INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

Academic Requirements and Regulations for Master of Science by Research Programmes

This is a research-oriented programme. The admitted students are expected to have an advanced CS/ECE background. There is no prescribed course work for this programme. Students should use the first year to make up for the deficiencies in their back ground, if any, and to gain advanced knowledge into the stream area of their choice.

Breadth/Depth Requirements:

A student should demonstrate the knowledge of the basic courses in at least 4 of the stream areas and must take an advanced course in a minimum of 3 stream areas.

Credit Requirements:

A student should take 48 credits at the institute to be eligible for the MASTER OF SCIENCE degree of which at least24 credits should be through course work and at least 24 of which should be Master of Science thesis credits. Credits of courses at level 4000 and above only count towards the degree. Students must register for at least 16 credits for the first three semesters and for at least 4 credits in fourth semester.

Semester	Course Type	Credits
1	Breadth / Depth Elective	4
	Breadth / Depth Elective	4
	Breadth / Depth Elective	4
	Total	12
2	Breadth / Depth Elective	4
	Breadth / Depth Elective	4
	Breadth / Depth Elective	4
	Total	12
3	Thesis Credits	12
4	Thesis Credits	12
Total Credits		48

Note: Course selection in each semester is based on the advisor's recommendations.

Master of Science Thesis:

Once the student completes his/her Thesis work, he/she has to submit the Graduation RequestForm(GRF)intheacademicofficeasperthe guidelinesgivenonintranethttp://intranet.iiit.ac.in/Information/academic/thesisEvaluation.html.

The Master Science thesis has to be defended to a committee of typically consists of 3 faculty members including the advisor, a faculty member nominated by PG Chair and a nominee of the Dean(R&D).

Academic Performance:

A student should complete the requirements with a minimum CGPA of 7.0 to receive the Master of Science degree.

Residency Requirements:

Full-time students: Minimum of 3 semesters and maximum of 6 semesters.
Part-time students: Minimum of 3 semesters and maximum of 8semesters.

Fees:

Full-time post-graduate fees during the coursework

Course descriptions for elective courses

Title of the Course: **Modern Complexity Theory**
Course Code: **CS1.405**
Faculty Name: **Srinathan Kannan**
L-T-P: 3-1-0
Credits: 4

Prerequisite Course / Knowledge:

Should have taken Introduction to Algorithms, and Formal Languages, or equivalent courses

Course Outcomes (COs):

After completion of this course successfully, the students will be able to..

CO-1: Understand different models of computation including Turing Machines, Boolean Circuits and complexity measures of time, space, depth.

CO-2: Demonstrate familiarity with various complexity classes including P, NP, PSPACE, NC and problems like Halting Problem, 3SAT.

CO-3: Design reductions between problems to show hardness of solving a problem in a complexity class.

CO-4: Synthesize proofs of upper and lower bounds of resources required for solving a computational problem using clear mathematical and logical arguments.

CO-5: Apply principles of NP-Completeness and NP-Hardness to avoid intractability in design of computational problems.

CO-6: Create mathematical models and complexity measures for novel computational models.

Detailed Syllabus:

Unit 1: Models of Computation and Impossibility Results: Turing Machines, Circuits, Encoding of Problems, Halting Problem, Shannon's Counting Lower bound.

Unit 2: Complexity Measures and Classes: Time, Space, Depth measures of complexity, Time, Space hierarchy theorems, Nondeterminism, Savitch's theorem, P, NP, P/poly, PSPACE, EXP, L, NL. **Unit 3:** Completeness and Hardness Reductions: 3SAT, Cook-Levin Theorem, NP-Complete, NL- Complete, Hardness reductions for common problems like VertexCover, Independent Set, Knapsack etc.

Unit 4: Advanced Topics: Definitions and relationships between PH, RP, BPP, NC including theorems like Karp-Lipton, Adleman's theorem, Derandomization Techniques.

Reference Books:

- S. Arora and B. Barak (2000), Computational Complexity: A Modern Approach, Cambridge University Press.
- C. Moore & S. Mertens (2011), The Nature of Computation, Oxford University Press.
- M. Sipser (2014), Introduction to Theory of Computation, Cengange Learning.
- C. Papadimitriou (1994), Computational Complexity, Addison Wesley Longman.

Teaching-Learning Strategies in brief (4 to 5 sentences):

Lectures will initially introduce the motivations, concepts, definitions along with simpler examples. This will be followed by assignments and quizzes that will make sure that the students have understood the concepts. These will be followed by deeper lectures and assignments which lead the students to the bigger questions in the area. The students will be given an advanced topic and will be required to summarize it in a presentation or a term paper. This will encorage self-exploration and lead the student to do research on fundamental questions.

Assessment methods and weightages in brief (4 to 5 sentences):

- Light In-class Quizes: 20%
- Assignments: 20%
- Deep Quiz 1: 10%
- Deep Quiz 2: 10%
- Mid and End Exam: 30%
- Student Presentation and Scribe notes: 10%

Title of the Course: Principles of Programming Languages (PoPL)
Course code: CS1.402
Faculty Name: Venkatesh Choppella

1 Course structure

Name Principles of Programming Languages
Credits 4, Lectures-Tutorials-Practicals=3-1-0 (hours/week)

2 Prerequisite courses

1. Computer Programming
2. Discrete Mathematics (with some exposure to writing proofs)
3. Automata Theory

3 Course outcomes

A student graduating from a PoPL course should be able to perform each of the following sample tasks:

1. **CO1: Document Abstract Syntax** Document and critique the ab- stract syntax of industrial scale programming language like C or Java.
2. **CO2: Design domain specific languages** Design a small, domain pur- pose languages like a language for propositional logic and implement them.
3. **CO3 Design object small oriented language** Design a small object oriented language implement it either using an interpreter or by em- bedding it into a base language.
4. **CO4: Compare languages** Compare and analyse the semantic express- ibility (in terms of first class values) between imperative languages like C and functional languages Racket, and object oriented languages like Java and Python.
5. **CO5: Specify application interfaces** Specify the structure of a soft- ware application like a spreadsheet or a word processor in terms of its interface as a language of user operations and its internal structure as an abstract machine.

4 Syllabus

Functional Programming: Abstract vs. Concrete Syntax. Racket syn- tax, Functions, recursion, syntactic extensions, higher-order functions, map, reduce and other combinators.

Operational Semantics: Abstract Reduction Systems, , Transition Sys- tems, Reduction, Simplification and Evaluation relations. Judgements and Rules. Rule Induction.

Scope: Identifiers, Scope and extent, Lexical scope, Environments, 'Dy- namic scope' and parameters. Closures

State: Stores and imperative constructs, explicit and implicit store refer- ences, objects, invariants and safety, interfaces and constructors, in- heritance, Parameter passing. Call-by-value, call-by-name and lazy evaluation.

Control: Tail calls, Contexts, continuations, continuation passing style, ex- ceptions, threads.

Types: Types syntax, type safety theorems. Type inference

Special Topics (if time permits): Monads, Concurrency.

5 Texts and References

Textbook

EOPL *Essentials of Programming Languages* 3rd Edition. Friedman and Wand. This is the main text for the course. Available on Amazon.in.

References

HtDP *How to Design Programs*. Felleisen et al. Available online.

SICP *Structure and Interpretation of Programs*. Abelson and Sussman. Available online. Accompanying video lectures also available online.

TRaAT *Term Rewriting and All That*. Baader and Nipkow. Chapters 1 and 2.

PLAI *Programming Languages: Application and Interpretation* 2nd Edi- tion. Available online.

SSICS *Simply Scheme: Introducing Computer Science*. Brian Harvey and Matthew Wright. Available online.

RG *Racket Guide*. Available as part of Racket language documentation.

6 Teaching and Learning strategies

Lectures will cover the theoretical aspects of operational semantics but will have plenty of examples explaining interpreters of programming langugaes visually and interactively. Question-answer discussion will accompany each class. Quizzes each week will test student's attention diligence, and con- cept recall, understanding and application. Summative assessments will be through a mid-semester and a final exam or project. Reading assignments will precede each lecture. Homework (programming) assignments will mostly involve implementation of interpreters discussed in the class and the text book. Tutorials will walk-through abstract syntax tree annotation, compo- nents of the interpreter implementation, and inductive proofs of properties in operational semantics.

7 Assessment (Tentative)

Item	Weight (%)
Quizzes (1 per week)	25
HW	25
Mid-semester exam	20
Final exam/Project	30

Appendix: Programme and Programme Specific Out- comes

Programme Outcomes

PO1 :: Engineering knowledge Use concepts from varied disciplines in- cluding Computer Science, Electronics, Mathematics, and the Sciences, to engineer and develop systems of varying scale.

PO2 Problem analysis Identify, formulate and analyze complex engineer- ing problems reaching substantial conclusions using first principles of Mathematics, Natural Sciences and Engineering Sciences.

PO3 Design/Development of solutions Identify and bring to fore the necessary concepts from Computer Science and arrive at creative ways to solve problems that take into account the societal, cultural, and ethical considerations.

PO4 Conduct investigations of complex problems Interpolate and ex- trapolate based on existing knowledge base and self-learning skills to investigate the dynamics of complex problems and find solutions.

PO5 Modern tool usage Demonstrate requisite hands-on skills to work with a variety of software packages, libraries, programming languages, and software development environment tools useful in engineering large scale systems

PO6 The engineer and society Make judicious use of resources and un- derstand the impact of technology across the societal, ethical, environ- mental, and economic aspects.

PO7 Environment and sustainability Find technological solutions by considering the environmental impact for sustainable development

PO8 Ethics Practice principles of professional ethics and make informed decisions after a due impact analysis.

PO9 Individual and team work Work efficiently in individual and team- oriented projects of varying size, cultural milieu, professional accom- plishments, and technological backgrounds.

PO10 Communication Effectively communicate and exchange ideas and solutions to any individual including peers, end-users, and other stake- holders.

PO11 Project management and Finance Apply the principles of project management in general and software project management in particu- lar with focus on issues such as the life cycle, scoping, costing, and development.

PO12 Life-long learning Exhibit the aptitude for independent, continu- ous, and life-long learning required to meet their professional and career goals.

Programme Specific Outcomes (PSOs)

PSO1 Exhibit specialized knowledge in some sub-areas of Computer Science and Engineering such as Theoretical Computer Science, Computer Sys- tems, Artificial Intelligence, Cyber-physical Systems, Cyber-security and use this specialized knowledge base to solve advanced problems

PSO2 Perform gap analysis in terms of systems and technologies and pre- pare roadmaps for incorporating state-of-the-art technology into sys- tem analysis, design, implementation, and performance.

PSO3 Demonstrate research and development skills needed to define, scope, develop, and market futuristic software systems and products.

PSO4 Demonstrate knowledge and skills at the required depth and breadth to excel in post-graduate and research programs.

Faculty Name: Ankit Gangwal
Email: gangwal@iiit.ac.in
and
Faculty Name: Kannan Srinathan
Email: srinathan@iiit.ac.in

Course Code: **CSE418**
Title of the Course: **Principles of Information Security**
Credits: **4**
L-T-P: **3-1-0 (L= Lecture hours, T=Tutorial hours, P=Practical hours)**

1. Prerequisite Course / Knowledge:
Basic principles of algorithms.

2. Course Outcomes (COs) :

After completion of this course successfully, the students will be able to..

CO-1 Discuss mathematical concepts of cryptographic primitives
CO-2 Describe fundamental concepts and algorithms of cryptography, including encryption/decryption and hash functions
CO-3 Summarize different authentication techniques and describe programs like PGP & S/MIME
CO-4 Discuss network security principles, applications, and practices
CO-5 Analyse protocols for various system security objectives using cryptographic tools
CO-6 Evaluate the role of different security mechanisms like passwords, access control mechanisms, firewalls, etc.

3. Detailed Syllabus:

Unit 1: Introduction: Security Trends, Security attacks, Security services, Security Mechanisms, A Model for Network Security Model, Classical Encryption Techniques, Symmetric Cipher Model, Substitution Techniques, Transposition Techniques, Rotor Machines, Steganography.

Unit 2: Block Ciphers and Data Encryption Standard: Block Cipher Principles, Data Encryption Standard, Strength of DES, Differential and Linear Cryptanalysis, Block Cipher Design Principles, Advanced Encryption Standard, Evaluation Criteria of AES,AES Cipher, Multiple encryption and Triple DES, Block Cipher Modes of Operation, RC4.

Unit 3: Public-key Encryption and Hash Functions: Principles of Public Key Cryptosystems, RSA Algorithm, Key Management, Message Authentication and Hash Functions, Authentication Requirements, Authentication Functions, Message Authentication, Hash Functions, Security of Hash Functions and MACs, Digital Signatures, Authentication Protocols, Digital Signature Standard.

Unit 4: Network Security Applications: Kerberos, X.509Authentication Service, Public Key Infrastructure, Pretty Good Privacy, S/MIME , IP Security Overview, IP Security architecture, Authentication Header, Encapsulating Security Payload, Combining Security associations, Key Management.

Unit 5: System Security: Secure Socket Layer and Transport Layer Security, Secure Electronic Transaction, Intruders, Intrusion Detection, Password Management, Malicious Software, Firewalls, Trusted Systems

Reference Books:

1. W. Stallings, Cryptography and Network Security Principles & Practices, 4th edition, Prentice Hall, 2005
2. J. Katz and Y. Lindell, Introduction to Modern Cryptography, CRC Press, 2007
3. B. Schneier, Applied Cryptography, 2nd edition, John Wiley & Sons, Inc, 2001
4. Research papers

4. Teaching-Learning Strategies in brief (4 to 5 sentences):

Lectures by integrating ICT into classroom teaching; tutorials involving problem solving; being a fundamental course, it requires critical thinking and active learning by the students to solve problems.

5. Assessment methods and weightages in brief (4 to 5 sentences):

Assignments	30 marks
Mid Semester Examination	30 marks
End Semester Examination	40 marks

TITLE : Optimization Methods
Course Code : CS1.404
CREDITS : 4 Credits
L-T-P : 3-1-0
TYPE-WHEN : Spring 2022
FACULTY NAME : Dr. Naresh Manwani
PRE-REQUISITE : Strict Prerequisites: NIL

EXPECTED BACKGROUND:
To follow this course, some level of familiarity with linear algebra (specially, vectors and matrices) is expected. In addition, student is expected to know the fundamentals of algorithms and some of the popular problems (eg. shortest path.)

OBJECTIVE:

1. To enable students to formulate and solve problems in an optimization framework.
2. To expose a set of powerful tools and techniques to the students. To demonstrate how these tools (i.e. optimization methods) can be used in practice.
3. To visualize the optimization algorithms and know the numerical and practical issues in their implementation.
4. To relate the optimization methods to applications in diverse areas.

COURSE TOPICS :

1. CO-1: Linear Programming, Geometric Interpretation, SimplexMethod, Duality, primal dual method, Interior point methods, Ellipsoidal methods, Computational Issues.
5. CO-2: Integer programming, LP relaxation, Examples from combinatorial optimization. Shortest paths, network flows and matchings.
6. CO-3: Convex sets and functions. Need for constrained methods in solving constrained problems.
7. CO-4: Unconstrained optimization, Optimality conditions, Gradient Descent, Newton Method, Quasi- Newton Methods, Trust Region Methods. Conjugate Gradient Methods. Least Squares Problems.
8. CO-5: Constrained Optimization, Optimality Conditions and Duality. Convex Programming Problem. Quadratic Programming. Dual Methods, Penalty and Barrier Methods, Interior Point Methods.
9. CO-6: Linear Equations, Solutions based Matrix Factorization, Singular Value Decomposition,

10. CO-7: **Additional topics** (if time permits) related to
    1. Specific Algorithms (eg. Cutting plane algorithms, Stochastic gradients)
    2. Applications in Approximate Algorithms
    3. Computational issues in large scale optimization
    4. Heuristic methods for optimization

PREFERRED TEXT BOOKS:

1. S. Boyd and L Vandenberghe, ``Convex Optimization'', Cambridge University Press (Online Copy available at: http://www.stanford.edu/~boyd/cvxbook/).
11. L Vandenberghe, Lecture Notes for Applied Numerical Computing, (Online available at: http://www.ee.ucla.edu/~vandenbe/103/reader.pdf).
12. Edwin K. P. Chong, Stanislaw H. Żak, Introduction to Optimization, Fourth Edition, Wiley-Interscience Series in Discrete Mathematics and Optimization, John Wiley & Sons.

REFERENCE BOOKS:

1. M T Heath, ``Scientific Computing'', TMH (Most of First six chapters)
13. C H Papadimitriou and K Steiglitz,Combinatorial Optimization: Algorithms andComplexity'' (Most of First seven chapters), Dover.
14. D Bertisimas and J N Tsitsiklis, ``Introduction to Linear Optimization'', Athena Scientific.
15. J Matousek and B. Gartner, ``Understanding and Using Linear Programming'', Springer, 2007.

OUTCOME:

This course will help in sharpen the problem solving skills of students. Students will have experience informally stating problems with the associated constraints, and solving them with computer friendly algorithms.

GRADING PLAN:

Type of Evaluation	**Weightage (in %)**
Small Quizzes (10 quizzes)	10%
Mid-Sem Exams (2)	30%
End Sem Exam	20%
Assignments	25%

Term Paper/Project	10%
Scribe	5%

Title of the Course: **Advanced Algorithms**

Name of the Faculty: Kishore Kothapalli,

Course Code:

L-T-P…… **3-1-0**

Credits…… 4

Prerequisite Course / Knowledge:

Should have taken Introduction to Algorithms, and Formal Languages, or equivalent courses

Course Outcomes (COs):

After completion of this course successfully, the students will be able to..

CO-1 :Demonstrate familiarity with using randomness in computing

CO-2: Apply principles of randomized algorithm design and analyze them for correctness and efficiency

CO-3: Synthesize randomized algorithms with either zero-error or one sided error for a variety of problems

CO-4: Explain the significance of parallelism to modern day computing and problem-solving needs

CO-5: Apply principles and paradigms of parallel algorithm design and analyze parallel algorithms for correctness and efficiency

CO-6: Create efficient parallel algorithms for a variety of semi-numerical problems and problems on graphs

Detailed Syllabus:

Unit 1: Randomness in computing: Tail inequalities and applications, fingerprinting, proofs of existence, expander graphs

Unit 2: randomized rounding, approximate counting

Unit 3: Parallelism in computing: Models of PRAM, Basic algorithms for prefix, search, sort, merge,

Unit 4: Parallel algorithms for lists, graphs,andsymmetry breaking

Reference Books:

1. R. Motwani and P. Raghavan (1995), Randomized Algorithms, Cambridge University Press.USA.
2. J. JaJa (1992), Introduction to Parallel Algorithms, Addison-Wesley, USA.
3. G. Tel (2000).Distributed Algorithms,2nd Edition, Cambridge University Press. USA.

Teaching-Learning Strategies in brief (4 to 5 sentences):

The course lectures will include activities that promote the understanding of the lecture content by using small examples that students work out during the class itself and promote active and participatory learning. A good part of the lecture will involve problem solving and finding solutions to problems rather than expositing known material. In class tests that are held periodically are useful as summative assessments. Homework assignments are designed to reiterate the material covered in class lectures and also solve problems that are based on simple extensions of concepts described in the lectures.

Assessment methods and weightages in brief (4 to 5 sentences):

- Homeworks: 20%
- In-class Objective Tests: 20%
- Quiz1: 15%
- Quiz 2: 15%
- End Exam: 30%

Title of the Course: Distributed Systems

Faculty Name: **Kishore Kothapalli**
Course Code: **CS3.401**
L-T-P **3-1-0**
Credits **4**
(L= Lecture hours, T=Tutorial hours, P=Practical hours)

1.Prerequisite Course / Knowledge:

An understanding of operating systems, networks, and algorithms

2.Course Outcomes (COs):

After completion of this course successfully, the students will be able to..

CO-1 : Explain the challenges faced by distributed systems in terms of lack of global time, synchrony, faults, programming support, etc.

CO-2 :Employ standard distributed programming frameworks to write distributed programs for problem solving

CO-3 : Explain the properties and design principles of various real-world and practical distributed systems

CO-4 :Interpret the impact of faults in distributed systems in the context of important problems such as distributed agreement, distributed consensus, and distributed transaction processing

CO-5 :Analyze distributed algorithms for graphs with respect to correctness, round complexity, and message complexity.

CO-6 : Analyze the limitations of distributed systems and assess the operational scope of large scale distributed systems

3.Detailed Syllabus:

- Unit 1
  - Introduction
  - Communication models
  - Time and Synchronization
  - Practice: MPI/Map-Reduce
- Unit 2
  - Distributed file systems
  - Consensus, Agreement, Locking
  - Practice: GFS, Chubby
- Unit 3
  - Distributed Database systems
  - Practice: NoSQL, MongoDB
- Unit 4
  - Limitations of distributed computing
  - Self-Stabilization
  - CAP Theorem
- Unit 5
  - Distributed algorithms for graphs
  - Advanced Topics such as Blockchain, Distributed Storage, and Distributed Program Verification

Reference Books:

1. A.D. Kshemkalyani, M. Singhal, (2011) Distributed Computing: Principles, Algorithms, and

Systems, ISBN: 9780521189842, paperback edition, Cambridge University Press, USA.
2. N. Lynch, 1996. Distributed Algorithms, Morgan Kauffman, USA, Chapter 5.
3. Other significant papers from conferences such as OSDI, USENIX, NSDI, for material that is not part of textbooks

4.Teaching-Learning Strategies in brief:

Lectures of the class use the active learning methodology and allow students to learn concepts thoroughly in class along with practising small examples. Homeworks assigned as part of the course are useful to impart knowledge of using practical distributed programming tools and libraries. To promote team work, some of the homeworks are done in a team of two students. The overall learning from the course is enhanced by doing a substantial practice-based project – usually in a team of two students. The course will also have a summative assessment in the form of a final/end-semester exam.

5.Assessment methods and weightages in brief :

i. In-class Quiz Exams (Cumulative over several): 15%
ii. Homeworks: 20%
iii. Project: 25%
iv. End Semester Examination: 40%

Title of the Course : Data Systems

Name of the Faculty : Kamal Karlapalem
Course Code : CS4.401
L-T-P : 3-1-1
Credits : 4
(L= Lecture hours, T=Tutorial hours, P=Practical hours)

1. Prerequisite Course / Knowledge:

Basic principles of Operating systems, Structured Query Language, Relational Data Model, Data structures, Programming language, Algorithms,

2. Course Outcomes (COs)

After completion of this course successfully, the students will be able to..

CO-1. Develop the tree-based and hash-based indexing algorithms to improve efficiency of the retrieval

CO-2. Tune the optimizer module of DBMS to meet the performance demands of diverse applications, including distributed applications.

CO-3: Design the recovery sub-system of any given information system

CO-4. Design archival strategy for any given information system

CO-5. Develop a concurrency control algorithm for any given database system

CO-6. Develop a framework for building a large scale big data system.

3. Detailed Syllabus:

Unit 1: Introduction, Data storage, Representing data elements (9 hours)
Unit 2: Index structures, Multidimensional indexes (7.5 hours);
Unit 3: Query execution, The query compiler (9 hours)
Unit 4: Coping with system failures, Concurrency control (7.5 hours);
Unit 5: Transaction management, NoSQL and big data systems (9 hours)

- Five mini projects related to the above syllabus will be done by students in the laboratory

References :

- Hector Garcia-Molina, Jeffrey D. Ullman and Jennifer Widom , Database System Implementation, Pearson Education, 2003
- Elmasri&Navathe, Fundamentals of Database Systems, 6th Edition, Pearson Education, 2013
- Raghu Ramakrishnan and Johannes Gehrke ,Database Management Systems, Third edition, Mc Graw Hill, 2017
- Abraham Silberschatz, Henry F.Korth, S.Sudarshan, Database system concepts, fifth edition, Mc Graw Hill, 2006
- Research papers

4.Teaching-Learning Strategies in brief:

Lectures by integrating ICT into classroom teaching, weekly tutorials involving problem solving and active learning by students and Project-based Learning by doing 5 mini-projects in laboratory by the students

5. Assessment methods and weightages in brief :

Assignments in theory: 10 marks, Quizzes in theory: 10 marks, Mid Semester Examination in theory: 20 marks , End Semester Examination in Theory: 30 marks, Assessment of 5 mini projects in Laboratory: 30 marks

---

Title of the Course:	**Compilers**
Faculty Name:	**Suresh Purini**
Course Code:	**CS1.403**
L-T-P:	**3-1-0.**
Credits:	**4**

(L = Lecture Hours, T = Tutorial Hours,
P = Practical Hours)

1. Prerequisite Course / Knowledge:

Computer Programming. Data structures and algorithms. Computer Systems Organization. Operating Systems. Automata Theory.

2. Course Outcomes (COs)

After completion of this course successfully, the students will be able to:

CO-1: Explain the principles and practices underlying production quality compilers such as GCC and LLVM (Cognitive Level: **Understand**)

CO-2: Modify open source compilers such as GCC and LLVM to support new languages and processor architectures; and write custom analysis and transformation passes. (Cognitive Leve: **Apply**)

CO-3: Identify problems or sub-problems in real world projects which can be solved by building custom compilers and interpreters of varying scale and complexity. (Cognitive Levels: **Analyze, Evaluate and Create**)

CO-4: Employ software engineering principles and practices to design, develop and manage complex software engineering tasks. Examples include object oriented design and programming, choosing appropriate design patterns, good support for debugging the system with ease and, develop comprehensive test suite with good coverage. (Cognitive Levels: **Analyze, Evaluate** and **Create**)

CO-5: Use software management tools such as GIT, build systems such as Make/Ant etc. Write proper software design documents and end-user manuals (Cognitive Levels: **Apply**)

3. Detailed Syllabus

- **Unit 1: Syntax Analysis**
  - Micro and macro syntax specification using regular expressions and context free grammars
  - Lexical Analysis
  - Top-down (LL(1)) and bottom-up (LR(1), LALR(1)) parsing
- **Unit 2: Semantic Analysis and IR Generation**
  - Abstract Syntax Tree (AST) construction
  - Static and Dynamically typed language
  - Type Checking
- **Unit 3: Intermediate Representations and their Generation**
  - Intermediate representations such as three address tuples, stack code
  - AST to linear intermediate representation generation
  - Basic blocks and control flow graphs
  - Static Single Assignment Form (SSA)
  - LLVM IR case study
- **Unit 4: Machine Independent Optimizations**
  - Local and regional optimizations using value numbering optimization as a case study
  - Global optimizations like constant propagation and dead code elimination
  - Data flow analysis theory and practice. Examples include Available expressions analysis and live variable analysis.

    - Compiler phase sequencing problem
- **Unit 5: Code Generation and Register Allocation**
    - Runtime environment for C-like programming languages
    - Scope and lifetime of variables. Parameter passing mechanisms.
    - Generating machine code with virtual registers from machine independent linear intermediate representation.
    - Local and global register allocation. Mapping virtual registers to physical registers.
    - Basics of instruction scheduling

Reference Books:

1. Keith Cooper and Linda Torczon. 2011. Engineering a Compiler, Second Edition. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Alfred V. Aho, Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman. 2006. Compilers: Principles, Techniques, and Tools (2nd Edition). Pearson.

4. Teaching-Learning Strategies in brief

The most important component of this course is the project in which students design a C like imperative programming language. Write a manual for their programming language specifying syntactic and semantic rules along with example programs written in their own language. Over the course, as students are introduced to principles and practices involved in designing various compiler modules, they build the corresponding modules for their programming language. At the end of the course, students will be able to run the example programs they have written by compiling them with the compiler built by them. The target language for the compiler is usually LLVM IR.

Through the mini homeworks, theoretical ideas introduced in the class are reinforced. Students get continuous support through tutorial sessions, office hours conducted by teaching assistants and the concerned faculty.

5. Assessment methods and weightages in brief

1. Mini Homeworks (7 to 8) : 15 percent
2. Course Project
    a. Syntax Analysis: 10 percent
    b. AST Construction: 10 percent
    c. Semantic Analysis: 10 percent
    d. LLVM IR Generation: 10 percent
3. Mid Term Quiz: 15 percent
4. Final Theory Exam: 30 percent

Title of the Course Advanced Computer Networks

Course Code: **CS3.402**

Faculty Name **Ankit Gangwal**

Credits: **4**

L-T-P **3-1-0 (L= Lecture hours, T=Tutorial hours, P=Practical hours)**

1.Prerequisite Course / Knowledge:

Basic principles of computer networks and algorithms.

2.Course Outcomes (COs) (5 to 8 for a 3 or 4 credit course):

After completion of this course successfully, the students will be able to..

CO-1 Demonstrate a familiarity with concepts of network management, standards, and protocols

CO-2 Discuss variousprivacy-enhancing techniques used in modern computer networks

CO-3 Apply the knowledge of distance-vector (RIPand IGRP) and link-state (OSPF and IS-IS) routing protocols to find routing paths for a variety of networks

CO-4 Analyse wireless LAN technologies including IEEE 802.11

CO-5 Design efficient routing protocols for advanced computer networks (e.g., SDN and ICN)

CO-6 Develop a framework for building a large-scale enterprise network

3.Detailed Syllabus:

Unit 1: Modeling and measurement: Network traffic modeling, network measurement, simulation issues, network coding techniques

Unit 2: Flow and congestion control, TCP variants, TCP modeling, active queue management

Unit 3: Routing: Router design, scheduling, QoS, integrated and differentiated services

Unit 4: Wireless networks: Mobility supports, MAC, multicast

Unit 5: Overlay networks and Emerging applications: SDN, ICN, P2P, CDN, Web caching, cross-layer optimizations, VoIP, SIP, video over P2P

Reference Books:

1. Larry L. Peterson and Bruce S. Davie, Computer Networks: A Systems Approach, 5th edition, Elsevier, 2012
2. James F. Kurose and Keith W. Ross, Computer Networking: A Top-Down Approach, 6th edition, Pearson Education, 2013
3. Jean Walrand and Pravin Varaiya, High-Performance Communication Networks, 2nd edition, Elsevier, 2000
4. Research papers

4.Teaching-Learning Strategies in brief (4 to 5 sentences):

Lectures by integrating ICT into classroom teaching; tutorials involving problem solving; being a systems course, it requires hands-on working as well as critical thinking and active learning by the students to solve practical problems; and finally, project-based learning by implementing semester-long project(s) to solve real-world issues.

5.Assessment methods and weightages in brief (4 to 5 sentences):

Assignments	20 marks
One at home project	30 marks

Mid Semester Examination	20 marks
End Semester Examination	30 marks

Title of the Course: Software Engineering
Course Code: CSE461
Faculty Name: **Raghu Reddy**
L-T-P: 3-0-1
Credits: 4
(L= Lecture hours, T=Tutorial hours, P=Practical hours)

1. Prerequisite Course / Knowledge:

Students must have taken Intro to Software Systems, Design and Analysis of Software Systems or Equivalent courses

2. Course Outcomes (COs) (5 to 8 for a 3 or 4 credit course):

After completion of this course successfully, the students will be able to…

CO-1: Demonstrate familiarity with various process models, design patterns, architecture patterns and the characteristics of good software architectures
CO-2 Apply principles of user interface design, sub-system design and analyze the designs for good Software Engineering principles
CO-3: Demonstrate the use of tools to quantitatively measure and refactor existing software systems
CO-4: Compare design trade-offs between different patterns and/or different implementations of the same pattern
CO-5: Design the major components and user interface for a small-scale software system using modeling approaches such as UML class diagrams, and sequence diagrams
CO-6: Critique the quality of a software design and use product quality metrics to assess the quality of delivered software

Note: Each Course Outcome (CO) may be mapped with one or more Program Outcomes (POs) and PSOs. Write '3' in the box for 'High-level' mapping, 2 for 'Medium-level' mapping, 1 for 'Low'-level' mapping

3 Detailed Syllabus:

Unit 1: Software Development Lifecyle and importance of architecture and design in the lifecycle, Process models; Modeling using UML.

Unit 2: Anti-patterns; Metrics and Measurement; Reverse Engineering and Refactoring.

Unit 3: Design Principles and Classification of Patterns

- Structural patterns: Adapter, Composite, Façade, Proxy, Decorator
- Behavioral patterns: Iterator, Observer, Mediator, Command, Memento, State, Strategy,

Chain of Responsibility

- Creational patterns: Abstract Factory, Builder, Singleton, Factory Method

Unit 4: Software architecture and Architectural business cycle; Quality attributes and Tactics for achieving attributes; Architectural styles and Techniques; Designing Architectures, Case studies.

Reference Books:

1. Design Patterns: Elements of Reusable Object- Oriented Software. E. Gamma, R. Helm, R. Johnson, and J. Vlissides. Pearson, 2015, ISBN-13 : 978-9332555402
2. Refactoring: Improving the Design of Existing Code. Martin Fowler. Addison-Wesley, 2018. ISBN-13 : 978-0134757599
3. Software Architecture in Practice, 3rd edition by Len Bass, Paul Clements and Rick Kazman, Addison- Wesley, 2012. ISBN-13 : 978-9332502307

4.Teaching-Learning Strategies in brief (4 to 5 sentences):

The course is delivered using project based learning methodology. Topics like software subsystems modeling, design analysis, design trade-offs, language agnostic designs and component-based software development are taught and reinforced via unit level projects. The lectures emphasize the study and development of software sub-systems, comprehension and analysis of design quality attributes. The focus is on application of these concepts to concrete design problems through in-class design exercises and analysis of existing designs of currently implemented software systems. Entire class is run in a studio mode to facilitate discussion between student teams and discuss design trade-offs among students within student teams. Students present their designs and implementations to other students who are expected critique the designs.

5.Assessment methods and weightages in brief (4 to 5 sentences):

Final Exam	22 %
Mid-term Quiz	12 %
Unit Questions	12 %
3 Unit Projects (2 * 17) + (1 * 10)	44 %
Other In-class Activities	10 %

Title of the Course: Statistical Methods in Artificial Intelligence

Faculty Name: **Anoop Namboodiri**

Course Code: **CS7.403**

L-T-P **3-1-0**

. Credits **4**

1 **Prerequisite Course / Knowledge:**

Should have taken Basic courses in maths (related topics: Linear Algebra,

Probability, Differential Calculus).

2. **Course Outcomes (COs):**

After completion of this course successfully, the students will be able to..

CO-1 : Demonstrate capability to model and represent physical entities as vectors (feature vectors) and carry out numerical computation.

CO-2: Formulate and solve many practical problems as classification and regression. Also appreciate other problem settings like clustering, structured prediction.

CO-3: Explain the fundamental mathematical ideas behind the popular machine learning algorithms

CO-4: Discuss the practical (computational) challenges in design and implementation of machine learning algorithms including (i) dimensionality reduction (ii) computational complexity (iii) convergence of the algorithm (iv) offline and online computation

CO-5: Apply the learnings on practical problems and real life data. Appreciate the challenges with the real world data sets.

CO-6: Discuss the nuances of conducting experiments, analyzing performances and expose the world of empirical science in computation.

3. **Detailed Syllabus:**

Unit 1: Representation, Vectors, Distributions, Dimensionality reduction, problems and challenges in machine learning

Unit 2: Basic algorithms in machine learning, PCA, Perceptrons, Decision Trees, Analysis

Unit 3: Popular algorithms and settings including unsupervised learning, Support Vector Machines, Kernels, Bias and Variance, Model Selection.

Unit 4: Neural Network Learning, Multi LayerPerceptrons, Backpropagation Algorithms, Exposure to Deep Learning.

Reference Books:

1. MDeisenroth, A. Faisal, C.Ong, Mathematics for Machine Learning, Cambridge Univ Press, 2020
2. R. Duda, P. Hart and D. Stock, Pattern Classification, Wiley, 2007
3. I Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016

4. **Teaching-Learning Strategies in brief (4 to 5 sentences):**

Course lectures will connect the algorithms and approaches to the real world examples. This motivates the student and also convince the need of formal and mathematical way of approaching the real world problem solving. Lectures also introduce the visualization skills of the data and distribution with the aim of appreciating the data. Associated sessions and components (tutorials, homeworks) expose the popular libraries and software infrastructure for machine learning today.

5. **Assessment methods and weightages in brief (4 to 5 sentences):**
   - Homeworks: 30%
   - In-class Objective Tests: 10%
   - Projects/Term Papers: 10%
   - Mid semester exam1: 15%
   - Mid Semester exam2: 15%
   - End SemesterExam: 20%

Title of the Course: Information Retrieval and Extraction

Name of the Faculty: **Vasudeva Varma**
Course Code: **CS4.406**
L-T-P: **3-1-1**
Credits: **4**
(L= Lecture hours, T=Tutorial hours, P=Practical hours)

1. Prerequisite Course / Knowledge:

Basic principles of Computer programming, Statistical Methods in Artificial Intelligence, Programming languages, and Algorithms.

2. Course Outcomes (COs)

After completion of this course successfully, the students will be able to..

CO-1. Develop algorithms to retrieve information from unstructured data
CO-2. Design and architect information retrieval systems for world wide web
CO-3: Design Web crawling systems
CO-4. Design algorithms to process noisy data in document repositories
CO-5. Develop information extraction systems

3. Detailed Syllabus:

Unit 1: Introduction to Information retrieval, Information Extraction and Information Access systems. (6 hours)
Unit 2: Information Retrieval Models and Evaluation of IR systems (7.5 hours);
Unit 3: Web Information Retrieval (4.5 hours)
Unit 4 Natural Language Processing in IR (7.5 hours)
Unit 5: Machine Learning in Information Retrieval Systems (12 hours)

Unit 6: Information Extraction (4.5 Hours)
Unit 7: IR Applications (12 Hours)
References :

- Introduction Information Retrieval – Chris Manning et al (the Stanford IR Book) (ISBN-13: 978-0521865715)
- Search Engines: IR in Practice – Bruce Craft et al (ISBN-13: 978-0136072249)
- Research papers

4.Teaching-Learning Strategies in brief:

Lectures by integrating ICT into classroom teaching, weekly tutorials involving problem solving and active learning by students and Project-based Learning by doing one mini-project and a major project by the students

5. Assessment methods and weightages in brief :

Assignments in theory: 10 marks
Quizzes in theory: 10 marks
Mid Semester Examination: 20 marks
End Semester Examination: 60 marks

Title of the Course: Advanced Natural Language Processing

Faculty Name: **Manish Shrivastava**
Course Code : **CS7.501**
Credits: **4**
L - T - P : **3-1-0**
(L - Lecture hours, T-Tutorial hours,
P - Practical hours)
Semester, Year: **Monsoon, 2022**
(Ex: Spring, 2022)

Pre-Requisites **: None**

Course Outcomes :

After completion of this course successfully, the students will be able to –

I. Demonstrate the knowledge of Advanced building blocks of NLP
II. Apply NLP machine learning algorithms for Machine Translation, Summarization
III. Demonstrate the knowledge of Dense and contextual representation for NLP
IV. Explain the concepts behind Deep Learning models
V. Discuss the approaches to global and contextual semantic representation
VI. Apply the above concepts for fundamental NLP tasks.

Course Topics :

A. Distributed Semantics
    - Contextual Distributed Semantics
B. Models such as ELMO, BERT, ERNIE and their derivatives
C. Statistical Machine Translation methods
    - Early Neural Machine Translation models
D. Extractive and Abstractive Summarization
    - Neural Summarization Methods
E. Reinforcement learning for NLP

Preferred Text Books :
None. Mostly research papers.

Reference Books :
Statistical Machine Translation by Philip Koehn
Deep Learning by Ian Goodfellow

E-book Links :
1. https://www.deeplearningbook.org/
2. **http://www.cs.cmu.edu/~tom/mlbook.html**

Grading Plan :
(The table is only indicative)

Type of Evaluation	Weightage (in %)
Quiz-1	**2.5**
Mid SemExam	**10**
Quiz-2	**2.5**
End Sem Exam	**20**
Assignments	**15**
Project	**40**
Term Paper	**10**
Other Evaluation	

Teaching-Learning Strategies in brief (4-5 sentences) :

Lectures by integrating ICT into classroom teaching, weekly tutorials involving problem solving and active learning by students and Project-based Learning by doing fourassignments and a project. Evaluation based on personal viva to judge deeper understanding.

Title of the course **Data Analytics-I**

Course Code **CS4.405**

FACULTY NAME **P. Krishna Reddy**

L-T-P **3-1-1**

Credits: **4**

(L= Lecture hours, T=Tutorial hours, P=Practical hours)

TYPE-WHEN Fifth semester and onwards

1.<u>Prerequisite Course / Knowledge</u>:

1. Data and Applications, *or equivalent courses that cover Data modelling, normalization, SQL*
2. First courses on programming, data-structures and algorithms
3. Basics of Python language, to be able to use relevant libraries and toolkits for data analytics

2.<u>Course Outcomes (COs)</u>

Objective: In a computerized and networked society, vast amount of data is being collected every day in multiple domains. We are drowning in data, but starving for knowledge or actionable insights. Data mining or data analytics constitute a collection of concepts and algorithms, which are being developed to answer "how" questions by extracting interesting and useful knowledge of from large data. Data analytics based platforms are being operated in multiple domains to extract valuable and actionable insights from the data to improve the business performance. The objective of this first level course is to learn the important concepts and algorithms related to data mining functionalities such as summarization, pattern mining, classification, clustering and outlier analysis.

The Course Outcomes (COs) are as follows:

1. After completing the course successfully, the students are able to
    1. CO-1. describe the concepts of data summarization, data warehousing, pattern mining, classification and clustering approaches
    2. CO-2. perform the task of data summarization, pattern mining, classification and clustering based on the requirement.
    3. CO-3. prescribe a single or a combination of data summarization, pattern mining, classification and clustering approaches for the problem scenario of a business/organization.
    4. CO-4. construct the improved data analytics methods for existing services.
    5. CO-5. formulate new data mining problems for creating new services and design the corresponding solutions

3.Detailed Syllabus

(please list the order in which they will be covered)
Unit 1: Introduction, data summarization through characterization, discrimination and data warehousing techniques (9 hours)
Unit 2: Concepts and algorithms for mining patterns and associations (9 hours) Unit 3: Concepts and algorithms related to classification and regression (9 hours) Unit 4: Concepts and algorithms for clustering the data (9 hours)
Unit 5: Outlier analysis and future trends. (3 hours)

- ive mini projects related to the above syllabus will be done by students in the laboratory

Reference Books and materials:

1. Book: Jiawei Han and Micheline Kamber, Data Mining: Concepts and Techniques, Third edition, 2012, Elseiver Inc.
2. Book: Pang-Nong Tan, Michael Steinbach and Vipin Kumar, Introduction to Data Mining, 2006, Pearson Education.
3. Research Papers: About 25 research papers from the proceeding of the conferences and journals related to data summarization, data warehousing, pattern mining, classification, clustering, outlier detection.

4.Teaching-Learning Strategies in brief

Lectures by integrating ICT into classroom teaching, weekly tutorials involving problem solving and active learning by students and Project-based Learning by doing 5 mini-projects in laboratory by the students

5.Assessment methods and weightages in brief

Two Class Room tests: 10 marks; Mid Semester Examination in theory: 20 marks , End Semester Examination in Theory: 40 marks, Assessment of 5 mini projects in Laboratory: 30 marks

---

Title of the Course: **Intro to Cognitive Science**
Course Code: CS9.426
Name of the Faculty: Vishnu Sreekumar
TAs: **Nancy**Hada and KumarNeelabh
Day/Time: Mondays and Thursdays: 2:00 pm – 3:25 pm.

Course Information
Course Description: Cognitive Science is a highly interdisciplinary field of study that seeks to understand how the mind works. In this course, we will discuss a diverse range of perspectives from philosophy, linguistics, psychology, neuroscience, and computer science, on how to unravel the mysteries of human cognition.

Credits: 4
L-T-P: 3-1-0 (L = lecture hours, T = tutorial hours, P = practical hours)

Prerequisite: None

Textbook & Course Materials

Recommended Texts & Other Readings: Lecture slides and supplementary readings will be posted to Moodle.

Course Technology Requirements

1. You will need access to the following tools to participate in this course.
    - Laptop/desktop computer
    - webcam
    - microphone
    - a stable internet connection (don't rely on cellular)

Course Structure

This course will be delivered fully in-person in a physical classroom unless COVID restrictions make us move online (Microsoft Teams).

Student Expectations

In this course you will be expected to complete the following types of tasks.

2. communicate via email
3. complete basic internet searches
4. download and upload documents to the course site on Moodle
5. read documents online
6. view online videos
7. participate in online discussions
8. complete quizzes/tests online
9. upload documents to a Dropbox/Moodle
10. participate in synchronous online discussions

Expected Instructor/TA Response Times

- We will attempt to respond to student emails within 24 hours. If you have not received a reply from us within 24 hours, please resend your email.
    - ***If you have a general course question (not confidential or personal in nature), please post it to the Course Q&A Discussion Forum found on the course homepage on Moodle. We will post answers to all general questions there so that all students can view them. Students are encouraged to answer each other's questions too.
- We will attempt to reply to and assess student discussion posts within 48 hours.

Course Outcomes (COs)

After successful completion of this course, students will be able to:

11. CO-1: demonstrate familiarity with seminal research findings in cognitive science.
12. CO-2: read, interpret, critique, and evaluate research in cognitive science.
13. CO-3: critically think about the relationship between diverse fields such as AI, philosophy, neuroscience, and cognitive science.
14. CO-4: identify flaws in how scientific results are communicated and critique scientific work in terms of confounds, experimental design, etc.
15. CO-5: appreciate the nature of scientific debate in cognitive science and be able to generate well-informed perspectives on these debates.

    You will meet the outcomes listed above through a combination of the following activities in this course:
16. Attend lectures and participate in class discussions (CO-1, CO-2, CO-3, CO-4, CO-5)
17. Debate sessions (CO-1, CO-2, CO-3, CO-5)
18. Quiz 1, Quiz 2, mid-semester, and end-semester exams (CO-1, CO-2, CO-3, CO-5)
19. Complete a term paper/debate reaction paper (CO-1, CO-2, CO-3, CO-5)

List of topics and activities

20. **Introduction**
21. **Evolution of Cognitive Science**
22. **A free-form discussion on consciousness**
23. **Empirical approaches in cognitive science**
24. **Brain: Organization; Intro to sensation and perception**
25. **Sensory systems**
26. **Perception and Perceptual Learning, Cross-modal interactions**
27. **Vision**
28. **Attention**
29. **Learning**
30. **Development**
31. **Memory**
32. **Language and Cognition**
33. **Knowledge Representation**
34. **Special topics: e.g. Music, mind, and technology**
35. **Several debate sessions with student debate teams**

Grading Policies

Graded Course Activities

Description	Percentage
Quiz 1(10 marks)	10%
Quiz 2 (10 marks)	10%
Debate reaction paper or debate team participation (20 marks)	20%
Mid-Sem exam(20 marks)	20%
End semester exam (40 marks)	40%
Total (100 marks)	100

Quizzes

Quiz 1 will cover topics covered until Quiz 1, and Quiz 2 will cover topics taught between Quiz 1 and Quiz 2. They will contain mostly multiple choice questions.

Mid-semester exam (20 marks)

The mid-semester exam will cover all material taught up to that point, and may include both multiple choice and descriptive questions.

End semester exam (40 marks)

The end semester exam will cover material taught during the whole semester and will include both multiple choice and descriptive type questions.

Debate participation (20 marks = 10 marks for presenting + 10 marks for a short report)

We will reserve at least 3-4 lecture slots for student debates on contemporary issues in Cognitive Science. A list of representative topics are as follows:

1. Are there top-downinfluences on basic perception? Evidence for and against.
2. Do 3 year olds have a theory of mind?
3. Is cognition/consciousness a computational process?
4. Do we need representations for cognition?

Each debate team will have 3 members. They will read the recommended material for the chosen topic, and organize their arguments distributed across the 3 members. Each member gets 5 minutes to present their arguments (15 minutes per team). They may choose to use slides or not but the arguments must be clearly presented. At the end of both teams' presentations, each team gets 5 minutes for rebuttal when they can pick 2-3 claims made by the opposite team and present counterarguments.

The students participating in debate teams will only be required to write a short report but the ***remaining students will need to write a reaction paper to any one debate session OR write a term paper on any other topic that they choose (see next main section).***

For debate team students (each person writes this separately without discussion with other team members, plagiarism software will be used to check your work), your short report should contain

the following:
The paper will first summarize the problem (2 marks), and then summarize the arguments made by both sides (3 marks), and then will provide the student's OWN opinion about where they stand on the debate and what arguments were convincing to them (5 marks).

Recommended: 2-3 pages, font size 12, single-spaced.
The debate teams will be made on a first-come first-serve basis. TAs will open sign-up forms and make announcements on the course page on Moodle. It is important to check announcements on Moodle regularly for this reason.
Submission window for the short report: Nov 1-10
No extensions will be given because this is a wide window.
You are welcome to make multiple submissions within this window.
IMPORTANT: See the last section of this syllabus for policies about plagiarism. There will be no exceptions to those policies.

Term Paper or debate reaction papers for non-debate team students (20 marks)

1. Introduction and clarity of describing the background literature and specifying the nature of the problem – 3 marks
2. Describing the different schools of thought that tackle the question – 7 marks
3. Offer your own thinking on the matter (either siding with one school of thought, or offering a new insight or suggestions for experiments or investigations, providing appropriate justifications) – 5 marks
4. Overall clarity, organization of thoughts, and originality – 3 marks
5. Formatting (Citations, References) – 2 marks

Recommended: 8-10 pages, font size 12, single-spaced.
Submission window for the term paper/debate reaction paper: Nov 1-10
No extensions will be given because this is a wide window.
You are welcome to make multiple submissions within this window.

Participation

Students are expected to participate in all activities as listed on the course calendar. *Failure to participate will result in students being unable to complete the term paper satisfactorily. The exams may also include questions from the in-class activities such as the debates and any resulting effect on the final grade is entirely the student's responsibility.*

Complete Assignments

All assignments for this course will be submitted electronically through the course page on Moodle unless otherwise instructed. Assignments must be submitted by the given deadline or special permission must be requested from instructor *before the opening of the submission window with documented evidence of an emergency*.

Late or missing assignments will affect the student's grade.

Late Work Policy

Be sure to pay close attention to deadlines—there will be no make-up assignments or quizzes, or late work accepted without a serious and compelling reason and instructor approval.

Viewing Grades on Moodle

Points you receive for graded activities will be posted to the course page on Moodle. Click on the Grades link to view your points.

Letter Grade Assignment

Final grades assigned for this course will be based on the percentage of total points earned and are assigned as follows:

Letter Grade	Percentage
A	[92,100]%
A-	[84,92)%
B	[76,84)%
B-	[68,76)%
C	[60,68)%
C-	[52,60)%
D	[45,52)%
F	< 45%

IMPORTANT NOTE:[x,y) indicates that x is included (square bracket) in the range and y is not (curly bracket). The normal rules of rounding will apply: So if you get 75.5, it will be rounded to 76 and you will get a B. However, if you get 75.444, it can only be rounded downwards and hence the final grade will be B-. No disputes on this matter will be entertained and such emails will not get a response.

Course Policies

Netiquette Guidelines

Netiquette is a set of rules for behaving properly online. Your instructor and fellow students wish to foster a safe online learning environment. All opinions and experiences, no matter how different or controversial they may be perceived, must be respected in the tolerant spirit of academic discourse. You are encouraged to comment, question, or critique an idea but you are not to attack an individual. Working as a community of learners, we can build a polite and respectful course community.

The following netiquette tips will enhance the learning experience for everyone in the course:

1. Do not dominate any discussion.
2. Give other students the opportunity to join in the discussion.
3. Do not use offensive language. Present ideas appropriately.

4. Be cautious in using Internet language. For example, do not capitalize all letters since this suggests shouting.
5. Avoid using vernacular and/or slang language. This could possibly lead to misinterpretation.
6. Never make fun of someone's ability to read or write.
7. Share tips with other students.
8. Keep an "open-mind" and be willing to express even your minority opinion. Minority opinions have to be respected.
9. Think and edit before you push the "Send" button.
10. Do not hesitate to ask for feedback.
11. Always assume good intentions and ask for clarification. Communication online is difficult without facial and gestural cues.

Adapted from:

Mintu-Wimsatt, A., Kernek, C., &Lozada, H. R. (2010). *Netiquette: Make it part of your syllabus*. Journal of Online Learning and Teaching, 6(1). Retrieved from http://jolt.merlot.org/vol6no1/mintu-wimsatt_0310.htm

Shea, V. (1994). Netiquette. Albion.com. Retrieved from: http://www.albion.com/netiquette/book/.

Build Rapport

If you find that you have any trouble keeping up with assignments or other aspects of the course, make sure you let your instructor know as early as possible. As you will find, building rapport and effective relationships are key to becoming an effective professional. Make sure that you are proactive in informing your instructor when difficulties arise during the semester so that we can help you find a solution.

Inform Your Instructor of Any Accommodations Needed

If you have a documented disability and wish to discuss academic accommodations, please contact your instructors as soon as possible.

Statement of Policy

The instructors of this course will modify requirements as necessary to ensure that they do not discriminate against qualified students with disabilities. The modifications should not affect the substance of educational programs or compromise academic standards; nor should they intrude upon academic freedom. Examinations or other procedures used for evaluating students' academic achievements may be adapted. The results of such evaluation must demonstrate the student's achievement in the academic activity, rather than describe his/her disability.

If modifications are required due to a disability, please inform the instructor

Commit to Integrity

As a student in this course (and at IIIT Hyderabad) you are expected to maintain high degrees of professionalism, commitment to active learning and participation in this class and also integrity in

your behavior in and out of the classroom.

IIIT Hyderabad Academic Honesty Policy & Procedures

Student Academic Disciplinary Procedures

(1) Academic misconduct is an act in which a student:

(a) Seeks to claim credit for the work or efforts of another without authorization or citation;

(b) Uses unauthorized materials or fabricated data in any academic exercise;

(c) Forges or falsifies academic documents or records;

(d) Intentionally impedes or damages the academic work of others;

(e) Engages in conduct aimed at making false representation of a student's academic performance; or

(f) Assists other students in any of these acts.

(2) Examples of academic misconduct include, but are not limited to: cheating on an examination; collaborating with others in work to be presented, contrary to the stated rules of the course; submitting a paper or assignment as one's own work when a part or all of the paper or assignment is the work of another; submitting a paper or assignment that contains ideas or research of others without appropriately identifying the sources of those ideas; stealing examinations or course materials; submitting, if contrary to the rules of a course, work previously presented in another course; tampering with the laboratory experiment or computer program of another student; knowingly and intentionally assisting another student in any of the above, including assistance in an arrangement whereby any work, classroom performance, examination or other activity is submitted or performed by a person other than the student under whose name the work is submitted or performed.

We will be using plagiarism detection software. Please do not copy-paste from other papers. If you use direct quotes, you have to use the quotation marks “xyz” and cite your source: e.g. (Johnson & Johnson, 1988, p. 5). Please use APA format. If plagiarism is detected, for the first violation, you will get 0 for the term paper or assignment in question. If plagiarism is detected a second time in another assignment/project write-up, then one letter grade will be deducted from the final grade (e.g if you get a B/B-, that will be changed to C/C-) and you will be reported to the appropriate authorities for further disciplinary action.

Note: This syllabus was adapted from a template provided at www.uwsp.edu

---

Title of the Course: **Spatial Data Science**
Faculty Name: **K S Rajan**
Course Code: **(PG-2 level course)**
Credits: **4**
L - T - P: **3-1-0**
(L - Lecture hours, T-Tutorial hours, P - Practical hours)
Semester, Year: **Spring 2022**
(Ex: Spring, 2022)

Pre-Requisites : Basic understanding of Locational Data and Computing – Any UG3, UG4, M.Tech., MS, and Ph.D. student should be able to take it.
Prior course work in Spatial Informatics may help.

Course Outcomes :
CO-1: Describe how Spatial Data Science helps uncover patterns
CO-2: Apply Geospatial techniques to Prepare the data for analysis
CO-3: Analyze the spatial and temporal data and interpret its outcomes
CO-4: Assessment of application of Spatial data science in key domain areas
CO-5: Design research projects that helps synthesize the learning into an application

Course Topics :
Module 1: Introduction to Spatial Data Science

- What is special about Spatial Data and Geo-AI?
- How Spatial and Spatio-temporal Big Data helps uncover patterns?
- Spatial Data Handling including spatial data models, data formats
- Challenges to computing approaches when applied to Spatial Data – Effects of Topology

Module 2: Geospatial Data Analysis and Modelling

- Vector Data Spatial Analysis
- Raster Data Spatial Analysis
- How to use temporal data in conjunction with Spatial data
- GeoSpatial Data Modelling Module 3: Spatial Sciences
- Spatial Statistics including Spatial auto-correlation, Spatial tessellation

o Data Mining applications on Spatial data including Spatio-temporal Data Mining

- Network Analysis and Graph theory
- Few relevant topics from Computational Geometry
- Geovisualization – Maps to WebGIS Module 4: Spatial Classification and Prediction
- Spatial decision trees
- Machine learning as applied to Spatial Data including Spatial-aware Neural Networks
- Hotspot Analysis
- Spatial Outliers detection

Module 5: Applications of Spatial Data Science

- Public Health – monitoring and mapping diseases, risk analysis and disease spread modelling
- Agriculture – crop growth monitoring, crop yield patterns and resource constraints
- Location based services – routing applications, ride-sharing algorithms, optimal location

Preferred Text Books :

1. Spatial Computing, By Shashi Shekar and Pamela Vold. The MIT Press. 2020

2. GIS – A computing perspective. By Micheal Worboys and Matt Duckham. CRC Press; 2nd edition 2004
3. Spatial Databases: A Tour. By S. Shekhar and S. Chawla, Prentice Hall, 2003, ISBN 013-017480-7 .
4. Selected Research Papers and Articles (will be shared with the topics taught on the course portal)

Reference Books :

1. Geographical Data Science and Spatial Data Analysis - An Introduction in R. By Lex Comber and Chris Brunsdon. SAGE Publications Ltd. 2020

E-book Links : Will be provided in Class as appropriate

Grading Plan :

Type of Evaluation	Weightage (in %)
Class Quizzes	**15.0**
Mid Sem Exams – 2	**20.0**
End Sem Exam	**30.0**
Paper reviews and Presentations by each Student in Class	**10.0**
Project/Term paper demonstrating the Practical applications	**25.0**

Teaching-Learning Strategies in brief (4-5 sentences) :

Teaching - Learning
Lectures Guest Lectures
Reading research papers
Class participation in Q&A, discussions Online discussions over MS Teams
Learning by doing Short Presentation and Discussion led by Student Course project on conceptualization and implementation Real world applications Multi-disciplinary approach

---

Title of the Course: Real-Time Embedded Systems

Faculty Name: **Deepak Gangadharan**
Course Code : **CS3.502**
L-T-P: **3-1-0**
Credits: 4
(L=Lecture hours, T=Tutorial hours, P=Practical hours)

1.Prerequisite Course/Knowledge

Computer Systems Organization, Basics of Operating Systems

2.Course Outcomes (COs)

After completion of this course successfully, the students will be able to

CO-1. Explain the features of real-time systems and classify different types of real-time systems such as hard real-time, soft-real time based on the timing requirements.
CO-2. Apply an appropriate task model (such as periodic, sporadic, aperiodic, etc) based on task/application characteristics to model a real-time system.
CO-3. Analyze the schedulability of a real-time system with different types of scheduling algorithms (static vs dynamic, preemptive vs non-preemptive) on a uniprocessor
CO-4. Analyze the schedulability of a real-time system with different types of scheduling algorithms (global, partitioned, semi-partitioned) on a multiprocessor platform **CO-5**. Analyze the schedulability of a real-time system with shared resources
CO-6. Assess the theory and experimental results presented in a relevant research paper and present it.
CO-7. Develop scheduling algorithms in a RTOS simulator

3.Detailed Syllabus

Unit 1: Real-Time Systems – Introduction and Concepts, Modeling Real-Time Systems **Unit 2:** Commonly used approaches to Real-Time Scheduling – Clock Driven approach, Weighted Round Robin approach, Priority Driven Approach, Dynamic vs Static Systems, Offline vs Online Scheduling, Preemptive vs Non-Preemptive
Unit 3: Clock Driven Scheduling – Scheduling Aperiodic and Sporadic Jobs, Schedulability test
Unit 4: Priority Driven Scheduling – Static Priority: Rate Monotonic and Deadline Monotonic Algorithms, Dynamic Priority: EDF Algorithm, Schedulability tests
Unit 5: Scheduling Aperiodic and Sporadic jobs in Priority Driven Systems – Deferrable Server, Sporadic Server, Constant Utilization Server, Total Bandwidth Server and Weighted Fair Queuing Server
Unit 6: Multiprocessor Scheduling
Unit 7: Resources and Resource Access Control

Reference Books:

1. Jane W S Liu, Real-Time Systems, Pearson Education
2. Giorgio C Buttazo, Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications, 3rd edition, Springer
3. C.M. Krishna & Kang G. Shin , Real Time Systems, McGraw Hill

4.Teaching-Learning Strategies in brief

Weekly lectures cover the topics in the syllabus and the advanced topics from research in real-time systems. Tutorials cover how to solve some design and analysis problems related to topics covered in the lectures. There are couple of assignments that will provide the students experience in

programming schedulers for RTOS platforms. There is a project which is either based on an idea the student wants to explore from the course topics or based on an existing research paper implementation and evaluation. Finally, there will be a presentation/discussion of a research paper.

5.Assessment methods and weightages in brief

Type of Evaluation	**Weightage (in %)**
Quizzes	**15**
Assignments	**15**
End Sem Exam	**20**
Project	**30**
Research Paper Presentation	**20**

Title of the Course: **Mechatronics System Design**
Faculty Name: Nagamanikandan
Course Code:
L-T-P: **3-1-0**
Credits: **4**
(L= Lecture hours,
T=Tutorial hours, P=Practical hours)

1.Prerequisite Course / Knowledge:
Basic programming (Python, C++), Linear Algebra, Numerical methods, Basic microcontroller knowledge.

2.Course Outcomes (COs):

After completion of this course successfully, the students will be able to CO-1 Describe important elements of mechatronics system
CO-2 Apply the previous knowledge of microcontroller programming for controlling multidisciplinary mechatronic systems.
CO-3 Describe and design basic mechanical elements and their feedback control. CO-4 Synthesize and analyze a range of mechanisms.
CO-5 Design and execute a multidisciplinary project based on the given specifications as part of a team.

3.Detailed Syllabus:
Unit 1: Sensors and Actuators:
Sensors for robotics application - position, speed, acceleration, orientation, range. Actuators -

general characteristics, motors, control valves.

Unit 2: Computer based feedback control:
Sampled data control, sampling and hold, PID control implementation, stability, bilinear transformation.

Unit 3: Introduction to mechanical elements and transformations, basic concepts of kinematics and dynamics.

Unit 4: Design and analysis of mechanisms.

Unit 5: Programming and hardware experiments.

Reference Books:

a. Bentley, John P. "Principles of measurement systems," Pearson education, 2005.
b. D.R. Coughanowr, "Process system analysis and control," McGraw Hill, 1991
c. G.F. Franklin, J.D. Powell and M.L. Workman, "Digital control of dynamic systems", Addison Wesley, 3rd edition, 1998.
d. Hartenberg, R., & Danavit, J, "Kinematic synthesis of linkages," McGraw Hill, 1964.
e. http://wiki.ros.org/
f. User manual of microcontroller and data sheets of sensors and actuators

4.Teaching-Learning Strategies in brief:

This course aims to teach the students about designing and developing a mechatronics system by providing them with essential hardware and software. Part of the class is devoted to a learn-by-doing lesson where the students will learn theory and get hands-on experience with various aspects of the mechatronics system. ebug the electromechan
The goal for the students is to design, build, and dical system for a given task as a part of the course project.

5.Assessment methods and weightages in brief:

Mid semester exam 20%

Assignments 40%
The class work assignments will be based on the application of a step-by-step engineering design process to a problem assigned in the course.

Project 40%
Proposal (5%)

Project demonstration (25%)
Final report (10%)

