integrals, Gamma and Beta functions.
Parameterised curves in space, arc length, tangent and normal vectors, curvature and torsion.

Functions of several variables: limit, continuity, partial derivatives, partial differentiation of composite functions, directional derivatives, gradient, local maxima and local minima of functions of two variables, critical point, saddle point, Taylor's formula for two variables, Hessian, second derivative test, method of Lagrange multipliers.
Evaluation of double integrals, improper integrals, change of variables, Jacobian, polar coordinates, triple integral, cylindrical and spherical coordinates, mass of a lamina, centre of gravity, moments of inertia.

Vector field: divergence, curl, identities involving divergence and curl, scalar potential.
Line integral, independence of path, irrotational and solenoidal vector fields, Green's theorem for plane, parameterized surface, surface area and surface integral, flux, Gauss' divergence theorem, Stokes' theorem.

References:

1. H. Anton, I. Bivens and S. Davis, *Calculus, 10th ed*, New York: John Wiley & Sons, 2015.
2. G. B. Thomas, M. D. Weirand and J. Hass, *Thomas' Calculus*, 12th ed, New Delhi, India: Pearson Education, 2015.
3. E. Kreyszig, *Advanced Engineering Mathematics,* 10th ed, New York: John Wiley & Sons, 2015.
4. T. M. Apostol, *Calculus Vol 1*, 1st ed. New Delhi: Wiley, 2014.

PH1001E PHYSICS OF MATERIALS

L	T	P	O	C
3	0	0	6	3

Total Lecture Sessions: 39

Course Outcomes:

CO1: Explain the fundamentals of quantum mechanics.

CO2: Apply quantum mechanics to electron in crystals and study the formation of bands in solid.

CO3: Apply quantum mechanics and study the electrical properties of solids.

CO4: Explain conductivity in semiconducting materials and influence of dopants on conductivity.

Quantum Mechanics
Wave-particle duality – de Broglie waves – group and phase velocity – Davison-Germer experi- ment – uncertainty principle – properties and significance of wave function – Schrodinger's wave equation – steady state equation, applications to a free particle and particle in a box.

Band theory of solids
Electrons in periodic potential – origin of band in solid – Bloch theorem – Kronig-Penny model (qualitative) – E-k diagram for free electron and electrons in periodic potential – one dimensional zone scheme – band gap.

Electrical conductivity
Classical electron theory – conductivity – factors affecting resistivity – Quantum mechanical consideration, Fermi energy and Fermi Surface – Fermi distribution function, density of states – Effective mass of electron.

Semiconductors
Intrinsic and extrinsic semiconductors – carrier concentration in *n* and *p* types semiconductors – Fermi level – Temperature dependence of electrical conductivity – variation of Fermi level with temperature.

References:
1. A. Beiser, *Concepts of Modern Physics (6th Edition)*, McGraw-Hill, 2009.
2. K. Krane, *Modern Physics (4th Indian Edition)*, Wiley, 2021.
3. R. E. Hummel, *Electronic Properties of Materials (4th Edition)*, Springer, 2014.
4. M. A. Wahab, *Solid State Physics – Structure and Properties of Materials (3rd Edition)*, Narosa, 2015.

BT1001E BIOLOGY FOR ENGINEERS

L	T	P	O	C
3	0	0	6	3

Total Lecture Sessions: 39

Course Outcomes:

CO1: Explain the evolutionary basis of life and the structure-function relationship of biomolecules.

CO2: Describe the characteristics of the cellular composition, communication, and principles of genetics.

CO3: Evaluate and apply the biological principles to solve real-world problems related to food, agriculture, and environmental sciences

CO4: Apply advanced engineering techniques and tools to create medical devices, diagnostics, and therapies.

Exploring Biology
Origin of life, Darwinian Evolution, historical perspectives – Structure and function of Biomolecules: carbohydrates (mono-, di-, and poly- saccharides), lipids, proteins (amino acids, peptides), and nucleic acids (DNA & RNA) - central dogma of molecular biology and gene regulation.

Cell Biology & Genetics
Cell structure and function: Prokaryotic & Eukaryotic cells – Bioenergetics - Cell communication: Signal Transduction - Cell division: Cell Cycle, Binary fission, Mitosis & Meiosis - Genetics & inheritance: Mendelian Genetics, Chromosomal Aberrations.

Human Organs & Bio-design
Brain: Nervous system, Brain-computer interfaces, EEG, Engineering Solutions for nervous disorder – Eye: Vision physiology, Bionic eye – Heart: Cardiac Function, ECG, stents and pacemakers – Lungs: Physiology, Ventilators & Spirometry, Heart-Lung machine – Stem cells and Regenerative Medicine.

Global Challenges in Medicine, Agriculture and Environment
Human Diseases, Disorders & Drugs, Biosensors, Biomedical Diagnostics, Vaccines & Antibiotics - Genetic modification of crops, precision farming and sustainable agriculture - Biodegradation and bioremediation of pollutants, biofuels - Recent advances in biology.

References:

1. J. B. Reece, L. A. Urry, M. L. Cain, S. A. Wasserman, P. V. Minorsky and R. B. Jackson, *Campbell biology (Vol. 9)*, Boston. Pearson, 2014.
2. S. Thyagarajan, N. Selvamurugan, M. P. Rajesh, R. A. Nazeer, R. W. Thilagaraj, S. Bharathi and M. K. Jaganathan, *Biology for Engineers,* McGraw Hill Education, 2013.
3. R. M. Cummings, W. S. Klug, C. A. Spencer, M. A. Palladino and D. Killian. *Concepts of Genetics,* 12th ed., Pearson, 2019.
4. D. L. Nelson, and N. M. Cox, *Lehninger principles of biochemistry,* 8th ed.. W.H. Freeman, 2021.
5. S. I. Fox, *Human physiology,* 13th. ed. New York, NY: McGraw--Hill, 2011.
6. A. G. Webb, *Principles of biomedical instrumentation.* Cambridge University Press, 2018.
7. A. T. Johnson, *Biology for engineers,* CRC Press, 2011.

MS1001E PROFESSIONAL COMMUNICATION

L	T	P	O	C
3	1	0	5	3

Total Lecture Sessions: 39

Course Outcomes:
CO1: Distinguish the role and purpose of communication at the workplace and for academic purposes.
CO2: Decide strategies and modes for effective communication in a dynamic workplace.
CO3: Combine multiple approaches for successful and ethical information exchange.
CO4: Estimate best communication practices to assist productivity and congeniality at the workplace.

Listening and Reading Comprehension
Conversation starters: introductions and small talk - Seek and provide information, clarification, polite enquiries, requests, congratulate people, apologise, give and respond to feedback - Describe graphs, tables, and charts - Words often confused: Lexicon and Meaning - Sense Groups - Listening for specific purposes: Listening to lectures, Summarise academic lectures for note-taking - Appropriate Language to Request and Respond - Public Speaking

Vocabulary and Speaking
Developing professional vocabulary - Basic Sentence Structures from Reading Texts - Concord - Functions of Auxiliary Verbs and Modals - Strategies for Effective Reading - Skimming and Scanning, Determine themes and main ideas, Predicting content using photos, images and titles - Critical Reading: Discussing and Summarising text points - Understanding Text Structures: sequencing, comparing and contrasting, relating cause and effect, problems and problem-solving - Discussing Rhetorical and Cultural Aspects in Texts - Text Appreciation: Drawing inferences, Framing Opinions and Judgments on Reading Text

Effective Writing
Note Making and Summarising: Prepare notes from reading texts, Paraphrasing - Use of Multimedia for Assistive Purposes - Paragraph Writing: cohesive devices to connect sentences in a paragraph - transitional devices - Use Text Structures in Paragraphs: sequencing, comparing and contrasting, relating cause and effect, problems and problem-solving - Avoiding Ambiguity and Cleft Sentences - Applications- Writing Instructions, Descriptions and Explanations - Official Letters of Request and Denial - Official E-mails - Abstract Writing - Digital Resources for Effective Communication

Communication at Workplace
Communication Theory - Process of Communication - Modes of Communication - Verbal and Non-Verbal Communication - Tone in Communication - Formal and Informal Communication at Workplace - Passive, Assertive and Aggressive Styles of Communication - Positive Body Language - Group Discussions - Presentation - Workplace Communication - Active Listening - Giving Feedback - Communication Etiquette - Persuasion - Negotiation - Tone and Voice - Telephone etiquette - Establishing Credibility in Conversations - Digital Communication and Netiquette: Conducting Oneself in Virtual Interactions, Constructive use of Social media - Ethical and Culturally Sensitive Communication: Ethical considerations in professional communication, Addressing diversity, Inclusive Communication Practices

References:

1. Bhatnagar, N. and Bhatnagar, M., *Communicative English for engineers and professionals*, Dorling Kindersley, 2010.
2. Foley, M. and Hall, D., *Longman advanced learners' grammar: A self-study reference & practice book with answers*, Pearson Education, 2018.
3. Garner, B. A., *HBR Guide to better business writing: Engage readers, tighten and Brighten, make your case*, Harvard Business Review Press, 2012.
4. Hewings, M., *Advanced grammar in use: A reference and practice book for Advanced learners of English*. Cambridge University Press, 2013.
5. Ibbotson, M., Cambridge *English for Engineering*. Cambridge University Press, 2015.
6. Kumar, S. and Lata, P., *Communication Skills*. Oxford University Press, 2015.
7. Sudarshana, N. and Savitha, C., *English for Technical Communication*. Cambridge English, 2016.

CS1001E COMPUTER PROGRAMMING

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Analyse a computational problem and design approaches for solving it

CO2: Design algorithms for simple computational problems

CO3: Illustrate algorithmic solutions in the C programming language

Introduction to Computing

Fundamentals of Computing: historical perspective, early computers. Formal problem specification, pseudocode and flowcharts. Memory, Variables, Values, Instructions, Programs.

Data Types, Operators, Expressions and Statements

Variables and constants - declarations - arithmetic and logical operators Assignment operator Input/output. Control Flow: Statements and blocks if-else, switch, while, for and do-while statements break and continue goto and labels.

Functions and Program structure

Basics of functions, Parameter passing, scope rules, recursion.

Aggregate data types and File Management

Single and multidimensional arrays, structures and unions, Pointers to arrays and structures -passing arrays and pointers as arguments to functions
File management - opening and closing files, reading and writing to files, operations on files.

References:

1. B. S. Gottfried, *Programming with C (Schaum's Outline Series)*, 2nd ed. McGraw-Hill, 1996.
2. S. C. Kochan, *Programming in C,* Sams Publishing, 3rd ed. 2004.
3. B. W. Kernighan and D. M. Ritchie, *The C Programming Language,* 2nd ed. UK: Prentice Hall, 1988.
4. W. Kernighan and B. Pike, *The Practice of Programming,* UK: Addison-Wesley, 1999
5. H. M. Deitel and P. J. Deitel, *C: How to program,* 8th ed. Pearson Education, 2015.
6. P. Prinz and T. Crawford, *C in a Nutshell: The Definitive Reference*, 2nd ed., O'Reilly Media, 2016.

CS1002E INTRODUCTION TO COMPUTING SCIENCE

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Illustrate the elements of computing system and representation of data in computing system

CO2: Illustrate the functions and features of system software

CO3: Apply computational thinking skills to analyse and solve problems efficiently and effectively

Elements of Computing Systems

Brief introduction to history of computing - Von Neumann Architecture - Data Representation: binary numbering system, representation of numeric, text and image data - Basic logic gates : half adder- Logic gates as building blocks of a computer - Components of a modern computer system : Basics of ALU, Memory and Control Unit Instruction set - Fetch-execute process of assembly code, execution of simple programs like adding two numbers.

Overview of Systems Software

Concept of system software - assembly language translation and loading - compilers and language translation, introduction to the general structure of a compiler in brief - operating systems and application interface - user interface - efficient allocation of resources - system security - safe use of resources

Computational Thinking

What is Computational Thinking Sample Datasets - Iterator Flowcharts - Variables: Count Sum Average Accumulator - Filtering: Simple Conditions, Compound Conditions, Looking for a data element, Dynamic Conditions (Maximum Minimum) - Data types: Basics - Compound data types: Subtypes - Pseudocode

References:

1. G. Michael Schneider and Judith Gersting, *Invitation to computer science*, 1st ed. India: Cengage Learning, 2022.
2. G. Venkatesh and Madhavan Mukund, *Computational Thinking: A Primer for Programmers and Data Scientists*, 1st ed. India: Notion Press, 2021.

CS1003E DISCRETE STRUCTURES - I

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Classify a relation as equivalence relation/partial order/lattice and perform closure operations.

CO2: Formulate and solve recurrence equations describing the complexity of recursive algorithms

CO3: Formulate elementary problems with graphs and identify elementary algorithmic methods to solve them.

Sets and Relations

Sets and Relations: countable and uncountable sets, diagonalization, equivalence relations and partitions, posets and lattices, digraph representation for relations, adjacency matrix/list representations, transitive closure computation - Floyd Warshall algorithm.

Recurrences

Induction and Counting: mathematical Induction - review and examples, pigeonhole principle and inclusion exclusion principle. Recurrences: inductive formulations and iterative solution methods, applications to analysis of recursive algorithms, solution by the method of generating functions.

Graph Theory

Graph Theory: elementary properties of graphs and trees, BFS and DFS algorithms, bipartite graphs and properties, two colouring of bipartite graphs, Eulerian and Hamiltonian graphs, matching and Hall's matching condition, planar graphs and five colour theorem, five colouring algorithm.

References:

1. R. P. Grimaldi and B. V. Ramana, *Discrete and Combinatorial Mathematics: An Applied Introduction,* 5th ed. India: Pearson Education, 2006.
2. L. Lovasz, J. Pelikan, and K. Vesztergombi, *Discrete Mathematics: Elementary and Beyond*, 1st ed. Springer, 2003
3. B. Kolman, R. Busby and S. C. Ross, *Discrete Mathematical Structures,* 6th ed. India: Pearson Education, 2015.

CS1091E PROGRAMMING LABORATORY

L	T	P	O	C
0	0	3	3	2

Total Sessions: Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate an ability to work in a UNIX environment

CO2: Demonstrate an ability to develop algorithmic solutions for simple computational problems

CO3: Implement algorithmic solutions using the C programming language

Syllabus:

Linux, Editor, Compiler and Debugger - Introduction to C programming - Statements, Assignment statements, Control Statements, Loop Statements - Arrays-Strings - Pointers - Functions - Recursion - Structures and Union - File Input Output

References:

1. B. S. Gottfried, *Programming with C (Schaum's Outline Series)*, 2nd ed. McGraw-Hill, 1996.
2. S. C. Kochan, *Programming in C,* Sams Publishing, 3rd ed. 2004.
3. B. W. Kernighan and D. M. Ritchie, *The C Programming Language,* 2nd ed. UK: Prentice Hall, 1988.
4. W. Kernighan and B. Pike, *The Practice of Programming,* UK: Addison-Wesley, 1999.
5. H. M. Deitel and P. J. Deitel, *C: How to program,* 8th ed. Pearson Education, 2015.
6. P. Prinz and T. Crawford, *C in a Nutshell: The Definitive Reference*, 2nd ed., O'Reilly Media, 2016.

MA1012E MATHEMATICS II

L	T	P	O	C
3	1	0	5	3

Total: 39 Lecture sessions

Course Outcomes
CO1: Acquire sufficient knowledge about convergence of sequences and series and various methods of testing for convergence.
CO2: Solve linear ODEs with constant coefficients.
CO3: Test the consistency of the system of linear equations and solve it.
CO4: Acquire sufficient knowledge about vector spaces, linear transformation and theory of matrices.
CO5: Diagonalise symmetric matrices and use it to find the nature of quadratic forms.

Numerical sequences, Cauchy sequence, convergence of sequences, series, convergence of series, tests for convergence, absolute convergence. Sequence of functions, power series, radius of convergence, Taylor series. Periodic functions and Fourier series expansions, Half-range expansions.

Existence and uniqueness of solution of first order ordinary differential equations (ODEs), methods of solutions of first order ODE, Linear ODE, linear homogeneous second order ODEs with constant coefficients, fundamental system of solutions, Wronskian, linear independence of solutions, method of undetermined coefficients, solution by variation of parameters.

System of linear equations: Gauss elimination method, row echelon form, row space, row rank, existence and uniqueness, homogeneous system, solution space, rank-nullity relation for homogeneous linear system.
Abstract vector space, subspace, linear independence and span, basis, dimension, linear transformation, kernel, range, rank-nullity theorem.

Coordinates, matrix representation of linear transformation, base changing rule, eigenspace, diagonalisation of linear operator. Eigenvalues and eigenvectors of a matrix, Cayley-Hamilton theorem, diagonalisation of symmetric matrices, quadratic forms, transformation into principal axes, eigenvalue method of solving system of first order linear ODEs with constant coefficients.

References:

1. H. Anton, I. Bivens and S. Davis, *Calculus,* 10th ed. John Wiley & Sons, 2015.
2. T. A. Apostol, *Calculus Vol 1*, 1st ed. New Delhi: Wiley, 2014.
3. E. Kreyszig, *Advanced Engineering Mathematics,* 10th ed. Wiley, 2015.
4. G. Strang, *Differential Equations and Linear Algebra*, Cambridge Press, 2014.
5. S. W. Goode and S. Annin, *Differential Equations and Linear Algebra,* Pearson Prentice Hall, 2007.
6. O. Bretscher. *Linear algebra with applications,* New Delhi: Prentice Hall, 1997.

CS1011E PROGRAM DESIGN

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Design, analyse and prove the correctness of simple, iterative and recursive algorithms.

CO2: Analyse algorithms for sorting and searching.

CO3: Select appropriate data structure for solving a given problem.

Searching and Asymptotic Analysis

Review of Programming Constructs - Conditional, Iterative and Control constructs, Functions, Recursion, Searching - Linear and Binary, correctness and step count analysis, Asymptotic notation for complexity analysis.

Sorting Algorithms

Sorting - Insertion and Selection sorts, Divide and conquer, Merge Sort, Quick sort, Linear and External Sorting. Correctness, Analysis and Applications of sorting.

Basic Data Structures

Pointers and dynamic memory allocation, Strings manipulation using Multidimensional arrays and pointer arrays, Abstract Data Types, Lists - Singly and doubly linked list, Stacks, Queues using array and pointer based implementations.

Trees and Hashing

Introduction to Graphs, Trees, Binary trees, Heaps, Heap Sort and Priority queues. Binary search trees, and traversal algorithm, Hashing - Chaining and open addressing methods

References:

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data Structures and Algorithms*, 1st ed. India: Dorling Kindersly, 2009.
2. T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, *Introduction to Algorithms*, 3rd ed. MIT Press, 2009.
3. E. Horowitz, S. Sahni and D. Mehta, *Fundamentals of Data Structures in C++*, 2nd ed. Universities Press, 2008.
4. S. Dasgupta, C. H. Papadimitriou and U. Vazirani. *Algorithms* 1st ed. McGraw-Hill, 2006.

CS1012E LOGIC DESIGN

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Assess various number systems and apply them in digital design.

CO2: Design logic functions utilising logic gates and programmable logic.

CO3: Design simple digital systems.

Number theory and boolean algebra

Number systems and codes, Boolean algebra: postulates and theorems, constants, variables and functions, switching algebra, Boolean functions and logical operations, Karnaugh map: prime cubes, minimum sum of products and product of sums.

Design and analysis of combinational logic

Quine-McCluskey algorithm, prime implicant chart, cyclic prime implicant chart, Petrick’s method, Combinational Logic: introduction, analysis and design of combinational logic circuits, parallel adders and look-ahead adders.

Design of digital logic devices

Comparators, Decoders and encoders, code conversion, multiplexers and demultiplexers, parity generators and checkers, Programmable Logic Devices: ROMs, PALs, PLAs.

Design and analysis of sequential logic

Introduction to sequential circuits, memory elements, latches. Flip-flops, analysis of sequential circuits, state tables, state diagrams, design of sequential circuits, excitation tables, Mealy and Moore models, registers, shift registers, counters

References:

1. T. L. Floyd and R. P. Jain, *Digital Fundamentals*, 8th ed. Pearson Education, 2006.
2. C. H., Roth Jr. and L. L. Kinney, *Fundamentals of Logic Design*, 6th ed. Cengage Learning, 2009.
3. M. M. Mano and M. D. Ciletti, *Digital Design*, 4th ed. Pearson Education, 2008.
4. B. J. LaMeres, *Introduction to Logic Circuits & Logic Design with Verilog*, 1st ed. Springer, 2017.

CS1013E DISCRETE STRUCTURES - II

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Formulate and solve problems in propositional/predicate logic and perform formal deductions
CO2: Apply elementary algebraic and number theoretic concepts to solve modular linear equations and related problems.
CO3: Draw elementary probabilistic inferences and compute mathematical expectation in simple algorithmic and combinatorial problems.

Logic, Sets, and Relations

Propositional Logic: formulas and truth assignments, logical consequences and deductions, formula equivalences, inference by contradiction and contraposition, resolution algorithm

Predicate Logic: quantifiers, deduction rules, models and satisfiability.

Algebra and Number Theory

Algebra: Elementary properties of groups, rings, integral domains and fields - Lagrange's theorem.

Number Theory and Applications: Euclid's algorithm, complexity analysis - solution to congruences, modular exponentiation and inversion, Euler's theorem and Fermat's theorem, Fermat's test for primality.
Chinese remaindering, overview of RSA cryptosystem.

Probabilistic Method

Elementary combinatorics - ball- bin problems, discrete probability, conditional probability and probabilistic reasoning - expectation and conditional expectation, probabilistic method, applications to algorithm design - expected and average case analysis.

References:

1. R. P. Grimaldi and B. V. Ramana, *Discrete and Combinatorial Mathematics: An Applied Introduction,* 5th ed. India: Pearson Education, 2006.
2. I. M. Copi, *Symbolic Logic*, 5th ed. India: Pearson Education, 2015.
3. K. Ireland and M. Rosen, *A classical Introduction to Modern Number Theory*, 2nd ed. India: Springer, 1990.
4. L. Lovasz, J. Pelikan and K. Vesztergombi, *Discrete Mathematics: Elementary and Beyond*, 1st ed. Springer, 2003.

CS1092E PROGRAM DESIGN LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13, Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Implement fundamental algorithms like sorting and searching.

CO2: Implement basic data structures like list, stack, queue and tree.

CO3: Develop efficient algorithmic solution for a given problem and implement the solution using appropriate data structures.

Theory

Review of dynamic memory allocation - use of pointers - review of recursion. File organization.

Practical

1. Iterative and recursive algorithms
2. Linear and Binary search implementations
3. Sorting: Insertion sort, Selection sort and Linear sort implementations
4. Quick sort and Merge sort implementations
5. Stack and Queue implementation using arrays and linked list
6. Binary tree representation, Arithmetic expression to postfix
7. Postfix to expression tree, tree traversal and evaluation
8. Heap sort and priority queue implementation
9. Binary search tree - insert, delete and search.

References:

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, *Introduction to Algorithms*, 3rd ed. MIT Press, 2009.
2. E. Horowitz, S. Sahni and D. Mehta, *Fundamentals of Data Structures in C++*, 2nd ed. Universities Press, 2008.
3. M. A. Weiss, *Data structures and algorithm analysis in C++*, 4th ed. Addison-Wesley, Boston, 2014.

MA2002E MATHEMATICS III

L	T	P	O	C
3	1	0	5	3

Total: 39 Lecture sessions

Course Outcomes
CO1: Analyse different types of linear operators on inner-product-spaces and their spectral decomposition.
CO2: Find the singular value decomposition of linear transformations and matrices and use it to obtain low rank approximations.
CO3: Use the Cauchy-Riemann equations to check the complex differentiability of functions.
CO4: Classify singularities and poles, find residues and evaluate complex integrals using the residue theorem.

Linear Algebra

Review of vector spaces, linear transformations, minimal polynomial, eigenvalues, diagonalizability, inner product spaces, norm, orthogonality of the subspaces associated to a linear transformation, orthonormal basis, Schur's theorem, linear functional, Riesz representation theorem, orthogonal complement, direct sum, orthogonal projection, best approximation, least square approximation, pseudoinverse, Adjoint of a linear operator, null-space and range of adjoint, self-adjoint and normal operators, minimal polynomial of self-adjoint operator, spectral theorem for self-adjoint operators, spectral theorem for normal operators, singular value decomposition, low-rank approximation, Eckart-Young Theorem.

Complex Analysis

Complex functions, derivative, holomorphic function, Cauchy-Riemann equations, harmonic functions, harmonic conjugate, Line integral in the complex plane, Cauchy's integral Theorem, Cauchy's integral formula, Derivatives of analytic functions, power series, zeros of holomorphic functions, isolated singularities, Laurent's series, residue integration method, evaluation of real improper integral.

References:

1. S. Axler, *Linear Algebra Done Righ*t, 4th Edition, Springer, 2023
2. G. Strang, *Introduction to Linear Algebra*, 5th Edition, Wellesley Cambridge Press, 2016.
3. J W Brown, R V. Churchill, *Complex Variables and Applications*, 9th Edition, McGraw-Hill Education, 2009.
4. S. Ponnusamy, H Silverman, *Complex Variables with Applications*, Birkhauser, 2006.
5. E. Kreyszig, H. Kreyszig, E. J. Norminton, *Advanced Engineering Mathematics*, 10th Edition, Wiley, New Delhi, 2015.
6. C. R. Wylie, L. C. Barrett, *Advanced Engineering Mathematics*, 6th Edition, McGraw-Hill, New Delhi, 1995.

CS2001E DATA STRUCTURES AND ALGORITHMS

L	T	P	O	C
3	1	2	6	4

Total Sessions: Lecture 39, Tutorial 13, Practical 26

Course Outcomes:

CO1: Illustrate the different methods of Hashing and their applications

CO2: Model the real world problems as Graphs and solve the given computational problem using appropriate graph algorithms

CO3: Design and analyse efficient algorithms for computational problems using appropriate data structures

Review: Simple data structures and applications, Time and space complexity analysis, Lower bound theory for sorting, Dictionaries, Hashing. Trees and Positional trees. Review of Binary Trees and Binary Search Trees. Rotations. Red black Trees, AVL Trees, Splay trees

Data Structure to handle priority queue - Heaps, Min-Max Heaps, Leftist Trees, Deaps, Meldable heaps - Binomial and Fibonacci heaps - algorithms for different operations on these heaps and their analysis.

Graph representation - DFS, BFS, and their applications. Minimum spanning tree problem - Kruskal's algorithm - analysis and implementation using disjoint set data structure – Prim’s algorithm - Shortest path problem - Dijkstra's algorithm - analysis and implementation of Prim's and Dijkstra's algorithms using priority queues. Bellman Ford, Floyd-Warshall algorithms

Data structures to handle Strings and String processing: String Operations, Brute-Force Pattern Matching, The Boyer-Moore Algorithm, The Knuth-Morris-Pratt Algorithm, Standard Tries, Compressed Tries, Suffix Tries.

References:

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3/e, MIT Press, 2003
2. A. V. Aho, J. D. Ullman, and J. E. Hopcroft, *Data Structures and Algorithms*, Addison Wesley, 1996.
3. Dasgupta, Sanjoy, Christos Papadimitriou, and Umesh Vazirani, *Algorithms*. McGraw-Hill, 2008.
4. R. Sedgewick, and K. Wayne. *Algorithms*, Addison-Wesley, 4th Edition. 2011.

CS2002E COMPUTER ORGANIZATION

L	T	P	O	C
3	1	2	6	4

Total Sessions: Lecture 39, Tutorial 13 and Practical 26

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Analyse hardware components of a computer system and its overall organization.

CO2: Analyse the architectural design of the MIPS processor and its instruction set.

CO3: Analyse a RISC processor's performance using standard performance evaluation metrics.

Computer abstraction and technology: basic principles, hardware components, Instructions: operations and operands of the computer hardware, representing instructions, making decisions, supporting procedures, character manipulation, styles of addressing, starting a program.

Computer arithmetic: signed and unsigned numbers, addition and subtraction, logical operations, constructing an ALU, multiplication and division, floating point representation and arithmetic, parallelism and computer arithmetic

Measuring performance: evaluating, comparing, and summarizing performance The processor: building a data path, simple and multicycle implementations, microprogramming, exceptions, Pipelining, data path and control, different types of hazards in pipelined processors

Memory hierarchy: caches, cache replacement policies, cache performance, virtual memory, common framework for memory hierarchies

References:

1. D. A. Patterson and J. L. Hennessy, *Computer Organisation and Design: The Hardware/Software Interface*, 6/e, Morgan Kaufmann, 2018.
2. V. P. Heuring and H. F. Jordan, *Computer System Design and Architecture*, Prentice Hall, 2003.

CS2091E DATA STRUCTURES AND ALGORITHMS LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Design and Implement different Hash algorithms

CO2: Identify suitable data structures to implement graph algorithms efficiently

CO3: Design and implement efficient algorithms for computational problems using appropriate data structures

Syllabus:

Theory

Review of dynamic memory allocation - use of pointers - review of recursion. File organisation.

Practical

1. Hash tables
2. AVL trees, Red Black trees and Splay trees
3. Min-max Heaps, Deaps
4. Linear time DFS and BFS implementation with adjacency list representation.
5. Kruskal's algorithm implementation in O((n+e)log n) complexity.
6. Prim's algorithm implementation in O((n+e)log n) complexity.
7. Dijkstra's algorithm implementation in O((n+e)log n) complexity.
8. Bellman Ford and Floyd Warshall algorithms
9. String algorithms.

References:

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3/e, MIT Press, 2003
2. S. Sahni, *Data Structures, Algorithms, and Applications in C++*, McGraw Hill, 1998
3. A. V. Aho, J. D. Ullman, and J. E. Hopcroft, *Data Structures and Algorithms*, Addison Wesley, 1996.
4. Dasgupta, Sanjoy, Christos Papadimitriou, and Umesh Vazirani, *Algorithms*. McGraw-Hill, 2008.
5. R. Sedgewick, and K. Wayne. *Algorithms*, Addison-Wesley, 4th Edition, 2011

CS2092E HARDWARE LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Design and implement combinational and sequential logic circuits.

CO2: Design and implement (in a HDL) memory units.

CO3: Design and implement (in a HDL) a simple single cycle processor.

Theory

Hardware Description Languages and use cases.

Practical

Design and implementation of the following using HDLs

1. Combinational and sequential logic.
2. Arithmetic Logic Unit
3. Register file
4. Control Unit
5. Single cycle processor implementation

References:

1. S. Palnitkar, *Verilog HDL: A Guide to Digital Design and Synthesis* 2nd ed , Prentice Hall PTR 2003.
2. B. J. LaMeres, *Introduction to Logic Circuits & Logic Design with Verilog*, 1/e, Springer, 2017.
3. S. Brown and Z. Vranesic, *Fundamentals of Digital Logic with Verilog Design*, McGraw-Hill Higher Education, Har/Cdr edition, 2002.
4. M. M. Mano and M. D. Ciletti, *Digital Design*, 4/e, Pearson Education, 2008
5. T. L. Floyd and R. P. Jain, *Digital Fundamentals,* 8/e, Pearson Education, 2006.

MA2012E MATHEMATICS IV

L	T	P	O	C
3	1	0	5	3

Total: 39 Lecture sessions

Course Outcomes:

CO1: Solve problems involving random variables and functions of random variables
CO2: Perform de-correlation of random vectors using PCA.
CO3: Apply regression and correlation analysis for studying relationship between variables
CO4: Use probabilistic and statistical analysis in various applications of engineering.

Probability
Sample space, probability, conditional probability, independence, random variable, probability distributions, binomial distribution, Poisson distribution, geometric distribution, continuous random variable, probability density function, normal distribution, uniform distribution, Markov's inequality, Chebyshev's inequality, moments and moment generating function, normal approximation of binomial distribution, functions of random variables.
Jointly distributed random variables, marginal and conditional distributions, covariance, correlation coefficient and independence, joint probability distribution of functions of random variables, bi-variate normal distribution, random vectors, expectation vectors and covariance matrices, autocorrelation matrix, properties of covariance matrices, decorrelation of random vectors, multidimensional gaussian, transforming to independence, principal components analysis.

Statistical Inference
Population, sample and statistic, central limit theorem, sampling distribution of the mean, point estimation, unbiased estimator, sample variance, Chi square distribution, distribution of sample mean and sample variance, interval estimation of population mean and variance, Student's t-distribution, interval estimation of mean when variance is unknown, maximum likelihood estimator, testing of hypothesis, testing hypothesis concerning mean, hypotheses tests concerning variance, F distribution, Chi square test for goodness of fit, analysis of contingency tables, simple linear regression, method of least squares, sampling distribution of regression coefficients, inference concerning regression parameters sample correlation coefficient of paired data, regression to the mean,

References:

1. F. Dekking, C. Kraaikamp, H. Lopuhaä,, L. Meester, *A Modern Introduction to Probability and Statistics: Understanding Why and How*, Springer London, 2005.
2. H. Stark, J. W. Woods, *Probability, Statistics, and Random Processes for Engineers*. Pearson, 2012.
3. N. S. Matloff, *Probability and Statistics for Data Science: Math + R + Data*, CRC Press, 2020.
4. N. T. Kottegoda, R. Rosso, *Statistics, Probability, and Reliability for Civil and Environmental Engineers*. Colombia: McGraw-Hill, 1998.
5. S. M. Ross, *Introduction to Probability and Statistics for Engineers and Scientists*, Elsevier Science, 2009.

CS2011E DATABASE MANAGEMENT SYSTEM

L	T	P	O	C
3	1	2	6	4

Total Sessions: Lecture 39, Tutorial 13 and Practical 26

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate the physical and conceptual representation of relational databases.

CO2: Design good relation schemas based on functional dependencies and normal forms

CO3: Demonstrate various algorithms used for relational database schema design

CO4: Analyse transaction processing and database recovery techniques

Database System Concepts and architecture, Data Modeling using Entity Relationship (ER) model and Enhanced ER model, Specialization, Generalization, Data Storage and indexing, Single level and multi level indexing, Dynamic Multi level indexing using B Trees and B+ Trees.

The Relational Model, Relational Database design using ER to relational mapping, Relational algebra and relational calculus, Tuple Relational Calculus, Domain Relational Calculus, SQL, Query execution.

Database design theory and methodology, Functional Dependencies and Normalisation of relations, Normal Forms, Properties of relational decomposition, Algorithms for relational database schema design.

Transaction processing concepts, Schedules and serializability, Concurrency control, Two Phase Locking Techniques, Optimistic Concurrency Control, Database recovery concepts and techniques.

References:

1. R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems,* 6/e, Pearson Education, 2011.
2. R. Ramakrishnan and J. Gehrke, *Database Management Systems*, 3/e, McGraw Hill, 2003.
3. P. Rob and C. Coronel, *Database Systems- Design, Implementation and Management*, 7/e, Cengage Learning, 2007.

CS2012E OPERATING SYSTEMS

L	T	P	O	C
3	1	2	6	4

Total Sessions: Lecture 39, Tutorial 13 and Practical 26

Course Outcomes:

At the end of the course the student will be able:

CO1: To demonstrate the basic operating system functions and services.

CO2: To analyse the design features of traditional operating systems

CO3: To propose solutions to multitasking problems using operating system services.

Introduction to Operating System Concepts

Operating systems fundamentals - Overview -system structures -hardware - CPU, Memory and I/Odevices. Hardware interface . Interrupts and exceptions -interrupt service routine.Operating system interface -system calls -system call implementation. I/O devices-device controllers - device interface-polling, interrupt driven and DMA. I/O software layers- device drives, device independent I/O software , userspace I/O software. Booting the Operating System.

Process Management

Process management - - processes and threads - process address space - process hierarchy - scheduling algorithms - uniprocessor and multiprocessor scheduling -Case studies: O(1) scheduler, Completely Fair scheduler**.** Interprocess communication-Pipe, Named Pipe ,shared memory .synchronisation - Critical section- hardware and software support for synchronisation. Deadlock - prevention - avoidance - detection and recovery.

MemoryManagement

Memory management - memory allocation strategies - swapping- various management strategies - contiguous allocation, paging. Virtual memory - demand paging - page fault, page replacement algorithms, issues related to virtual memory implementation. Memory allocation for operating systems-Buddy system and slab allocator.

FileSystem Management

File management -Disk structure structure - secondary memory allocation. file abstraction and file naming - file system implementation: data structures and system calls -Virtual file systems. case studies: linux file system

References:

1. Charles Crowley, *Operating System: A design-oriented approach*, India ed. McGraw Hill Education, 2017.
2. A.Silberschatz, P. B.Galvin, and G.Gagne, *Operating System Principles*, 10th ed.John Wiley, 2013.
3. W. Stallings, *Operating Systems:Internals and design Principles*, 9th ed. Pearson Education, 2018.
4. A. S. Tanenbaum and Herbert Bos, *Modern Operating Systems*, 4th ed. Pearson Education, 2017.
5. G. J. Nutt, *Operating Systems - A Modern Perspective*, 3rd ed. Pearson Education, 2009.

CS2013E THEORY OF COMPUTATION

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Classify a given language according to its level in the Chomsky hierarchy and design grammars /machines for the language.

CO2: Construct finite state machines, pushdown automata and Turing machines for a given language.

CO3: Prove undecidability/decidability of a given problem using diagonal method or reduction.

CO4: Prove completeness and/or hardness of a given problem using reduction.

Basic concepts of Languages, Automata and Grammar. Regular Languages - Regular expression - finite automata equivalence, Myhill Nerode theorem and DFA State Minimization, Pumping Lemma and proof for the existence of non-regular languages

Context Free languages, CFL-PDA equivalence, Pumping Lemma and proof for existence of non- Context Free languages, CYK Algorithm, Deterministic CFLs, Chomsky Schutzenberger Theorem.

Turing Machines: recursive and recursively enumerable languages, Universality of Turing Machine, Church Thesis. Chomsky Hierarchy, Undecidability, Reducibility, Undecidability: Recursive and Recursively enumerable sets.

Complexity: Time and space complexity classes, hierarchy theorems, reductions and completeness, NP Completeness and PSPACE completeness, examples.

References:

1. M. Sipser, *Introduction to the Theory of Computation*, Thomson, 2001.
2. D. C. Kozen, *Automata and Computability*, Addison Wesley, 1994.
3. J. C. Martin, *Introduction to Languages and the Theory of Computation*, McGraw Hill, 2002
4. John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, *Introduction to Automata Theory, Languages, and Computations,* Pearson Education India, 3rd edition, 2008.

CS2019E PROFESSIONAL ETHICS

L	T	P	O	C
1	0	0	2	1

Total Sessions: Lecture 13

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Develop a clear understanding of human values and use it as the basis for all the activities.

CO2: Understand and follow the ethical aspects of the engineering profession.

CO3: Align with the Code of Ethics prescribed by ACM/IEEE in all professional activities.

CO4: Assimilate the elements of academic integrity and Honour Codes, and adopt them in all relevant activities.

Human Values

Morals, values and ethics – integrity – work ethic – service learning – civic virtue – sharing – honesty – courage – valuing time – cooperation – commitment – empathy – self-confidence – character.

Ethics in Professional Practice

Ethics in professional context – ethical basis of engineering activities – ethical responsibilities to consumers and customers – safety and risk – ethics in management of intellectual property – environmental matters and sustainability.

Code of Ethics and Academic Integrity

Code of Ethics as prescribed by professional bodies like ACM and IEEE – elements of Academic Integrity: honesty, trust, fairness, respect, responsibility – plagiarism as a violation of academic integrity – Honour Codes: specifying the expected ethical standards from the stakeholders of an organisation.

References:

1. R.S. Naagarazan, *A Textbook on Professional Ethics and Human Values*, 3rd edn., New Age International Pvt. Ltd., 2022.
2. A.F. Bainbridge, *Ethics for Engineers: A Brief Introduction*, CRC Press, 2021.
3. E.G. Seebauer and R.L. Barry, *Fundamentals of Ethics for Scientists and Engineers,* 1st ed., OUP India, 2008.
4. “ACM Code of Ethics and Professional Conduct”, acm.org. https://www.acm.org/code-of-ethics, (accessed June 1, 2023).
5. “IEEE Code of Ethics”*,* ieee.org. https://www.ieee.org/about/corporate/governance/p7-8.html, (accessed June 1, 2023).
6. International Center for Academic Integrity, academic integrity.org. https://academicintegrity.org/, (accessed June 1, 2023).

CS3001E COMPUTER NETWORKS

L	T	P	O	C
3	1	2	6	4

Total Sessions: Lecture 39, Tutorial 13 and Practical 26

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Analyse and evaluate datalink, network, transport, and application layer protocols

CO2: Apply the theory of basic networking concepts in the performance analysis of the protocol stack

CO3: Design, implement, and analyse solutions to problems in various layers of the protocol stack.

Computer Networks and Internet, The network edge, The network core, Network access, Delay and loss, Protocol layers and services, Application layer protocols, Web 2.0, Web 3.0, Socket Programming,

Transport layer services, UDP, TCP, New transport layer Protocols, Congestion control, New versions of TCP, Network layer services, Routing, Software Defined Networking, Openflow, P4 programming, IPv4, Routing in Internet, Router, IPV6, Multicast routing.

Link layer services, Error detection and correction, Multiple access protocols, ARP, Ethernet, Hubs, Bridges, Switches, MPLS, VLAN.

Wireless links, Mobility, Visible light communication techniques, Multimedia networking, Streaming stored audio and video, Real-time protocols, Network management.

References:

1. J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach Featuring Internet,* 6/e, Pearson Education, 2012.
2. L. L. Peterson and B. S. Davie, *Computer Networks, A systems approach,* 5/e, Morgan Kaufmann, 2011.
3. A. S. Tanenbaum and D. J. Wetherall, *Computer Networks*, 5/e, Pearson, 2013.

CS3002E COMPILER DESIGN

L	T	P	O	C
3	1	2	6	4

Total Sessions: Lecture 39, Tutorial 13 and Practical 26

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Interpret and examine the fundamental principles and techniques used in compiler design

CO2: Design the different phases of a compiler, specifically lexical analyzer, syntax analyzer, semantic analyser and intermediate code generator

CO3: Analyse the basic techniques and methods used for code optimization and machine code generation

Introduction to Programming language translation. Lexical analysis: Specification and recognition of tokens.

Syntax analysis: Top-down parsing-Recursive descent and Predictive Parsers. Bottom-up Parsing- LR (0), SLR, and LR (1) Parsers.

Semantic analysis: Type expression, type systems, symbol tables and type checking. Intermediate code generation: Intermediate languages. Intermediate representation - Three address code and quadruples. Syntax-directed translation of declarations, assignments statements, conditional constructs and looping constructs.

Runtime Environments: Storage organization, activation records. Introduction to machine code generation and code optimizations.

References:

1. A. V. Aho, M. S. Lam, R. Sethi, and J. D. Ullman, *Compilers: Principles, Techniques and Tools*, Pearson Education, 2007.
2. A. W. Appel and J. Palsberg, *Modern Compiler Implementation in Java*, Cambridge University Press, 2002.
3. D. Grune, K. van Reeuwijk, H. E Bal, C. J. H. Jacobs, and K. Langendoen. *Modern Compiler Design*, 2/e, Springer 2012.

CS3003E DESIGN AND ANALYSIS OF ALGORITHMS

L	T	P	O	C
3	1	2	6	4

Total Sessions: Lecture 39, Tutorial 13 and Practical 26

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Analyse a computational problem to identify it's computing requirements and recognize the algorithm design paradigms suited to solve the problem efficiently.
CO2: Design, evaluate and implement algorithms to meet the desired needs of a computational problem.
CO3: Classify computation problems based on their hardness by applying the notion of problem reductions.

Recap - Time and Space Complexity Analysis, Lower bounds for a problem - Searching and Sorting, Algorithms on Numbers - GCD; Binary Exponentiation; Modular Arithmetic ; Primality Testing, Order Statistics - Minimum and Maximum, Selection in Expected Linear Time and Worst Case Linear Time.

Divide and Conquer - Introduction, Multiplication of two numbers, Medians, Strassen Matrix Multiplication, Convex Hull, Maximum Sum Contiguous Subsequence, Fast Fourier Transform.

Greedy Algorithms - Introduction, Minimum Spanning Trees, Huffman Coding, Fractional Knapsack, Horn Formulas, Set Cover.

Dynamic Programming - Introduction, Longest Increasing Subsequence, Maximum Sum Contiguous Subsequence, 0/1 Knapsack, Matrix Chain Multiplication, Edit Distance, Optimal Binary Search Trees.

Linear Programming and Reductions, Complexity: complexity classes - P, NP, Co-NP, NP-Hard and NP-Completeness - Cook's theorem, NP-Complete Reductions, Coping with NP Completeness: Intelligent Exhaustive Search, Approximation Algorithms, Local Search Heuristics.

References:.

1. Dasgupta, Sanjoy, Christos Papadimitriou, and Umesh Vazirani, *Algorithms*. McGraw-Hill, 2008.
2. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 4/e, Prentice Hall India, 2022
3. A. Levitin, *Introduction to the Design & Analysis of Algorithms*, Pearson Education. 2003

CS3011E SOFTWARE ENGINEERING

L	T	P	O	C
3	1	2	6	4

Total Sessions: Lecture 39, Tutorial 13 and Practical 26

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Apply the basic concepts, principles and theories in software engineering to build software systems from scratch.
CO2: Analyse real world problems/requirements, and design systems by developing specifications and abstractions to make development of complex systems easy.
CO3: Apply rapid development technologies for developing software systems.

Introduction to Software Engineering – Reasons for software project failure – Similarities and differences between software and other engineering products. Software Development Life Cycle (SDLC) – Overview of Phases. Detailed Study of Requirements Phase.

Principles of software Design - Problem partitioning (subdivision) - Power of Abstraction. Concept of functional decomposition – UML diagrams - emphasis on usecase, class, object, sequence, activity diagrams. Introduction to architectural patterns - Design Patterns - Creational, Structural, Behavioural patterns. Coding - Programming Paradigms - Code Quality - Defensive programming – Methods and tools for version control. Testing - Quality Assurance - Types of testing – Specification of test cases - Test Automation – Code review and inspection process - Refactoring. Deployment.

Software Project Management: Software Process Models - Rapid Development - Continuous Integration and Continuous Development - Agile Approach: Agile Principles - Scrum - DevOps. Estimation - Size, Effort, Schedule. Current trends in Software Engineering: Introduction to Service Oriented Architecture - Web Services - REST Architecture - Microservices. No Code (Low Code) Platforms. Software Development Outsourcing. AIOps

References:

1. R. S. Pressman and Bruce Maxim, *Software Engineering: A Practitioner’s Approach*, 9/e, McGraw Hill, 2020.
2. T. C. Lethbridge and R. Laganiere, *Object Oriented Software Engineering*, 2/e, Tata McGraw Hill, 2004.
3. Gary Gruver and Tommy Mouser, *Leading the Transformation: Applying Agile and DevOps Principles at Scale*, 2015.
4. Kenneth Rubin, *Essential Scrum: A Practical Guide to the Most Popular Agile Process* Addison-Wesley, 2012
5. Jez Humble and David Farley, *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*, Addison-Wesley, 2010.
6. Steve McConnell, *Software Estimation: Demystifying the Black Art*, Microsoft Press, 2006.

CS3012E ARTIFICIAL INTELLIGENCE

L	T	P	O	C
3	1	2	6	4

Total Sessions: Lecture 39, Tutorial 13 and Practical 26

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Analyse a given problem and apply appropriate state space search strategy for solving the problem.

CO2: Apply appropriate knowledge representational schemes to represent the given problem and apply them for problem solving.

CO3: Demonstrate, apply and compare various approaches to machine learning from examples.

CO4 : Examine the role of Natural Language Processing, Machine perception and Robotics in Artificial Intelligence.

Artificial Intelligence: Introduction,History and Applications; Intelligent Agents;Solving problems by Searching: Structures and Strategies for state space search- Data driven and goal driven search,Uninformed Search strategies, Informed(Heuristic) Search Strategies, Heuristic Functions, Local Search Algorithms and Optimization Problems, Searching with Nondeterministic actions, Constraint satisfaction, Using heuristics in games- Minimax Search, Alpha Beta Procedure,Stochastic Games. Languages and Programming Techniques for AI, Implementation of state space search algorithms.

Knowledge representation: Knowledge based agents, Propositional calculus, First-Order Logic (Predicate Calculus),Inference in First-Order Logic, Forward and Backward chaining, Theorem proving by Resolution, Answer Extraction, AI Representational Schemes- Semantic Nets, Conceptual Dependency, Scripts, Frames, Planning, Planning and acting in the real world, Introduction to Logic programming, Production System examples in PROLOG.

Learning: Learning From Examples, Knowledge in Learning, Learning probabilistic Models, Artificial Neural Networks, Reinforcement Learning, Evolutionary Computation.

Communication, Perception and Action - Introduction to Natural Language Processing, Overview of Machine Perception, Robotics.

References:

1. S. Russell and P. Norvig, *Artificial Intelligence:A Modern Approach*, 4/e, Pearson Education, 2022.
2. G. F. Luger, *Artificial Intelligence- Structures and Strategies for Complex Problem Solving*, 6/e, 2021, Pearson Education.
3. E. Rich, K. Knight and S B Nair, *Artificial Intelligence*, 3/e, 2017, Tata McGraw Hill.
4. I. Bratko, *Prolog Programming for Artificial Intelligence*, 4/e, Addison Wesley, 2011.

CS3099E PROJECT

L	T	P	O	C
0	0	0	9	3

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Compile findings in a topic of interest relevant to the domain.
CO2: Choose a problem from the topic to study / solve.
CO3: Summarise the findings / solution as a technical report.
CO4: Present the findings / solution and defend the same.

The project must be carried out in the institute itself - individually or in a group.

CS4097E SUMMER INTERNSHIP

L	T	P	O	C
-	-	-	*	2

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Identify and appraise a relevant problem in the domain with the help of experts and / or literature.
CO2: Organise the existing knowledge / background relevant in solving the problem.
CO3: Select and use appropriate tools / techniques for solving the problem.
CO4: Prepare and present the professional work carried out towards the solution of a problem.

The internship may be carried out in industry or in academic organisations.

CS4021E LOGIC FOR COMPUTER SCIENCE

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Verify basic programs/systems (design) using SPIN and SMV

CO2: Formulate specifications for basic programs/systems (design) using logical formulas.

CO3: Demonstrate the connection between logics and automata

Omega regular languages and Buchi automata - properties, complementation, deterministic Buchi-regular languages. Buchi's complementation theorem.

Linear time temporal logic - Vardi-Wolper construction, model checking using Buchi automata. Introduction to SPIN. Software verification using SPIN.

Monadic second order logic, equivalence of MSO and Buchi automata, Safra construction. S2S. Rabin's tree automata theorem (no proof).

Computation Tree Logic, Binary decision diagrams, NuSMV model checker. Software verification using NuSMV.

References:

1. M. Huth and M. Ryan, *Logic in Computer Science: Modeling and Reasoning about Systems*, 2/e, Cambridge University Press, 2005.
2. C. Baler, *Principles of Model Checking*, MIT Press, 2008.
3. E. M. Clerke. Jr, O. Grumberg, and D. Peled, *Model Checking*, MIT Press, 2001.

CS4022E PROGRAM ANALYSIS

L	T	P	O	C
3	0	0	6	3

Total Sessions:Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate and examine various program analysis techniques

CO2: Examine different kind of problems and apply suitable analysis techniques

CO3: Build simple program analyzer tools.

Approaches to Program Analysis - Data-flow Analysis, Constraint based analysis, Abstract Interpretation, Type and effect systems.

Data-flow Analysis: Program Representations - Control Flow Graph, Analysis for computing : available expressions, reaching definitions, live variables. Applications of analysis in code improving transformations.

Data-flow Analysis Framework: Lattice theoretic modelling - Monotone frameworks, Distributive frameworks. Constant Propagation framework, Iterative algorithm for monotone frameworks - Maximal Fixed Point solution, Meet Over All Paths solution.

Type and effect Systems - control flow analysis, Introduction to inter procedural analysis and shape analysis.

References:

1. F. Nielsen, H. R. Nielson, and C. Hankin, *Principles of Program Analysis*, Springer 1999.
2. A. V. Aho, M. S. Lam, R. Sethi, and J. D. Ullman, *Compilers: Principles, Techniques, and Tools*, Pearson Education, 2007
3. S. Muchnick, *Advanced Compiler Design and Implementation*, Morgan Kaufmann, 1997

CS4023E FORMAL SEMANTICS

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate and appraise different approaches to formal semantics

CO2: Verify simple programs using axiomatic techniques

CO3: Demonstrate and examine semantics of simple programs using formal techniques.

Preliminaries, Mathematical foundations. Methods for Semantics Specification - Introduction to Operational, Denotational, and Axiomatic semantics.

Operational Semantics - Natural semantics, Structural operational semantics. Semantics of a simple imperative language - operational semantics of expressions, assignments, conditional and looping constructs.

Denotational Semantics: Semantic functions, Semantics of expressions, assignments, conditional and looping constructs. Fixed points, existence of fixed points.

Axiomatic Program Verification - Partial Correctness assertions, Inference systems. Extensions of the axiomatic system – Total correctness assertions.

References:

1. F. Nielson and H. R. Nielson, *Semantics with Applications: A Formal Introduction*, John Wiley & Sons Inc., 1992.
2. G. Winskel, *The Formal Semantics of Programming Languages - An Introduction*, MIT Press, Cambridge, MA, 1994.
3. D. Schmidt, *Denotational Semantics: A Methodology for Language Development*., Kansas State University, 2011

CS4024E COMPUTATIONAL COMPLEXITY

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Classify problems into appropriate complexity classes.

CO2: Design reductions from one problem to another for establishing their complexity relationships.

CO3: Apply interactive proofs for proving complexity class relationships.

Review of Complexity Classes: L, NL, P, NP, PSPACE and EXP, log-space and polynomial-time reductions, completeness, Hierarchy theorems, Savitch, and Immerman theorems.

Circuit complexity, P/Poly, NC and AC, P-completeness, polynomial hierarchy, Karp Lipton theorem, alternation, relationship between circuit depth and space complexity.

Randomized Complexity classes: Adleman's theorem, Sipser Gacs theorem, counting class, #P, Valiant’s Theorem, Toda’s theorem (no proof).

Arthur Merlin games, Graph Isomorphism problem, Goldwasser-Sipser theorem, Interactive Proofs, Shamir's theorem.

References:

1. S. Arora and B. Barak, *Computational Complexity: A Modern Approach,* Cambridge University Press, 2009.
2. C. H. Papadimitriou, *Computational Complexity*, 1/e, Addison Wesley, 1993.
3. R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.
4. V. Vazirani, *Approximation Algorithms*, 1/e, Springer, 2004.

CS4025E FORMAL VERIFICATION

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Model systems/programs (design) as Kripke structures

CO2: Formulate properties of systems/programs (design) as logical formulas

CO3: Verify simple systems/programs (design) using model checkers: SPIN, NuSMV and UPPAAL

Fundamentals

Significance of Formal Verification, Kripke Structures, Fair Discrete Systems (FDS) for representing programs, synchronous and asynchronous parallel composition for representing concurrent systems, Examples (such as Peterson's mutual exclusion algorithm) and their analysis by hand computation.

Logics for Specification

Review of propositional logic, Linear Temporal Logic - syntax, models, semantics, abbreviations - Eventually and Globally. Invariants, Safety, Liveness properties and their examples. Computation Tree Logic - syntax, FDS as models, semantics, Examples, Comparison of expressive power of CTL and LTL. CTL* - models, syntax and semantics. Model checking using a tool such as NuSMV or SPIN including modelling hardware circuits and concurrent systems.

Automata-based Technique

Büchi Automata - Nondeterministic Büchi Automata (NBW), Closure under Union, Intersection and Complementation, Omega Regular languages, LTL formula to NBW conversion by Vardi and Wolper. Blow-up in the translation. Complexity of LTL/CTL satisfiability checking and model checking using Büchi automata based algorithms.

Advanced Topics

Timed Automata, TCTL, syntax and semantics, TCTL model-checking, Usage of UPPAAL. Reduced Ordered Binary Decision Diagrams (BDDs), Using BDDs for symbolic model checking of Kripke structures for CTL and LTL properties. Bounded Model Checking using SAT solvers, Usage of SAL.

References:

1. Michael Huth, and Mark Ryan, *Logic in Computer Science: Modelling and Reasoning about Systems*, 2nd ed. Cambridge University Press, 2004.
2. Christel Baier and Joost-Pieter Katoen, *Principles of Model Checking,* MIT Press, 2008.

CS4026E VEHICULAR NETWORKS: THEORY TO PRACTICE

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate the basic theories and principles, technologies, standards, and system architecture of vehicular ad-hoc networks (VANET)
CO2: Interpret and examine specific communication technologies and routing protocols
CO3: Analyse, design, and evaluate vehicular communication protocols for various kinds of safety and infotainment applications

VANET Applications

Road safety status report by WHO and road issues, Connected autonomous vehicles, Introduction to Ad Hoc Wireless Networks, Vehicular network architecture: In domain, ad-hoc domain, Infrastructure domain, Vehicle-to-Everything(V2X), VANET applications: safety and non-safety, Vehicular network characteristics and challenges.

Vehicular Communication Networks

Vehicular Radio Access Technologies: DSRC, cellular, and Millimeter-Wave (mmWave-IEEE 802.11ad), DSRC standards: spectrum allocations and compatibility issues, PHY and MAC sublayer: IEEE 802.11p, enhanced distributed channel access (EDCA), WAVE protocol stack.

Networking Services and Application Layer

Network and transport layer - IEEE 1609.3. Routing: topology and position-based. Routing protocol - unicast, multicast, geocast, and broadcast. Challenges in VANET routing protocols. TCP and MPTCP in VANET - performance and challenges. Applications layer: Basic safety messages and message format.

Research Directions and VANET Experiments

Named Data Networking (NDN) enabled VANET, Software Defined Networking (SDN) based VANET, VANET Simulation, VANET Testbed - Setting up Raspberry Pi, Interfacing Race cars, establishing communication between vehicles

References:

1. H. Hartenstein and K. P. Laberteaux, *VANET: Vehicular Applications and InterNetworking Technologies,* Wiley, 2010.
2. P. H.-J. Chong, I. W.-H. Ho, *Vehicular Networks: Applications, Performance Analysis and Challenges,* Nova Science Publishers, 2019.
3. H. Moustafa, Y. Zhang, *Vehicular Networks: Techniques, Standards, and Applications,* CRC Press, 2009.
4. C. Sommer, F. Dressler, *Vehicular Networking,* Cambridge University Press, 2015.

CS4027E COMPUTATIONAL GEOMETRY

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Design algorithms for geometric problems and analyse these algorithms.

CO2: Prove the correctness and the theoretical guarantee of the geometric algorithms.

CO3: Formulate a geometric solution for an application and design an efficient algorithm with the most suitable data structure.

CO4: Develop advanced programs using recent Software tools (OpenGL/ CGAL)

Introduction to Computational Geometry, its applications. Preliminaries of asymptotic analysis and the notations to represent asymptotic complexity, Example of a geometric problem and its complexity analysis. Art Gallery problem and its associated theorems, Triangulation of a polygon and its theory, Area of a polygon. Polygon partitioning, Monotone partitioning, Trapezoidalization, Plane sweep, Partitioning to monotone mountains, Linear time triangulation. Introduction to Computational Geometric Algorithms Library (CGAL) and openGL and coding of simple programs with visualisation using QT.

Convex hull in two dimensions, Algorithms for convex hull with their complexity analysis: Extreme points, Extreme edges, Gift wrapping, Quickhull, Graham’s algorithm, Incremental algorithm, Divide and conquer. Applications of convex hull. Implement Convex Hull algorithms and one application using CGAL & visualisation using QT.

Voronoi diagram: Basic concepts, Definitions, Properties, Algorithm for construction of Voronoi diagram with its complexity analysis. Delaunay triangulation : Preliminaries and properties. Medial axis transform and its properties. Applications of Voronoi Diagram / Delaunay triangulation / Medial axis transform: Facility location, Reconstruction problem and its algorithms. Implement one application of Voronoi diagram/ Delaunay triangulation using CGAL & visualisation using QT.

Arrangements, Incremental algorithm, Voronoi diagram, Delaunay triangulation and convex hull in three dimensions and their applications. Implement one application of 3D Voronoi diagram / 3D Delaunay triangulation using CGAL & visualisation using QGL Viewer.

References:

1. J. O'Rourke, *Computational Geometry in C*, 2/e, Cambridge University Press, 1998.
2. M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, *Computational Geometry: Algorithms and Applications*, 2/e (revised), Springer-Verlag,2000.
3. F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, 1985.

CS4029E PRINCIPLES OF PROGRAMMING LANGUAGES

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate programming language concepts and constructs.

CO2: Develop the formal specification of programming language semantics.

CO3: Model programming language features using Lambda Calculus.

CO4: Design type systems with the objective of ensuring language safety.

Programming Languages: Concepts and Constructs. Untyped Arithmetic Expressions – Introduction, Semantics, Evaluation.

Untyped Lambda Calculus – Basics, Semantics. Programming in Lambda Calculus.

Typed Arithmetic Expressions – Types and Typing relations, Type Safety. Simply Typed Lambda Calculus – Function types, Typing relations, Properties of typing.

Extensions to Simply Typed Lambda Calculus – Unit type, Let bindings, Pairs, Records, Sums, Variants, References, Exceptions.

References:

1. B. C. Pierce, *Types and Programming Languages*, MIT Press, 2002.
2. D. A. Schmidt, *Programming Language Semantics. In Allen B. Tucker, Ed. Handbook of Computer Science and Engineering*, CRC Press, 1996.
3. L. Cardelli, *Type Systems.* In Allen B. Tucker, *Ed. Handbook of Computer Science and Engineering,* CRC Press, 1996.
4. M. L. Scott, *Programming Language Pragmatics*, 2/e, Elsevier, 2004.

CS4030E FOUNDATIONS OF PROGRAMMING

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate the fundamentals concepts and constructs in programming languages.

CO2: Design correct solutions using the techniques of Procedural Abstraction and Data Abstraction and prove the correctness of design.

CO3: Apply the concept of modular design to problem solving.

Programming methodology. Specification, Design, Coding. Separation of concerns such as correctness, efficiency, and maintainability. Fundamental concepts and constructs in programming languages.

Procedural Abstraction: Expressions - Naming and Environment - Combinators - Evaluation - Procedures - Substitution model - Conditional expression and predicates. Linear Recursion and Iteration - Tree recursion. Abstractions with Higher Order Procedures - Procedures as arguments - Constructing procedures – examples.

Data Abstraction: Hierarchical Data and Closure property - Symbolic Data - Data Directed Programming - Generic Operators - Combining data of different types.

Modular design: Modularity, Objects, and State: Local state - assignment, environment model for evaluation - frames, Modelling with mutable data. Encapsulation, inheritance, and polymorphism.

References:

1. H. Abelson and G. J. Sussman, *Structure and Interpretation of Computer Programs*, 2/e, Universities Press, 2005.
2. Companion Site to the Textbook. Available at http://mitpress.mit.edu/sicp/ Accessed on December 1, 2010.
3. T. C. Lethbridge and R. Laganiere, *Object Oriented Software Engineering*, 1/e, Tata McGraw Hill, 2004.

CS4031E NETWORK SECURITY

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able:

CO1: To examine contemporary security issues in the field of networks and provide solutions

CO2: To analyse network protocols and understand the underlying vulnerabilities

CO3: To design and develop secure solutions to build a safe and secure networked system

Network protocols and Threats to Network stack

Network Concepts – Transmission media, Protocol layers, Addressing and Routing; Security terminologies
Network Security attacks – Port Scanning, DoS, DDoS, Bots and Botnets, Remote Access attacks, LAN attacks, Honeypot/Honeynet
Defensive techniques – Firewalls, Intrusion Detection System, NAT

Authentication, Access Control, and Cryptography

Review of cryptographic algorithms - Authentication protocols – Password based, Challenge based Wireless authentication methods, Multi Factor authentication - Access Control policies – role based
Key exchange protocols

Secure Network Communication protocols

Secure protocols - SSH, SSL 3.0, TLS1.2, TLS 1.3, IPSec HTTP1.1 vs. HTTP2, HTTP3, HTTPS, Attacks on SSL/TLS – SSL stripping, Drown and Poodle attack

Application level security

Secure Email: PGP, DKIM and Spams - Secure DNS - Web security: Review of OWASP vulnerabilities
Review of open source security tools

References:

1. C. Kaufman, R. Perlman, and M. Speciner, *Network Security: Private Communication in a Public World*, 2/e, Pearson India, 2017.
2. J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach*, 8th ed. Pearson Education Asia, 2020.
3. W. Stallings, *Cryptography and Network Security - Principles and Practice,* 8/e, Pearson India, 2023.
4. Lyon Gordon, *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*, Nmap Project, 2009.
5. Wolfgang Osterhage, *Wireless Network Security*, 2ed ed. CRC Press, 2021.
6. James Forshaw, *Attacking Network Protocols: A Hacker's Guide to Capture, Analysis, and Exploitation*, No Starch Press, 2017.

CS4032E COMPUTER SECURITY

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate the design of modern cryptosystems

CO2: Design cryptographic protocols

CO3: Analyse and evaluate security of computer networks

Review of Cryptography Fundamentals - Symmetric and Asymmetric Cryptosystems, Cryptographic Hash, Digital Signatures.

Cryptographic Protocols for Authentication - Authentication Protocols: One way and Mutual Authentication, Challenge Response protocols, Needham Schroeder protocol. Interactive proof systems, Zero Knowledge Proof systems.

Network Security - Security at different layers – IPSec / SSL-TLS. Cloud security.

Principles of Secure Design - Software vulnerabilities - Buffer and stack overflow, Phishing. Malware - Viruses, Worms and Trojans. Security problems in network domain - Defense Mechanisms.

References:

1. B. Menezes, *Network security and Cryptography*, Cengage Learning India, 2010.
2. B. A. Forouzan and Mukhopadhyay, *Cryptography and Network Security*, 2/e, Tata McGraw Hill, 2011.
3. D. Gollmann, *Computer Security,* 3/e, John Wiley and Sons Ltd., 2014.
4. C. Kaufman, R. Perlman, and M. Speciner, *Network Security: Private Communication in a Public World*, 2/e, Pearson India, 2017.
5. W. Stallings, *Cryptography and Network Security - Principles and Practice,* 8/e, Pearson India, 2023.

CS4033E QUANTUM COMPUTATION

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate and apply the quantum computation model

CO2:. Analyse and apply quantum computation based algorithms

CO3: Design simple key distribution algorithms

CO4: Analyse the quantum information theoretic models

Course Contents:

Review of Linear Algebra. The postulates of quantum mechanics. Review of Theory of Finite Dimensional Hilbert Spaces and Tensor Products, spectral theorem for Hermitian and Normal operators.

Complexity classes. Models for Quantum Computation. Qubits. Single and multiple qubit gates. Quantum circuits. Bell states. Single qubit operations. Controlled operations and measurement. Universal quantum gates. Quantum Complexity classes and relationship with classical complexity classes.

Quantum Algorithms – Quantum search algorithm - geometric visualisation and performance. Quantum search as a quantum simulation. Speeding up the solution of NP Complete problems. Quantum search as an unstructured database. Grover's and Shor's Algorithms.

Quantum Cryptography and Coding: Quantum key distribution, Introduction to post quantum cryptography, Introduction to Quantum Coding Theory. Quantum error correction. The Shor code.

References:

1. M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge, UK, Cambridge University Press, 2002.
2. J. Gruska, *Quantum Computing*, McGraw Hill, 1999.
3. A. Peres, *Quantum Theory: Concepts and Methods*, Springer, 1993.

CS4034E ADVANCED COMPUTER NETWORKS

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate the functioning of various protocols in a network protocol suite.

CO2: Analyse communication protocol for simple network design requirements.

CO3: Address security issues in networking and describe their standard solutions.

Introduction- Internet design philosophy, Layering and end to end design principle. MAC protocols for high speed LANS, MANs, Wireless LANs and mobile networks, VLAN. Fast access technologies.

IPv6: Why IPv6, Basic protocol, extensions and options, Support for QoS, Security, Neighbour discovery, Auto-configuration, Routing. Changes to other protocols. Application Programming Interface for IPv6, 6bone. IP Multicasting, Wide area multicasting, Reliable multicast. Routing layer issues, ISPs and peering, BGP, IGP, Traffic Engineering, Routing mechanisms: Queue management, Packet scheduling. MPLS, VPNs

TCP extensions for high-speed networks, Transaction-oriented applications. New options in TCP, TCP performance issues over wireless networks, SCTP, DCCP.

DNS issues, other naming mechanisms, Overlay networks, P2P networks, Web server systems, Web 2.0, Internet traffic modelling, Internet measurements. Security – Firewalls, Unified threat Management System, Network Access Control.

References:

1. A. Farrel, *The Internet and its protocols a comparative approach*, Elsevier, 2005
2. M. Gonsalves and K. Niles, *IPv6 Networks*, McGraw Hill, 1998.
3. W. R. Stevens, *TCP/IP Illustrated, Volume 1: The protocols*, Addison Wesley, 1994.
4. G. R. Wright, *TCP/IP Illustrated, Volume 2: The Implementation,* Addison Wesley, 1995.

CS4035E PROBABILISTIC METHODS IN COMBINATORICS

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Apply basic concepts of counting and probability to solve graph-theoretic problems.

CO2: Prove non-constructive bounds for graph models for application problems. .

CO3: Apply LLL and semi-random methods in combinatorial models of application problems.

Introduction to Probabilistic Methods

Review of probability, counting, discrete probability, conditional probability, expectation and variance.
The Basic Methods - Lower Bound for Ramsey Number, Tournaments and Dominating Sets, Set Systems.

First Moment Methods and Alterations

Linearity of expectations, indicator random variables first moment principle, Markov’s inequality, 2-colorable hypergraphs, Hamiltonian paths in tournaments, splitting graphs, Turán’s theorem and independent sets, crossing number of graphs, alterations, bounds on Ramsey numbers, independent sets, extremal results on 2-colorable hypergraphs.

Random Graphs and Thresholds

Second moment method, variance, Chebyshev’s Inequality, covariance, estimating binomial coefficient, random graphs and thresholds, connectivity, Hamiltonicity, clique Number, Rodl nibble method, Chernoff Bound.

Lovasz Local Lemma and Applications

Lovasz Local Lemma, Applications to hypergraph Colouring, directed cycles, latin transversals, Ramsey number, constructive LLL, Moser’s entropy compression argument, derandomization, the method of conditional probabilities.

References:

1. Noga Alon and Joel H. Spencer, *The probabilistic method*, 4th ed. John Wiley & Sons, 2016.
2. Michael Molloy and Bruce Reed, *Graph Colouring and the Probabilistic Method*, Vol. 23. Springer Science & Business Media, 2002.
3. M. Mitzenmacher, E. Upfal, *Probability and Computing: Randomised Algorithms and Probabilistic Analysis*, Cambridge University PressCambridge University Press, 2005.

CS4036E ALGORITHMS IN OPTIMIZATION

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Formulate a given problem statement into an appropriate optimization problem for standard form.

CO2: Select appropriate algorithms for a formulated problem.

CO3: Demonstrate the functioning standard algorithms for solving various classes of optimization problems.

Mathematical Foundations

Review of multivariable differential calculus, Jacobian and Hessian, second order Taylor approximation.
Affine and convex sets, convex functions and mathematical properties, convex optimization problems, Lagrangian duality, linear and quadratic optimization problems.

Unconstrained Optimization

Newton’s method, Levenberg Marquardt method, gradient descent, convergence analysis, conjugate gradient algorithm, least squares.

Constrained Optimization

Linear programming, interior point method, Karmarkar’s algorithm, non-linear equality constraints, Lagrange conditions, Karush Kuhn Tucker conditions, stochastic gradient descent, conditions for convergence in convex optimization problems.

References:

1. E. K. P. Chong and S. H. Zak, *An Introduction to Optimization*, 4th ed., Wiley, 2013.
2. M. J. Kochenderfer, T. A. Wheeler, *Introduction to Optimization*, MIT Press, 2019.
3. S. Boyd, L. Vandenberghe, *Convex Optimization,* Cambridge University Press, 2004.

CS4037E ALGORITHMIC GRAPH THEORY

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Formulate problems using graph models.

CO2: Prove the structural properties of the graph models.

CO3: Design algorithmic solutions to problems modelled as graph problems.

Searching and Asymptotic Analysis

Review of graph theory – spanning trees, shortest paths. Connectivity, Ear Decompositions.
Review of Linear Algebra - Vector Space, Rank- Nullity Theorem.

Network Flows

Network Flows - Max flow min cut theorem, Algorithms of Ford-Fulkerson, Edmond Karp, preflow-push algorithms. Applications to graph theory - Konig's theorem, Hall's theorem, Menger's theorem

Linear Programming

Linear programming, basic feasible solutions, Simplex algorithm, Primal dual theorem (no proof) , duality based algorithm design. Applications to Network flow and other combinatorial problems, primal dual approximation algorithms.

Matching

Matching: Tutte-Berge Formula, Gallai-Edmonds Decomposition, Factor-critical graphs, Edmonds' ‘Blossom’ algorithm. Maximum matching in Bi-partite Graphs: Hopcroft Karp algorithm, Hungarian algorithm.

References:

1. D. Jungnickel, *Graphs Networks and Algorithms*, Springer 2005
2. Dexter Kozen. *The design and analysis of algorithms*. Springer Science & Business Media, 1992.
3. R. Diestel. *Graph Theory*, Springer, 2010.
4. C. E. Leiserson, R. L Rivest, C. Stein, Introduction to Algorithms, 4th ed., MIT Press, 2022.

CS4038E COMPUTATIONAL ALGEBRA

L	T	P	O	C
3	0	0	6	3

Total Sessions:Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate the standard algorithms with proof/derivation for classical computing problems in algebra and number theory for standard problems.
CO2: Apply methods studied under CO1 to solve simple application problems.
CO3: Design solutions to problems that require minor modifications to the standard algorithms described in CO1 for a solution, and prove the correctness of the proposed solutions.

Review of groups and rings and vector spaces, Euclid's algorithm, Structure of the ring Zn Algorithms for computation in the ring Zn - modular inversion, exponentiation, Chinese remaindering, RSA system, primality testing, Miller Rabin test, finding primitive elements modulo prime. Applications to cryptography.

Factorization of polynomials over finite fields Cantor Zassenhaus algorithm, Fourier Transform algorithm for finite fields. Linear codes and minimum distance decoding, decoding of Reed Solomon codes.

Review of linear algebra, Shortest vector problem, Gauss algorithm. Lattice basis and reductions, LLL algorithm.

Multivariate polynomials, Hilbert's Nullstellensatz, Grobner basis, Buchberger’s algorithm.

References:

1. V. Shoup, *A computational Introduction to Number Theory and Algebra,* Cambridge University Press, 2005.
2. H. Delfs and H. Knebl, *Introduction to Cryptography*, Springer, 1998.
3. J. von zur Gathen, *Modern Computer Algebra,* Cambridge University Press, 2003.
4. W. C. Huffman and V. Pless, *Fundamentals of Error Correcting Codes*, Cambridge University press, 2003,
5. D, Cox, David A. J. Little, John, O. Donal (1997). *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer 1997.

CS4038E COMPUTER ARCHITECTURE

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Analyse and quantify the performance of an architectural design using pre-existing matrices.

CO2: Demonstratethe various models for instructional level and data parallelism.

CO3: Demonstrate the mechanisms for thread level parallelism in multi-core architectures.

Fundamentals - Technology trend -performance measurement - Comparing and summarising performance quantitative principles of computer design - Amdahl's law - instruction set principles - memory addressing - type and size of operands - encoding an instruction set - role of compilers - Review of pipelining - MIPS architecture Memory hierarchy design - reducing cache misses and miss penalty, reducing hit time Main memory organisation - virtual memory and its protection.

Instruction level parallelism and its limits - static and dynamic scheduling - static and dynamic branch prediction - multiple issue processor - multiple issue with dynamic scheduling-hardware based speculation - limitation of ILP-Multithreading.

Multiprocessor and thread level parallelism- classification of parallel architecture-models of communication and memory architecture-Symmetric shared memory architecture-cache coherence protocols-distributed shared memory architecture-directory based cache coherence protocol - synchronisation Memory consistency-relaxed consistency models.

Data Level Parallelism-Vector processors-SIMD extensions, GPU, GPU and CUDA, Overview of CUDA C; threads, blocks and grids, warps, different GPU memories, Kernel-Based Parallel Programming, Request level parallelism, Domain specific Architecture.

References:

1. J. L Hennessy and D. A. Patterson, *A.Computer Architecture: A Quantitative approach*, 6/e, Morgan Kaufmann, 2017.
2. D. A. Patterson and J. L. Hennessy, *Computer Organisation and Design: The Hardware/ Software Interface*, 5/e, Harcourt Asia Pte Ltd (Morgan Kaufman), 2014.

CS4040E MATHEMATICAL FOUNDATIONS OF MACHINE LEARNING

L	T	P	O	C
3	0	0	6	3

Total Sessions:Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Solve simple problems involving various mathematical models.

CO2: Prove simple properties of algorithms and methods relating to data analysis problems.

CO3: Decide on the right mathematical model for a simple problem situation.

Statistical Methods

Review of Probability Theory: Discrete and Continuous Random Variables, Joint and Marginal Distributions, Markov, Chebyshev, Jensen and Hausdorff Inequalities, Law of Large Numbers, Central Limit Theorem (No proof). Classification and Estimation: Bayes classifier, maximum likelihood and Bayesian estimation techniques.

Linear Algebraic Methods

Review of Linear Algebra: Vector spaces, Rank Nullity Theorem, Metric and Normed Linear Spaces, Inner product spaces, Gram Schmidt Orthogonalization, Projections and Orthogonal Projections, Introduction to Hilbert spaces. Orthogonal Decomposition algorithms: Eigen Decomposition, Singular Value Decomposition, Principal component analysis, LU, QR, Cholesky Decompositions, Least Squares Approximation.

Introduction to Unconstrained Optimization

Review of Multivariate Analysis: Sequences and Limits, Differentiability, Level Sets and Gradients, multivariate Taylor Series. Unconstrained Optimization: Conditions for Local Minimizers of Continuously Differentiable Functions, Gradient Search, Analysis of Newton's method, Levenberg-Marquardt Modification, Quasi-Newton Methods, Rank One Correction Formula, DFP and BFGS Algorithms.

Introduction to Convex Optimization

Constrained Optimization: Tangent and Normal Spaces, Lagrange Condition, Second-Order-Conditions, Karush-Kuhn-Tucker Condition. Convex Optimization: Lagrange and Fenchel Duality, Proximal Algorithms, ADMM Algorithm.

References:

1. R. S. Ross, *Introduction to Probability and Statistics for Engineers and Scientists,* Academic Press, 2014.
2. K. Hoffman and R. Kunze. *Linear Algebra*, 2/e, Prentice Hall of India, 1990.
3. E. K. P. Chong and S. H. Zak, *An Introduction to Optimization,* 2/e, John Wiley & Sons, 2004.
4. B. Stephen and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

CS4041E INTRODUCTION TO MACHINE LEARNING

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Compute the VC dimension of a given learning model and analyse learnability.

CO2: Demonstrate algorithms for solving convex learning and prove correctness.

CO3: Design classifiers using various learning models for a given problem setting.

Fundamental Concepts

The PAC learning model, agnostic learning, uniform convergence, No Free Lunch Theorem, VC dimension. Fundamental Theorem of learning, non-uniform learning, learnability of linear models, hyperplane separators, boosting, validation.

Convex Learning

Convex learning problems, Lipshitz and smooth bounded learning problems, regularisation and stability, Tikhonov regularisation, stability of stochastic gradient descent, support vector machines, kernel methods,

Selected Topics

Multiclass predictors, decision trees, random forest classifiers, nearest neighbour, feed forward neural networks, clustering methods, dimensionality reduction, maximum likelihood estimation.

References:

1. S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms,* Cambridge University Press, 2014.
2. M. Mohri, A. Rostamizadeh, A. Talwalker, *Foundations of Machine Learning*, MIT Press, 2018.
3. C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

CS4042E RANDOMIZED ALGORITHMS

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Apply probabilistic methods to analyse time complexity of randomised data structures.

CO2: Apply random walks in solving problems on volume estimation.

CO3: Design polynomial time randomised algorithms for combinatorial problems.

Tail Bounds

Review of discrete probability spaces, probabilistic method, Markov and Chebyshev inequalities, moments of a random variable, Chernoff bounds, Martingales, Hoeffding and Azuma inequalities.

Randomised Data Structures and Algorithms

Randomised data structures and algorithms : Skip lists, hashing, randomised min-cut, verifying matrix multiplication, randomised quicksort, randomised selection, coupon collector's algorithm, randomised pattern matching. number theoretic algorithms: primality testing - Miller Rabin test.

Markov Chain Monte Carlo Methods

Markov chains and random walks, stationarity, Markov chain Monte Carlo (MCMC) methods, volume estimation, randomised complexity classes.

Derandomization

Probabilistic method and derandomization, the basic counting argument, expectation argument, derandomization using conditional expectation, sample and modify method, second Moment method, constructive Lovasz Local Lemma (LLL), Schwartz–Zippel lemma.

References:

1. R. Motwani and P. Raghavan, *Randomized Algorithms,* Cambridge University Press, 1995.
2. M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomization and probabilistic techniques in algorithms and data analysis,* 2nd ed., Cambridge University Press, 2017.
3. C. H. Papadimitriou, *Computational Complexity*, Addison Wesley, 1994.
4. M. Jerum, *Counting, Sampling and Integrating: Algorithms and Complexity,* 3rd ed., Birkhauser, 2003.

CS4043E INTRODUCTION TO PARAMETERIZED ALGORITHMS

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

CO1: Classify the computational problems into tractable and intractable problems

CO2: Demonstrate the techniques for the design of fixed-parameter tractable algorithms

CO3: Design and analyse the running time of fixed-parameter tractable algorithms for some of the classical NP-Hard graph problems

Review and Introduction

Review of complexity classes - P, NP, Co-NP, NP-Hard and NP-complete problems. Strategies for Coping with hard algorithmic problems; Exact exponential algorithms and the notion of fixed parameter tractability. Parameterizations and Parameterized problems- Satisfiability problem, Vertex cover - An illustrative example, Art of problem Parameterization.

Kernelization, Depth bounded search trees and Iterative Compression

Algorithmic methods: Data reduction and problem kernels. Depth bounded search trees for Maximum satisfiability, Cluster editing, Vertex cover, 3-Hitting set, Dominating set in Planar graphs. Iterative Compression technique for Vertex Cover and Feedback Vertex Set

Dynamic Programming and Randomized Algorithms

Dynamic programming – Set cover, Tree structured variants of set cover and Steiner Trees. Randomized methods in Parameterized algorithms – Simple randomized algorithm for Vertex cover, Feedback Vertex Set, Color coding algorithm for Longest path.

Tree decomposition

Path and Tree decomposition – Dynamic Programming on graphs of bounded treewidth – Treewidth and Monadic second-order logic, Graph searching, interval and chordal graphs. Computing treewidth – Balanced separators and separations – FPT approximation algorithm for treewidth.

References:

1. M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh, *Parameterized Algorithms*, Springer, June 2015.
2. R. Niedermeier, *Invitation to fixed-parameter algorithms,* Oxford university press, 2006.
3. R. G. Downey and M. R. Fellows, *Fundamentals of Parameterized Complexity,* Springer, 2013.

CS4044E INTRODUCTION TO PARAMETERIZED COMPLEXITY THEORY

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Classify the Parameterized problems into fixed parameter tractable and fixed parameter intractable

CO2: Illustrate the techniques for the proving W[1] and W[2] hardness of Parameterized problems

CO3: Analyse the given NP-hard problem by choosing appropriate parameters and design a fixed parameter tractable algorithm or prove that it is fixed parameter intractable

Fixed Parameter Tractability - Introduction, Parameterized Problems and Fixed-Parameter Tractability, Reductions and Parameterized Intractability - Fixed-parameter Tractable reductions - The class para-NP, and The class XP.

Parameterized complexity theory – Complexity class W[1] - Concrete parameterized reductions – W[1]-hardness proofs – Further reductions and W[2]-hardness. Lower bounds and the complexity class M[1], lower bounds and linear FPT-reductions.

Kernelization and Linear Programming Techniques, Tree Decomposition of Graphs, Computing Tree Decompositions, Algorithms on structures of Bounded Tree width. Applications of Courcelle's Theorem - Tree width reductions and Graph Minors

Machine models, limited nondeterminism, and bounded FPT. Selected case studies: Graph modification problems and Capacitated Vertex cover, Constraint Bipartite vertex cover, Graph coloring, Crossing number, Power dominating set.

References:

1. J. Flum and M. Grohe, *Parameterized Complexity Theory*, Springer, 2006.
2. R. Niedermeier, *Invitation to fixed-parameter algorithms,* Oxford university press, 2006.
3. R. G. Downey and M. R. Fellows, *Fundamentals of Parameterized Complexity,* Springer, 2013.

CS4045E IMAGE PROCESSING

L	T	P	O	C
3	0	0	6	3

Total Lecture Sessions: 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate and apply the basics of digital image representation.

CO2: Sketch the different Image transformations in the frequency domain.

CO3: Apply Image enhancement techniques in the spatial and the frequency domains.

CO4: Demonstrate and apply image segmentation techniques and image compression on digital images.

CO5: Apply the image processing techniques and algorithms for designing and building applications in various domains like medical imaging and robot vision.

Fundamentals of Image processing: Digital image representation, Elements of Digital image processing systems, Image model, Sampling and Quantization, Basic relations between pixels. Image transforms: One dimensional Fourier transform, Two dimensional Fourier transform, Properties of two dimensional Fourier transform. Walsh transform, Hadamard transform, Discrete cosine transform, Haar transform, Slant transform.

Image enhancement techniques: Spatial domain methods, Frequency domain methods, Intensity transform, Histogram processing, Image subtraction, Image averaging, Smoothing filters, Sharpening filters, Spatial masks from frequency domain.

Image Segmentation: Thresholding: Different types of thresholding methods, Point detection, Edge detection: Different types of edge operators, Line detection, Edge linking and boundary detection, Region growing, Region splitting, Region Merging.

Image Data Compression: Fundamentals, Compression models, Error free compression, Lossy Compression, Image compression standards. Applications of Image Processing: Medical imaging, Robot vision, Character recognition, Remote Sensing.

References:

1. R. C. Gonzalez and R. E. Woods, *Digital Image Processing,* Pearson Education, 2018.
2. A. K. Jain*, Fundamentals of Digital Images Processing,* Pearson Education India, 2015

CS4046E DEEP LEARNING FOR COMPUTER VISION

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate, analyse and apply the fundamental techniques used in Computer vision.

CO2: Design, implement and apply CNN & RNN based networks for Object Recognition and Image and video processing.

CO3: Demonstrate the basics of Deep generative models and apply them for building applications.

Fundamentals of Computer Vision
Computer Vision Basics : Introduction to Image Formation, Capture and Representation; Linear Filtering, Correlation, Convolution,Edge, Blobs, Corner Detection; Color, Texture, Segmentation, Scale Space and Scale Selection; SIFT, SURF; HoG, LBP, etc, Bag-of-words, VLAD; RANSAC, Hough transform; Pyramid Matching; Optical Flow, Object Recognition.

Deep Learning for Object Recognition
Review of Deep Learning, Multi-layer Perceptrons, Backpropagation,Introduction to CNNs; Evolution of CNN Architectures: AlexNet, ZFNet, VGG, InceptionNets, ResNets, DenseNets, Visualization of Kernels; Backprop-to-image/Deconvolution Methods; Deep Dream, Hallucination, Neural Style Transfer; CNNs for Recognition and Verification (Siamese Networks, Triplet Loss, Contrastive Loss, Ranking Loss); CNNs for Detection: Background of Object Detection, R-CNN, Fast R-CNN, Faster R-CNN, YOLO, SSD, RetinaNet; CNNs for Segmentation: FCN, SegNet, U-Net, Mask-RCNN.

Image and Video understanding and Description Generation
Review of RNNs; CNN + RNN Models for Video Understanding: Spatio-temporal Models, Action/Activity Recognition, Introduction to Attention Models in Vision; Vision and Language: Image Captioning, Visual QA, Visual Dialog; Spatial Transformers; Transformer Networks, Transformer model: Introduction of Attention Mechanism, Queries, Keys, and Values of Attention Network, Self-Attention and Positional Encodings, Attention-Based Sequence Encoder, Coupling the Sequence Encoder and Decoder, Cross Attention in the Sequence-to-Sequence Model, Multi-Head Attention, The Complete Transformer Network, BERT based models, Nemo, Self-supervision techniques, masked language modelling, autoregressive modelling.

Deep Generative Models and Applications
Review of Deep Generative Models: GANs, VAEs; Other Generative Models: PixelRNNs, Normalising Flows, Applications: Image Editing, Inpainting, Superresolution, 3D Object Generation, Security; Variants: CycleGANs, Progressive GANs

References:

1. Ian Goodfellow, Yoshua Bengio, Aaron Courville, Deep Learning, MIT Press, 2016.
2. Michael Nielsen, Neural Networks and Deep Learning, Determination Press, 2015.
3. Yoshua Bengio, Learning Deep Architectures for AI, now Publishers Inc., 2009.
4. Richard Szeliski, Computer Vision: Algorithms and Applications, Springer, 2010.
5. Simon Prince, Computer Vision: Models, Learning, and Inference, Cambridge University Press, 2012.
6. David Forsyth, Jean Ponce, Computer Vision: A Modern Approach, PHI Learning Pvt. Ltd., 2002.

CS4047E ADVANCED COMPUTER ARCHITECTURE AND SECURITY

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Critically analyse and compare different architectures and design solutions.

CO2: Analyse different hardware attacks from the architectural side.

CO3: Demonstrate and appraise the state of art machine architectures.

Multi-core and GPU architectures

Thread level parallelism: multithreaded processors and multicore processors concept, data level parallelism (DLP) - vector architecture, design issues and implementation, cache coherency protocols, Network On Chip (NOC), graphical processing unit (GPU) - GPU architecture, programming examples using OpenMP and CUDA

Hardware security

Side channel attacks: prime+probe, flush+reload, mitigation techniques, Prefetcher based attacks, PASS-P scheduling - performance and security tradeoffs

State-of-the-art architectures

Heterogeneous ISA architectures: scheduling mechanisms in heterogeneous ISAs, performance efficient dynamic cores, energy efficient dynamic cores, advanced cache replacement policies, application specific integrated circuit architectures (ASIC), Spectre and meltdown attacks

References:

1. D. A. Patterson and J. L. Hennessy, *Computer Organisation and Design: The Hardware/Software Interface*, 6/e, Morgan Kaufmann.
2. V. P. Heuring and H. F. Jordan, *Computer System Design and Architecture*, Prentice Hall, 2003
3. John Paul Shen and Mikko H. Lipasti, *Modern Processor Design: Fundamentals of Superscalar Processors*, 2013

CS4048E CLOUD COMPUTING

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Articulate the virtualization concepts

CO2: Identify the architecture, service models and deployment of Cloud

CO3: Master the programming aspects of Cloud

New Computing Paradigms & Services: Cloud computing , Edge computing , Grid computing , Utility computing , Cloud Computing Architectural Framework, Cloud Deployment Models, Virtualization in Cloud Computing, Parallelization in Cloud Computing, Cloud Economics , Metering of services, Security for Cloud Computing, Security threats and solutions in clouds

Cloud Service Models: Software as a Service (SaaS), Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Service Oriented Architecture (SoA), Elastic Computing, On Demand Computing, Cloud Architecture, Introduction to virtualization, Types of Virtualization, Virtual Machine Migration, Case studies- Xen, VMware, Eucalyptus, Amazon EC2.

Containers: Namespaces, cgroups, frameworks, VM checkpointing and Cloning, Software Defined Networks. Cloud storage: Key-value storage, Semi-structured data storage, Application specific optimizations, Caching, Case studies- Docker, Kubernetes,

Introduction to Mapreduce, Information retrieval through Mapreduce, Hadoop File System, GFS, Page Ranking using Map Reduce, Internet of Things and cloud, mobile cloud computing, Fog Computing, Serverless computing.

References:

1. Rajkumar Buyya, James Broberg, Andrzej M. Goscinski, *Cloud Computing: Principles and Paradigms, Editors:,* Wiley, 2011
2. Ronald L. Krutz, Russell Dean Vines,*Cloud Security: A Comprehensive Guide to Secure Cloud Computing*, Wiley- India,2010
3. Barrie Sosinsky,*Cloud Computing Bible*, Wiley-India, 2010

CS4049E DISTRIBUTED COMPUTING

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Articulate the fundamental properties of distributed systems and their implications for system design

CO2: Analyse the issues involved in distributed computing systems

CO3: Compare a variety of programming models and abstractions for distributed system

Characteristics of Distributed Systems, Distributed systems Versus Parallel systems, Models of distributed systems, Happened Before and Potential Causality Model, Models based on States, Logical clocks, Vector clocks, Verifying clock algorithms, Direct dependency clocks.

Mutual exclusion using Timestamps, Distributed Mutual Exclusion (DME) using timestamps, token and Quorums, Centralized and distributed algorithms, proofs of correctness and complexity analysis. Drinking philosophers problem, Dining philosophers problem under heavy and light load conditions. Implementation and performance evaluation of DME algorithms.

Leader election algorithms, Global state detection, Global predicates, Termination Detection, Control of distributed computation, disjunctive predicates. Performance evaluation of leader election algorithms on simulated environments.

Self stabilization, knowledge and common knowledge, Distributed consensus, Consensus under Asynchrony and Synchrony, Checkpointing for Recovery, R- Graphs

References:

1. V. K. Garg, *Elements of Distributed Computing*, Wiley & Sons, 2002.
2. S. Ghosh, *Distributed Systems An Algorithmic Approach, Chapman & Hall,* CRC Computer and Information Science Series, 2006.
3. A. S. Tanenbaum and M. V. Steen, *Distributed Systems: Principles and Paradigms,* 2/e, Pearson, 2007.
4. G. Coulouris, J. Dollimore, T. Kindberg, and G. Blair, *Distributed Systems: Concepts and Design,* 5/e, Addison Wesley, 2011.
5. R. Chow and T. Johnson, *Distributed Operating Systems and Algorithms*, Addison Wesley, 2002.

CS4050E NATURAL LANGUAGE PROCESSING

L	T	P	O	C
3	0	0	6	3

Total Sessions:Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Discuss various challenges associated with natural language understanding such as syntactic ambiguity, semantic ambiguity, discourse analysis, and pragmatics.
CO2: Formally describe the various syntactic structures in natural language processing.
CO3: Apply natural language processing tools and libraries (such as nltk, PyTroch, TensorFlow) and develop models for various NLP applications.

Introduction to Natural Language Processing, Representation and Understanding, Linguistic Background, Grammars and Parsing, Top down and Bottom up Parsers. Constituency Parsing, Dependency Parsing, Logical Representations of Sentence Meaning, Relation and Event Extraction, Word Senses and WordNet.

Semantic Role Labelling, Coreference Resolution, Discourse Coherence, Sequence Labeling for Parts of Speech and Named Entities: Hidden Markov Model, Maximum Entropy Markov Model, Conditional Random Fields; Vector Semantic and Embedding, Classifiers for NLP Tasks, N-grams Language models.

Neural language Models: Word2vec, Glove; Deep Learning Models for NLP: Recurrent neural networks, Attention Mechanism, Transformers and Pretrained Language Models.

NLP Applications: Machine Translation, Question Answering and Information Retrieval, Automatic Speech Recognition and Text-to-Speech, Sentiment Analysis.

References:

1. J. Allen, *Natural Language Understanding*, 2/e, Pearson Education, 2003.
2. T. Siddiqui and U. S. Tiwary, *Natural Language Processing and Information Retrieval*, Oxford University Press, 2008.
3. D. Jurafsky and J. H. Martin, *Speech and Language Processing*, Third Edition, 2023.

CS4051E INTRODUCTION TO BIOINFORMATICS

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Define and analyse the structure of biomolecules such as DNA, RNA and Protein.

CO2: Apply various data structures to represent biological data for efficiently solving biological problems.

CO3: Critically analyse different bioinformatics algorithms, and develop novel and efficient methods for biological data analysis.

Basic concepts and Sequence Analysis

Molecular Biology primer, gene structure and information content, Bioinformatics tools and databases, genomic information content, Sequence Alignment, Algorithms for global and local alignments, Scoring matrices, Dynamic Programming algorithms.

Phylogenetic Analysis

Introduction to Bioinformatics programming Languages, Motif finding, Gene Prediction, Molecular Phylogenetics, Phylogenetic trees, Algorithms for Phylogenetic Tree construction

Pattern Matching of Biological Sequences

Combinatorial Pattern Matching, Repeat finding, Keyword Trees, Suffix Trees, Heuristic similarity search algorithms, Approximate Pattern Matching.

Algorithms for Biological Data Analysis

Microarrays, Gene expression, Algorithms for Analysing Gene Expression Data, Data Mining in Bioinformatics, Protein and RNA structure prediction, Algorithms for Structure Prediction.

References:

1. N. C. Jones and P. A. Pevzner, *An Introduction to Bioinformatics Algorithms*, MIT Press, 2004.
2. D. E. Krane and M. L. Raymer, *Fundamental Concepts of Bioinformatics*, Pearson Education, 2003.
3. T. K. Attwood and D. J. Parry-Smith, *Introduction to Bioinformatics*, Pearson Education, 2003.

CS4052E NUMBER THEORY AND CRYPTOGRAPHY

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able :

CO1: To apply the number-theoretic foundations on modern cryptographic solutions to security

CO2: To build cryptosystem from conceptual understanding of the notions of security

CO3: To implement and analyse cryptographic and number-theoretic algorithms

Elementary Number Theory

Divisibility theory in integers: Extended Euclid's algorithm. Modular Arithmetic – exponentiation and inversion. Fermat's Little Theorem, Euler's Theorem. Solution to congruences, Chinese Remainder Theorem.

Abstract Algebra and Applications

Review of abstract algebra: Study of Ring Z_n, multiplicative group Zn* and finite field Z_p – Gauss Theorem (cyclicity of Z_p*) - Quadratic Reciprocity. Primality Testing – Fermat test, Carmichael numbers, Solovay Strassen Test, Miller Rabin Test - detailed analysis.

Secure Cryptosystems

Notions of security: Introduction to one secret key cryptosystem (DES) and one cryptographic hash scheme (SHA). Public Key Cryptosystems – Diffie Hellman Key Agreement Protocol, Knapsack crypto systems, RSA. Elgamal encryption and signature scheme. Key Management Protocols

References:

1. H. Delfs and H. Knebl, *Introduction to Cryptography: Principles and Applications,* Springer-Verlag, 2002.
2. S. Vaudenay, *A Classical Introduction to Cryptography: Applications for Communications Security*, Springer, 2009.
3. B. A. Forouzan and D. Mukhopadhyay, *Cryptography and Network Security,* 3rd ed. Tata McGraw Hill, 2015
4. Kenneth Rosen, *Elementary Number Theory: And Its Applications*, 6th ed. Pearson, 2010
5. Jonathan_Katz and Yehuda Lindell, *Introduction to Modern Cryptography,* 2nd ed. Chapman and Hall, 2015
6. Thomas Baigneres, Pascal Junod, Yi Lu, Jean Monnerat, and Serge Vaudenay, *A Classical Introduction to Cryptography Exercise Book*, 1st ed. Springer Piublication, 2005
7. Victor Shoup, *A Computational Introduction to Number Theory and Algebra,* 2nd ed., Cambridge Univ. Press, 2008.

CS4053E DATA MINING

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Apply appropriate data preprocessing techniques on a given dataset and use data visualisation tools to report the statistical similarity of the data
CO2: Apply, evaluate and compare the performance of various classification techniques for a given dataset
CO3: Demonstrate association analysis on a given transaction dataset and evaluate the usefulness of the association patterns identified
CO4: Apply, evaluate and compare the performance of various clustering techniques for a given dataset

Introduction to data mining - challenges and tasks -Data objects and attributes - Statistical description of data - Measures of data similarity and dissimilarity, Data visualization – concepts and techniques, Data preprocessing - data cleaning, data integration, data reduction, data transformation - Introduction to Data warehousing and OLAP.

Classification - Decision tree, Rule based classifier, Nearest-neighbour classifier, Bayesian classifiers, Support Vector Machines, - Model evaluation and selection - Techniques to improve classification accuracy - Class imbalance problem.

Mining frequent patterns and association analysis –Frequent itemset mining - Apriori algorithm, FP-growth algorithm - Evaluation of association patterns and correlation analysis.

Cluster analysis - Partitioning method - K means algorithm - Hierarchical clustering - Density based methods - DBSCAN - Cluster evaluation - Outlier detection - Application of data mining to appropriate problems.

References:

1. P. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, Pearson Education 2006.
2. J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2/e, Morgan Kaufmann, 2005.
3. T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning - Data Mining, Inference, and Prediction*, 2/e, Springer, California, 2008.

CS4054E EMBEDDED SYSTEMS

L	T	P	O	C
3	0	0	6	3

At the end of the course, the student will be able to:

CO1: Demonstrate the system components of an embedded computer.

CO2: Design and implement the tiny embedded system.

CO3: Demonstrate the latest tools for implementing embedded systems.

Definition of Embedded Systems, Embedded Systems vs. General Computing Systems, History of Embedded Systems, Classification, and Applications of Embedded Systems: Consumer, Industrial, Automotive, Home Appliances, Medical, Telecommunication, Commercial, Aerospace, and Military Applications

Characteristics and Quality Attributes of Embedded Systems, The core of the Embedded System: General Purpose and Domain Specific Processors, Microcontrollers, DSPs, FPGAs, ASICs, PLDs, Memory: ROM, RAM, and memory according to the type of interface. Onboard Computers for Embedded System Development: Raspberry Pi, Jetson Nano—hardware components; heterogeneous systems for embedded system development - GPUs and FPGAs

Software for Embedded Systems, Real-time operating System (RTOS), GPOS Vs RTOS, Components of an RTOS, Tasks - Processes and Threads, Process Scheduling, Process Synchronization - mutex and semaphore, priority inversion, commercially used RTOS - QNX, VxWorks, Shared Memory, Message Passing, Remote Procedure Calls and Sockets, Device Drivers, How to Choose an RTOS

The embedded software development tools, hosts, and target machines, Recent trends in embedded systems Embedded systems for Robotics & IoT, Embedded systems for various real-time applications, Integration with cloud services to build intelligent connected systems, Edge-computing

References:

1. S. Heath, *Embedded System Design*, 2/e, Elsevier, 2002.
2. D. E. Simon, *An Embedded Software Primer*, Pearson Education, 2010
3. DreamTech Software Team, *Programming of Embedded Systems*, Wiley DreamTech, 2002.
4. E. A. Lee and S. A. Seshia, *Introduction to Embedded Systems - A Cyber-Physical Systems Approach*, Second Edition, MIT Press, 2017.

CS4055E OBJECT ORIENTED SYSTEMS

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate the key concepts of Object Orientation

CO2: Implement efficient systems using appropriate design patterns and techniques

CO3: Evaluate the system developed

Foundations of Object-Oriented Systems

An Overview of Object Oriented Systems Development - Understanding Objects and Classes, Object Oriented System Development Life Cycle - Key Methodologies in OOS Development - Introduction to Unified Modelling Language (UML)- UML Diagram Types and Their Usage, Creating UML Class Diagrams, Association, Aggregation, and Composition in UML

Analysis & Design in Object Oriented Systems

Use Case based Object Oriented Analysis - Creating and Interpreting Use Case Diagrams - Classification in Object-Oriented Analysis, Identification of Object Relationships and Methods - Object Oriented Design - Principles of Object-Oriented Design - Understanding Design Patterns, Introduction to Creational and Structural Design Patterns - Designing Data Store: Principles and Best Practices - Designing User Interface in an OO Context

Architecture, Testing, and Metrics

Object Oriented Architecture and Testing - Key Principles and Components - Strategies for Effective OOAD Testing - Introduction to Quality Strategies in OOAD- Object Oriented Metrics - Key Metrics in OO Systems and Their Interpretation - Best Practices in OOA and OOAD Testing

Implementation and Management in OOAD

Introduction to OOAD Implementation Strategies - OOAD Pragmatics - Techniques and Best Practices for Release Management - Handling Changes in OOAD: Strategies and Tools- The Importance of Reuse in OOAD- Strategies for Effective Reuse in OOAD

References:

1. A. Bahrami, *Object Oriented Systems Development,* 1/e, McGraw-Hill, 2000
2. G. Booch, R. A. Maksimchuk, M. W. Engle, B. J. Young, and J. Conallen, *Object-Oriented Analysis and Design with Applications,* 3/e, The Addison Wesley Object Technology Series.
3. B. D. McLaughlin, G. Pollice, and D. West, *Head First Object-Oriented Analysis and Design*, 1/e, O'Reilly .
4. M. Weisfeld, *Object-Oriented Thought Process*, 4/e, Addison-Wesley Professional, 2013.

CS4056E APPROXIMATION ALGORITHMS

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Identify the design technique used in a given approximation algorithm.

CO2: Design and analyze approximation algorithms for NP-hard problems using various combinatorial techniques, randomization and LP methods.

CO3: Apply reduction techniques of proving hardness of approximation of simple combinatorial problems.

NP-optimization, approximation ratios and local search

Review of NP Hardness and reductions, NP-optimization problems, approximation ratio; multiplicative and additive, technique of lower bounding optimum, simple approximation algorithms for vertex cover and set cover. Local search: max-cut, minimum spanning tree, metric steiner tree and metric TSP, approximation preserving reduction for steiner tree.

Approximation schemes and dynamic programming

Layering: minimum vertex cover, feedback vertex set, Strong NP Hardness and pseudo-polynomial-time algorithms, Knapsack. Dynamic programming: FPTAS for the Knapsack, asymptotic PTAS for Bin Packing.

LP based techniques

Linear programming: simple rounding - MAX-SAT with small clauses, Set Cover, Bin Packing, primal-dual method - set Cover, steiner forest. Introduction to Semidefinite Programming: SDP approximation algorithm for Max Cut.

Hardness of approximation

Hardness of Approximation: Reductions from NP-complete problems, TSP, approximation preserving reductions, PCP Theorem, gap reductions, minimum vertex cover, maximum Clique.

References:

1. David P. Williamson and David B. Shmoys, *The design of Approximation Algorithms*, Cambridge University Press, 2011.
2. Vijay V. Vazirani, *Approximation Algorithms*, First Edition, Springer-Verlag Berlin Heidelberg, 2003.
3. G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti Spaccamela, M. Protasi, *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*, Springer, Second corrected printing 2003.
4. Sariel Har-Peled, *Geometric Approximation Algorithms, AMS Series in Mathematical Surveys and Monographs*, 2011
5. T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, *Introduction to Algorithms,* 3rd ed. MIT Press, 2009.

CS4057E DATA PRIVACY

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Appraise the need for privacy preservation techniques in the digital domain and differentiate between data privacy related concepts like anonymisation and pseudonymisation.

CO2: Define differential privacy and understand the fundamental techniques for achieving the same.

CO3: Devise privacy preserving approaches for practical applications using relevant tools.

Data Privacy - Definition, Requirements, Solution approaches. Ethical and Legal aspects.

Differential privacy - The formal approach for data protection, Private data analysis, Basic techniques, Composition theorems, Extensions and generalisations.

Building privacy into data pipelines - Tools for differentially private analysis.

References:

1. C. Dwark and A. Roth, *The Algorithmic Foundations of Differential Privacy,* Foundations and Trends in Theoretical Computer Science Vol. 9, Nos. 3–4, 2014.
2. J.P. Near and C. Abuah, *Programming Differential Privacy*, Available online at https://uvm-plaid.github.io/programming-dp/ (accessed October 10, 2023).
3. K. Jarmul, *Practical Data Privacy,* O'Reilly Media, Inc., 2023.
4. Nissim et al., *Differential Privacy: A Primer for a Non-technical Audience,* Available online at https://privacytools.seas.harvard.edu/files/privacytools/files/pedagogical-document-dp_new.pdf (accessed October 10, 2023).
5. OPENDP - Open Source Software Tools for Differential Privacy, Available online at https://docs.opendp.org/en/stable/index.html (accessed October 10, 2023).

CS4058E CODING THEORY

L	T	P	O	C
3	0	0	6	3

Total Lecture Sessions: 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Demonstrate various codes and their encoding decoding procedures.

CO2: Design a linear code for given specification.

CO3: Solve error bounds on various code families.

Linear Codes: Review of linear algebra - Linear codes and syndrome decoding. Generator and parity check matrices, Singleton bound, Plotkin bound. Hamming code, Hadamard code.

Cyclic codes: BCH codes, BCH–key equation and algorithms. Berlekamp's Iterative decoding Algorithm, decoding with Euclid’s algorithm, Reed Solomon codes, decoding Reed Solomon codes.

List decoding: Johnson’s bound, Sudan-Guruswami algorithm. Convolutional codes: trellis decoding, Viterbi’s algorithm.

Codes on Graphs: Concept of girth and minimum distance in graph theoretic codes. Expander Graphs and Codes – linear time decoding. Basic expander based construction of list decodable codes. Sipser Spielman algorithm. Bounding results.

References:

1. W. C. Huffman and V. Pless, *Fundamentals of error correcting codes*, Cambridge University Press, 2003.
2. J. H. Van Lint, *An Introduction to Coding Theory,* 2/e, New York: Springer-Verlag, 1992.
3. R. J. McEliece, *The Theory of Information and Coding*, Addison Wesley, 1997

CS4059E TERM PAPER

L	T	P	O	C
3	0	0	6	3

Total Sessions:Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Examine the relevant literature in a field of study.

CO2: Analyse critically methods, tools and techniques in the field of study.

CO3: Prepare a technical report based on the investigation.

The student is required to conduct a detailed literature survey in an area of research in the field of computing and compile a report of her/his study in the area. The student may include any observations from her/his own experimentation in the report.

References:

1. G. J. Alred, C. T. Brusaw, and W. E. Oliu, *The Handbook of Technical Writing,* 11/e, Bedford/St. Martins, 2015.
2. P. A. Laplante, *Technical Writing: A practical guide for engineers and scientists*, CRC Press, 2011.

CS4080E OPERATING SYSTEMS LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Implement a simple multi-tasking operating system kernel prototype in laboratory settings.

CO2: Implement a simple file system prototype to be attached to the OS constructed.

CO3: Test the functionality using application programs exploiting various OS features.

Theory

System programming fundamentals and system call interface. System calls for file system, process management and process synchronisation.

Practical

1. Loading executable programs into memory, implement file system calls Read(), Write(), Open (), Seek() and Close().
2. Multiprogramming-Memory management- Implementation of Fork(), Wait(), Exec() and Exit() System calls.
3. IPC system call implementation - Signal-Wait, Semaphores.
4. Demand paging and swapping -implementation.
5. Shell and System utilities - implementation.
6. Implementation Low level routines - disk and terminal driver, timer interrupt handler and scheduler.

References:

1. G. J. Nutt, *Operating Systems,* 3/e, Pearson Education, 2004.
2. D. P. Bovet and M. Cesati , *Understanding the Linux Kernel,* 3/e, O'Reilly Media, 2005
3. C. Crowley, *Operating Systems: A design oriented approach*, 1/e, Mc. Graw Hill, 2006.

CS4081E COMPILER LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

The student will implement the following for a programming language in an experimental setting:

CO1: Lexical and syntax analyzer using tools Flex and Bison.

CO2: Type checking, scope resolution and storage allocation for variables, arrays, structures and classes.

CO3: Code generation for a pre-specified target machine.

Theory

Introduction to the use of tools Flex and Bison. Syntax directed translation scheme in Bison. Code generation from abstract syntax tree, run time activation records, dynamic memory allocation.

Practical

Generation of lexical analyzer using Flex.
Parsing using Bison.
Symbol table implementation.
Type checking using syntax directed translation scheme of Bison.
Intermediate code generation - abstract syntax tree representation of programs.
Run time stack implementation for subroutine invocations.
Register allocation and code generation.

References: .

1. W. Appel, *Modern Compiler Implementation in C*, Cambridge University Press, 1998.
2. V. Aho, M. S. Lam, R. Sethi, and J. D. Ullman, *Compilers- Principles, Techniques & Tools*, 2/e, Pearson Education, 2007.
3. D. Grune, K. V. Reeuwijk, H. E. Bal, C. J. H. Jacobs, and K. Langendoen, *Modern Compiler Design*, 2/e, Pearson Education, 2007

CS4082E DIGITAL DESIGN LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Analyse the architectural design of the MIPS processor and its Instruction set.

CO2: Analyse a RISC processor's performance using standard performance evaluation metrics.

CO3: Design and implement simple single and multi cycle MIPS processors.

Theory

MIPS Instructions

Practical

Design and implementation of the following using MIPS

1. Arithmetic Logic Unit
2. Register file
3. Control Unit
4. Single cycle processor implementation
5. Pipeline processor

References:

1. B. J. LaMeres, Introduction to Logic Circuits & Logic Design with Verilog, 1/e, Springer, 2017.
2. S. Brown and Z. Vranesic, Fundamentals of Digital Logic with Verilog Design, McGraw-Hill Higher Education, Har/Cdr edition, 2002.
3. M. M. Mano and M. D. Ciletti, Digital Design, 4/e, Pearson Education, 2008
4. T. L. Floyd and R. P. Jain, Digital Fundamentals, 8/e, Pearson Education, 2006.

CS4083E DATABASE SYSTEM DESIGN LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

CO1: Develop code to implement the components of a simple single user RDBMS system.

CO2: Implement various algorithms for data access and retrieval in an experimental RDBMS

CO3: Store and retrieve data from an RDBMS using a query language.

Theory

Software architecture of a simple single user RDBMS. Review of run time data structures for RDBMS implementation and Low level file organisation, Run time catalogue, catalogue caching and disk buffering.

Practical

1. Query exercises for understanding disk data organisation and retrieval of stored data on an available RDBMS.
2. Implementation of memory buffering of disk blocks and meta data of open relations.
3. Implementation of catalogue operations - creation, deletion of relations.
4. Implementation of indexing algorithms and data structures (B/B+trees)
5. Implementation of algorithms for search, insertion and update of records.
6. Implementation of algorithms for select, project and join operations on relations.

References: .

1. H. Garcia-Molina, J. D. Ullman, J. Widom, *Database Systems: The Complete Book,* Pearson Prentice Hall, 2009.
2. H. Garcia-Molina, J. D. Ullman, J. D. Widom, Database System Implementation Pearson 1999

CS4084E NETWORKS LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Practice network debugging and performance analysis utilities

CO2: Implement client and server programs using socket programming

CO3: Implement simulations of various network scenarios using NS/NS3

Theory

Introduction, Overview of Unix Programming Environment, Unix Programing Tools, Introduction to Computer Networking and TCP/IP,

Introduction to Socket Programming, TCP Sockets UDP sockets and the variants.

Introduction to raw packet generation

Introduction to Berkeley Packet Filters

Introduction to network emulators and simulators - P4 Programming, OpenFlow, OpenvSwitch

Sample Experiments

1. Implementation of basic Client Server program using TCP and UDP sockets.
2. Implementation of TCP Client Server program with concurrent connection from clients.
3. Implementing fully concurrent application with a TCP server acting as a directory server and client programs allowing concurrent connection and message transfer (Eg. Chat system).
4. Fully decentralised application like a Peer to Peer system. This program is to implement without designated Server as in the case of experiment 4.
5. Experiments with open source Berkely packet filters/eXpress Data Path (XDP).
6. Experiments with generation of raw network packets.
7. Experiments with Emulators such as Mininet/NS3/OpenFlow/P4.

References: .

1. W. R. Stevens, *Unix Network Programming – Networking APIs: Sockets and XTI,* 2/e, Volume 1, Pearson Education, 2004.
2. W. W. Gay, *Linux Socket Programming by Example*, 1/e, Que Press, 2000
3. Brian “Beej Jorgensen” Hall, *Beej's Guide to Network Programming Using Internet Sockets,* https://beej.us/guide/bgnet/html/ (Last accessed on 09.10.2023).
4. Renzo Davoli, Berkeley Packet Filter: *Theory, Practice, and Perspectives*, 1st ed. https://amslaurea.unibo.it/19622/1/berkeleypacketfilter_distefano.pdf (Last accessed on 10.10.2023)

CS4085E SOFTWARE ENGINEERING LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Develop a software application using relevant tools.

CO2: Create necessary documentations relating to software development.

CO3: Coordinate effectively with a team for software development.

Theory

Introductory Lectures on the use of appropriate tools is to be given. Peer review discussions of deliverables, and the demonstration of prototypes will be done in theory sessions.

Practical

Objective is to develop a significant software product using sound software engineering principles by small student groups. Choice of appropriate methodology and standard tools are also expected. The lab will have deliverables at each milestone of development.

1. Problem Statement / Product Specification
2. Project Plan – Project Management Tool to be identified and necessary estimations done.
3. Requirements Document – Specification to be justified - In class Review
4. Design Document – Choice of Methodology to be justified - In class Review
5. Code and Test Report – Peer review documents of standards adherence to be provided
6. Demo – Integrated Product or Solution to the problem
7. Review of the process, and analysis of variation from initial plan and estimation.

These steps may be replaced by Agile Methodologies and associated tool sets like Trello, Git, Docker, CircleCi etc. depending on the kind of project and availability of committed customers.

References: .

1. R. S. Pressman and Bruce Maxim, Software Engineering: A Practitioner's Approach, 9/e, McGraw Hill, 2020.
2. T. C. Lethbridge and R. Laganiere, *Object Oriented Software Engineering*, 1/e, Tata McGraw Hill, 2004.
3. Jake Knapp, John Zeratsky and Braden Kowitz, *SPRINT: How to Solve Big Problems and Test New Ideas in Just Five Days*, Bantam Press, 2016.

CS4086E SYSTEMS PROGRAMMING LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Code simple system task scheduling and batch execution tasks using shell scripts.

CO2: Solve synchronisation problems using built in OS system calls and using software solutions.

CO3: Implement a simple dynamic memory allocator.

Theory

Basics of shell programming. Unix system call interface, semaphores, shared memory, Posix threads, thread and process synchronisation.

Practical

1. Unix/Linux shell programming exercises.
2. Inter-process communication and process synchronisation, semaphores and shared memory - solution to dining philosophers problem, reader-writer problem.
3. Signal handling, wait and kill operations.
4. Threads for multiprogramming - posix threads, reader- writer problem using threads.
5. Software synchronisation - Peterson's algorithm.
6. Dynamic memory allocation - variable cell allocation, Buddy system allocation.

References:

1. B. W. Kernighan and R. Pike, *The Unix Programming Environment,* Prentice Hall, 1983.
2. W. R. Stevens, *Advanced Programming in the Unix Environment,* 3/e, Addison Wesley, 2013**.**

CS4087E COMPUTER SECURITY LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: analyse the common vulnerabilities in Operating Systems, Web Servers and Application

CO2: apply various exploiting techniques in Operating Systems, Web Servers and Application

CO3: design and implement new exploiting techniques in applications

Theory

Review of Computer Networks - Associated vulnerabilities and prevention mechanisms. Review of Cryptographic Algorithms - Secret and Public Key Cryptography, Key Management, Hashing, Signature Review of Software Vulnerabilities - Buffer Overflow, Format String, Race Conditions Review of Internet Vulnerabilities - DoS, SQL injection, XSS. Review of Mobile apps and its vulnerabilities.

Practical

Simulating attacks based on vulnerabilities in Network, Web and Software (includes attacks like buffer overflow, format string, SQL injection, XSS)

Implementation of cryptographic primitives and solutions using cryptographic libraries in Python / Java Deploying and using tools like Wireshark, Webgoat, Nmap, Metasploit, Ettercap.

Static analysis: Reverse engineering executables and mobile applications using tools like IDA Pro and OllyDbg. Dynamic analysis.

References: .

1. Michael Gregg, *Build Your Own Security Lab*, Wiley India, 2008
2. B. Menezes, *Network security and Cryptography,* Cengage Learning India, 2010.

CS4088E OBJECT ORIENTED SYSTEMS LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: differentiate procedural and object oriented paradigms

CO2: design using object oriented principles using UML

CO3: implement object oriented projects in Java

Theory

Procedural vs. Object oriented approaches – Concept of Abstraction - Design and analysis using OO methodologies - Introduction to UML.

Practical

The implementation has to be done Java

1. Functions – Control structures – String handling – File handling
2. Error and Exception handling
3. Class – Object Instantiation
4. Principles of Inheritance, Encapsulation,
5. Polymorphism – Abstract classes, Interface, overloading, overriding, virtual functions, .
6. Design patterns

References:

1. B. Stroustrup, *The C++ Programming Language,* 3/e, Addison Wesley, 1997.
2. S. Oualline, *Practical C++ Programming,* 2/e, O'Reilly & Associates, 2002.
3. J. Nino and F. A. Hosch, *An introduction to programming and object oriented design using Java,* Wiley India, 2010

CS4089E MACHINE LEARNING LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Implement and apply supervised and unsupervised classification methods.

CO2: Implement parameter estimation techniques and linear data analysis methods.

CO3: Design, build and analyse ANN and CNN architectures for classification.

Theory

Fundamentals of Machine Learning, Bayes classifier, Parameter Estimation, Discriminant Functions, SVM, ANN, Clustering.

Practical

1. Bayes Classifier.
2. Parameter Estimation(MLE and Non-parametric Estimation).
3. Linear Data Analysis Methods(LDA,PCA,Regression).
4. Support Vector Machines.
5. KNN Classifier.
6. K-Means Clustering.
6. Decision Trees and Random Forest.
7. Artificial Neural Network and Deep Learning.

References:

1. R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification* , John-Wiley, 2004.
2. C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.
3. Yuxi(Hayden) Liu, *Python Machine Learning by Example*, Packt, 2017.

CS4090E IMAGE PROCESSING LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Implement the basic image processing algorithms

CO2: Analyse images in the frequency domain using various transforms

CO3: Implement and evaluate the techniques for image enhancement and restoration

Theory

An introduction to digital images- sampling, quantization. Basic image processing, arithmetic processing. Image enhancement and point operation. Image enhancement and spatial operation. Colour images and models. Frequency domain operations. Image Analysis.

Practical

1. Image resampling
2. Basic image processing, arithmetic processing
3. Image enhancement and point operation- Linear point operation, clipping, thresholding, negation, non-linear mapping, intensity slicing, image histogram, histogram equalisation.
4. Image enhancement and spatial operation- Convolution, correlation, linear filtering, edge detection. 5. Colour image processing - colour models, colour enhancement, colour thresholding.
6. Frequency domain operations- fourier transform, freq domain filtering
7. Image analysis after basic image processing algorithms.

References:

1. R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison Wesley, 2007.
2. A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, Englewood Cliffs, 2002.

CS4091E PROCESSOR DESIGN LABORATORY

L	T	P	O	C
1	0	3	5	3

Total Sessions: Lecture 13 and Practical 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Design and implement different primitive operations using HDLs.

CO2: Design and implement Single cycle, multi cycle and pipelined processor using HDLs.

CO3: Design and implement superscalar processor and analysing the performance.

Theory

HDLs, Pipeline architecture, Supersacalar architecture

Practical

Writing HDL programs for

1. ALU operations and Register file.
2. Design and Implementation of a Single-cycle processor.
3. Design and Implementation of a Multi-cycle processor.
4. Design and Implementation of a RISC-V Pipeline processor.
5. Design and Implementation of Superscalar processor

References:

1. B. J. LaMeres, *Introduction to Logic Circuits & Logic Design with Verilog*, 1/e, Springer, 2017.
2. S. Brown and Z. Vranesic, *Fundamentals of Digital Logic with Verilog Design*, McGraw-Hill Higher Education, Har/Cdr edition, 2002.
3. M. M. Mano and M. D. Ciletti, *Digital Design*, 4/e, Pearson Education, 2008
4. T. L. Floyd and R. P. Jain, *Digital Fundamentals,* 8/e, Pearson Education, 2006.
5. Samir Palnitkar, *Verilog HDL: A Guide to Digital Design and Synthesis*,Prentice Hall PTR, 2003.

CS4098E PROJECT

L	T	P	O	C
0	0	0	9	3

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Compile findings in a topic of interest relevant to the domain.
CO2: Choose a problem from the topic to solve and justify the choice.
CO3: Design the solution to the problem and document the same.
CO4: Attempt to solve the problem using relevant tools / techniques.
CO5: Summarise the findings / solution as a technical report.
CO6: Present the findings / solution and defend the same.

The project may be carried out in industry or academia, individually or in a group, and could be interdisciplinary.

CS4099E PROJECT

L	T	P	O	C
0	0	0	18	6

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Compile findings in a topic of interest relevant to the domain.
CO2: Choose a problem from the topic to solve and justify the choice.
CO3: Design the solution to the problem and document the same.
CO4: Attempt to solve the problem using relevant tools / techniques.
CO5: Summarise the findings / solution as a technical report.
CO6: Present the findings / solution and defend the same.

The project may be carried out in industry or academia, individually or in a group, and could be interdisciplinary.

CS4101E DEEP LEARNING: ALGORITHMS AND ARCHITECTURES

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:
CO1: Describe and analyze basics of neural networks and apply the learning rules for deep neural networks.
CO2: Design, implement and apply Deep Neural Networks using CNN and RNN for object detection, image segmentation and text related problems
CO3: Describe mathematical, statistical and computational challenges of building stable representations for high-dimensional data, such as images and text using case studies.

Neural Networks
Introduction to Human and Artificial Intelligence, History of AI, Forms of learning - Supervised and unsupervised learning, Perceptron Learning rule, Bio-inspired learning, Artificial Neural Networks, Backpropagation, Multi-layer Perceptron model, Activation Functions Loss functions, Optimization, Training Neural Networks - gradient descent, stochastic gradient descent, momentum, weight initialization, batch normalization, hyper parameter optimization, parameter updates, model ensembles.

Image Classification using CNNs
Convolutional Neural Networks: convolution layer, pooling layer, fully connected layer, Conv Net, Case study of ImageNet challenge: LeNet, AlexNet, VGG, GoogLeNet, ResNet, Inception Net, Efficient Net etc. Regularization Techniques, Data Augmentation: zooming, rotation, cropping, blurring, noise addition, self-supervision techniques, semi-supervised and weakly supervised learning, adversarial training Transfer Learning, freezing the input layers, fine tuning output layers.

Deep learning for Segmentation and Object detection
Image Localization, Image segmentation, masks, Image segmentation architectures: Unet, VNet, UNet++, Object Detection – Region Proposal Networks, Objection architectures RCNN, Fast and Faster RCNNs, Mask RCNN, YOLO, BiFPN layers, Centre Net, EfficientDet, Case study: RoI cropping in CT images

RNNs and Transformers
Sequential models, Recurrent Neural Networks, Long Short-Term Memory, Gated Recurrent Units, Backpropagation Through Time (BPTT), Transformer Networks, Introduction to Attention Mechanism - Queries, Keys, and Values, Multi-Head Attention, Self-supervision techniques, Identifying missing words in a paragraph, text summarization.
Hardware and Software requirements for Deep Learning - FPGA, ASIC, GPUs, GPU architectures – Pascal, Volta, Turing & Ampere, Data Parallelism in GPU, Kernels, TPUs, Frameworks for Deep Learning - PyTorch, TensorFlow.

References:

1. Y. Bengio, I. Goodfellow and A. Courville, *Deep Learning*, MIT Press, 2016.
2. Nielsen, Michael A. *Neural networks and deep learning.* Vol. 25. San Francisco, CA, USA: Determination press, 2015.
3. Bishop, C., M., *Pattern Recognition and Machine Learning*, Springer , 2006.
4. Francois Chollet, *Deep Learning with Python*, Manning Publications, 2017.
5. Prince, Simon JD. *Computer vision: models, learning, and inference*. Cambridge University Press, 2012.
6. Stuart Jonathan Russell, Peter Norvig, *Artificial Intelligence: A Modern Approach*, 4th Edn, Pearson Education, Inc, 2016.

CS4102E AI FOR HEALTHCARE

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Identify problems that healthcare providers are facing which deep learning can solve, to bring AI technologies into the clinic, safely and ethically.

CO2: Apply the building blocks of AI to understand emerging technologies and to determine the most effective treatments.
CO3: Solve the current and future applications of healthcare using AI to predict and improve patient and financial outcomes.

Medical data acquisitions, Data representation and formats: Human visual system and visual perception; Image sensing and acquisition Image file types; Pixel representation and spatial relationship. Signal processing overview; Image processing basics; Fundamental signals (1-D and 2-D); Introduction to medical imaging modalities, Capturing images of the parts of the human body, Advanced imaging (MRI &CT), Medical image representation formats – Dicom & NifTI, recent advances in analysis of retinal, CT, MRI, ultrasound and histology images
AI for Cancer Diagnosis and Staging: Cancer detection and diagnosis from images, AI for early detection of cancers in image guided examinations, Classical CNN architectures for image classification. Cervical cancer and its early detection, Cervical cancer detection methods: Pap smear, VIA & HPV test. Cervigrams, Region of Interest, object detection identifying the cervix area in medical images, use of self- supervision in biomedical imaging applications, Development of AI based biomedical devices for cancer detection, Urine Cytology images for Bladder Cancer detection using AI, AI for ER/PR/Her2 biomarker identification from H&E stained microscopic images.
Deep Segmentation Applications in Treatment and Planning: AI for treatment planning in Radiotherapy - Radiotherapy treatment workflow for cancer treatment, Imaging modality used in radiotherapy, Computerised Tomography, Fan Beam CT, Cone Beam CT, Treatment Planning System (TPS) in radiation therapy, Linear accelerators, Contouring for GTV and OARs, different image segmentation techniques - Atlas based segmentation, Deep Learning based segmentation, Unet, segmenting organs in CT using UNet, GANs, Synthesizing FBCT images from MR and CBCT images.
AI in Dentistry, Cardiology and Psychiatry: AI in *Dentistry:*Imaging modalities used in dentistry – OPG(OrthoPantomogram), CBCT, Segmenting teeth from a X Ray/CT image, Precise dental implant positioning and fixation using AI. AI in *Cardiology*: processing & analysing an echocardiogram for faster identification of heart diseases using AI, Deep learning-based ultrasound image processing. AI for *Psychiatry*: emotion detection, text simplification and summarization, Transformers for sequence modelling, Chatbots applications for psychiatric rehabilitation.

References:

1. Y. Bengio, I. Goodfellow and A. Courville, *Deep Learning*, MIT Press, 2016.
2. Francois Chollet, *Deep Learning with Python*, Manning Publications, 2017.
3. Bishop, C., M., *Pattern Recognition and Machine Learning*, Springer , 2006.
4. S. J. Russell, P. Norvig, *Artificial Intelligence: A modern approach*, 4th ed.,Pearson, 2016.
5. D. Foster, *Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play*, O'Reilly Media, 2019.

CS4201E CYBER-PHYSICAL SYSTEMS: MODELLING AND SIMULATION

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:
At the end of the course, the student will be able to:

CO1: Describe the evolution and current status of cyber physical systems in the real world.

CO2: Design and Model simple cyber physical systems and specify their safety and liveness requirements.

CO3: Develop prototype for a simple cyber physical system using software tools.

Introduction to Cyber-Physical Systems

Emergence of CPS - Key Features, Theoretical Foundations, Real world Application Domains, Security primitives and considerations.

Design and Modeling of Cyber-Physical Systems

Synchronous Models - Reactive Components, Finite State Machines - Deterministic and Nondeterministic Models. Asynchronous Models - Extended State Machines, Asynchronous Design Primitives - Blocking vs Non-blocking synchronizations, Deadlocks, Shared Memory. Safety and Liveness Requirements. Real-Time Scheduling.

Simulation of Cyber-Physical Systems

Prototype development using simulation tools. Discrete and Continuous simulation. Data Exchange and Time-based Coordination. Use of visualization techniques for cyber physical systems.

References:
1. R. Alur, *Principles of Cyber Physical Systems*, MIT Press, 2015.
2. R. Rajkumar, D. de Niz, M. Klein, *Cyber-Physical Systems*, Pearson Education, 2017.
3. W.M. Taha, A.M. Taha, J. Thunberg, *Cyber-Physical Systems: A Model-Based Approach*, Springer-Cham, 2020.

CS4202E FOUNDATIONS OF IoT AND M2M COMMUNICATIONS

L	T	P	O	C
3	0	0	6	3

Total Sessions: Lecture 39

Course Outcomes:

At the end of the course, the student will be able to:

CO1: Describe the functionality and services offered at various layers of TCP/IP protocol stack.
CO2: Analyse, design and implement protocols in each layer of the TCP/IP stack.
CO3: Configure and Implement M2M and IoT Architecture.

Computer Networks and Internet, the network edge, Core and access network, protocol layers and services, Application layer protocols, Transport layer services, UDP, TCP, Network layer services, Routing, IPv6 Protocol, Transition from IPv4 to IPv6, Network Design.

M2M to IoT – The Vision - Use case, M2M value chains, IoT value chains, Industrial structure for IoT.
M2M to IoT – An Architectural Overview, Building an architecture, Main design principles.
Security considerations.

M2M and IoT Technology Fundamentals, Business processes in IoT, Everything as a service (XaaS).
M2M and IoT Analytics, IoT Architecture – State of the Art, Architecture Reference Model, IoT Reference Architecture, IoT Use Cases with emphasis on security - Industrial Automation, Smart Grid, Smart Cities.

References:

1. Kurose J. F. and K. W. Ross, *Computer Networking: A Top-Down Approach Featuring Internet*, Pearson Education, 2017.
2. Höller, J., Vlasios T., Catherine M., Stamatis Karnouskos, Stefan A, David B., *From Machine-to-Machine to the Internet of Things - Introduction to a New Age of Intelligence*, Elsevier Ltd., 2014.
3. Peterson L.L. and Davie B.S., *Computer Networks, A systems approach*, Harcourt Asia, 2021.
4. Adrian Farrel, *The Internet and its Protocols: A Comparative Approach*, Elsevier, 2005.