Proposed CSE UG Curriculum

1. Preamble

The CSE Dept. proposes to define the B.Tech and DD curriculum by re– introducing Pre-requisite and co-requisite courses and introducing floating core courses. The floating core courses are associated with either even or odd semesters (under the assumption that a core course is offered only once a year.) A student has the freedom of doing a core course in any year, subject to pre-requisite and co-requisite requirements being satisfied. Floating courses are marked with a * in the subsequent section.

2. The Complete B.Tech Programme

This section lists the complete programme. Note that the course numbers are tentative and have been borrowed from existing courses wherever available.

Over all credit structure of B.Tech. Programme is as follows.

Department	Institute	Total Credits
134	115	249

Semester wise credit details are given below.

Semester 1						
Number	Course Title	L	T	P	C	Inst./Dept.
MA 105	Calculus	3	1	0	8	Inst.Core
CH 101	Chemistry I	2	1	0	6	Inst.Core
CS 101	Introduction to computer programming	2	0	2	6	Inst.Core
HS 101	Economics	3	0	0	6	Inst.Core
CH 115	Chemistry Lab	0	0	3	3	Inst.Core
ME 111	Workshop Practice	1	0	3	5	Inst.Core

Semester 2						
Number	Course Title	L	T	P	C	Inst./Dept.
MA106+ MA 108	Linear Algebra and Ordinary Differential Equations	3	1	0	8	Inst.Core
PH 102	Modern Physics	2	1	0	6	Dept.Option
XX 102	Data Analysis and Interpretation	2	1	0	6	Inst.Core
CS 152	Abstractions and Paradigms in Programming	3	0	0	6	DIC
PH 115	Physics Lab	0	0	3	3	Inst.Core

						Semester 2
CS 154	Abstractions and Paradigms in Programming Lab	0	0	3	3	DIC
ME 118	Engineering Graphics and Drawing	1	0	0	5	Inst.Core

Semester 3						
Number	Course Title	L	T	P	C	Inst./Dept.
MA 214	Numerical Analysis	3	1	0	8	Inst.Core
EE 101	Introduction to Electrical and Electronic Circuits	3	1	0	8	Inst.Core
CS 207	Discrete Structures	3	0	0	6	Dept.Core
CS 213	Data Structures and Algorithms	3	0	0	6	Dept.Core
XX 115	Experimentation and Measurement Lab	0	0.5	3	4	Inst.Core
CS 293	Data Structures and Algorithms Lab	0	0	3	3	Dept.Core

Semester 4						
Number	Course Title	L	T	P	C	Inst./Dept.
ES 403	Environmental Studies	3	0	0	6	Inst.Core
	Automata Theory and Logic*	3	0	0	6	Dept.Core
	Design and Analysis of Algorithms*	3	0	0	6	Dept.Core
	Logic Design	3	0	0	6	Dept.Core
CS 296	Software Systems Lab	2	0	2	6	Dept.Core
	Logic Design Lab	0	0	3	3	Dept.Core

Semester 5						
Number	Course Title	L	T	P	C	Inst./Dept.
HS 102	Literature/Philosophy/Psychology/Sociology	3	0	0	6	Inst.Core
	Computer Architecture *	3	0	0	6	Dept.Core
CS 347	Operating Systems *	3	0	0	6	Dept.Core
CS 317	Database and Information Systems*	3	0	0	6	Dept.Core
CS 387	Database and Information Systems Lab*	0	0	3	3	Dept.Core
	Computer Architecture Lab*	0	0	3	3	Dept.Core
CS 377	Operating Systems Lab*	0	0	3	3	Dept.Core

Semester 6						
Number	Course Title	L	T	P	C	Inst./Dept.
CS 344	Artificial Intelligence *	3	0	0	6	Dept.Core
	Implementation of Programming Languages *	3	1	0	8	Dept.Core
CS 348	Computer Networks *	3	0	0	6	Dept.Core
CS 386	Artificial Intelligence Lab*	0	0	3	3	Dept.Core
	Implementation of Programming Languages Lab*	0	0	3	3	Dept.Core
CS 378	Computer Networks Lab*	0	0	3	3	Dept.Core

Semester 7						
Number	Course Title	L	T	P	C	Inst./Dept.
	Elective 1	3	0	0	6	
	Elective 2	3	0	0	6	
	Elective 3	3	0	0	6	
	Institute Elective 1	3	0	0	6	

Semester 8						
Number	Course Title	L	T	P	C	Inst./Dept.
	Elective 4	3	0	0	6	
	Elective 5	3	0	0	6	
	Elective 6	3	0	0	6	
	Institute Elective 2	3	0	0	6	

If we do not count CS 101 (Institute Core) and Abstractions and Paradigms in Programming (Department Introductory course) among department courses, then we have the following structure:

- 11 Core CSE theory courses
- 9 Core CSE laboratory courses
- 6 Electives
- 7 possible empty slots for doing additional courses

- Flexibility of exchanging all 3rd year courses with 4th year courses and postponing some 2nd year courses..

List of Electives

We list some of the elective courses here. It is proposed that all PG courses should be available to UG students as electives. Any course from any other department can be taken as an elective subject to the approval of the faculty advisor.

CS 451 Distributed Systems
CS 407 Digital Signal Processing
CS 462 Analytical Models of Computing Systems
CS 467 Functional and Logic Programming
CS 449 Topics in Artificial Intelligence Programming
CS 336 Computer Aided Geometric Design
CS 475 Computer Graphics
CS 415 Numerical Computation
CS 444 Database Management Systems
CS 468 Computational Models in Pattern Recognition and Learning
CS 460 Natural Language Processing
CS 470 Modeling and Simulation
CS 346 Software Engineering
CS 352 Machine Learning
CS 406 Cryptography and Network Security
CS 414 Introduction to Wireless Networks
CS 329 Principles of Programming Languages
CS 435 Linear optimization
R & D Project I
R & D Project II

3. Requirements of the Minor Programme

All CSE courses (including the DIC) are open to non-CSE students subject to pre-requisites being satisfied. If and when such students earn 30 credits through CSE courses, they will qualify for a minor in CSE. Since Data Structures is a pre-requisite for almost all CSE courses, the department will make two offerings of the course every year: A regular offering for CSE students in third semester and a separate offering for minor students. A small number of minor students may be allowed in the regular offering. CSE students who fail the course in regular offering will be allowed to clear the course by registering for the offering meant for minor students.

A minor student can do up to one R&D project towards minor programme.

- Intake of minor students will be restricted to 30 for the courses with combined offering for CSE and minors. With the increase in intake for OBC students, this number may be extended to 50.

- The number of students who might be interested in CSE minor is likely to be very high. Restricting the number requires defining a clear non-subjective criterion for selecting students. The default criterion is overall CPI.

- All minor students are expected to register for co-requisites (i.e. Associated lab) also. This is particularly important where the courses are very closely related. However if there are time-table conflicts, a minor student may be exempted from the lab course provided the student performs the experiments necessary for understanding the theory course as may be advised by the instructor.

4. Requirements for B.Tech. (Honours)

A student should earn 30 additional credits over the minimum B.Tech. requirements to be eligible for the B.Tech. (Honours) degree. Of these, 12 credits have to be earned through elective CSE courses. The remaining 18 credits can be earned in any of the following ways:

- CSE Elective courses .

- A 6 credit B.Tech project I

- A 12 credit B.Tech Project II.
  - B.Tech. Project II will be available to a student only if the student gets a minimum BB grade in B.Tech. Project I.
  - B.Tech. Project II must be a continuation of B.Tech. Project I under the supervision of the same faculty.

5. Requirements of the DD Programme

A dual degree student is required to earn the following additional credits beyond the requirements of the B.Tech. Degree

- 54 credits through CSE elective courses, of which 24 credits must be through graduate-level courses
- A two- stage DD project of 72 credits

In the entire DD programme, a student can do at most 3 R & D projects as electives.

CSE Department UG Course Contents

I	Title of the Course	**CS 101- Introduction to Computer Programming**			
Ii	Credit Structure	L 2	T 0	P 2	C 6
Iii	Prerequisite, if any (for the student)	Nil			
Iv	Course content (separate sheet may be used, if necessary)	An introduction to problem solving with computers using a modern language such as Java or C/C++. Introduction to Unix environment and tools Programming features: Machine representation, primitive types, arrays and records, objects, expressions, control statements, iteration, procedures, functions, and basic I/O, recursion, pointers. Sample problems in engineering, science, text processing, and numerical methods. Two hours of laboratory time which will include practice on computers.			
V	Texts/References	1. Kernighan and Ritchie, The C programming language (2nd edition). Prentice Hall of India, 1988. 2. Coohoon and Davidson, C++ Program Design: An introduction to Programming and Object-Oriented Design. Tata McGraw Hill 3rd edition. 2003. 3. G. Dromey, How to Solve it by Computer, Prentice-Hall Inc., Upper Saddle River, NJ, 1982. 4. Yashwant Kanetkar, Let's C, Allied Publishers, 1998.			

I	Title of the Course	**CS 152 Abstractions and Paradigms in Programming**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	CS 101			
Iv	Course content (separate sheet may be used, if necessary)	Importance of abstraction in programming. Abstractions supported by the major programming paradigms functional, imperative and object-oriented: Expressions, data and control abstractions,recursion, higher order functions, state and assignment, classes, objects, encapsulation and inheritance. Inductive reasoning of functional programs, loop invariants. Abstraction and its impact on efficiency. The course should be centered around programming examples and applications that demonstrate the importance of the abstractions mentioned.			
V	Texts/References	1. Harold Abelson, Gerald Jay Sussman and Julie Sussman, Structure and Interpretation of Computer Programs, 2nd edition, The MIT Press, 1996.			

I	Title of the Course	**CS 154 Abstractions and Paradigms in Programming Lab.**			
Ii	Credit Structure	L 0	T 0	P 3	C 3
Iii	Prerequisite, if any (for the student)	Nil			
Iv	Course content (separate sheet may be used, if necessary)	Experiments to support the associated theory course.			
V	Texts/References				

I	Title of the Course	**CS 207 Discrete Structures**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	Nil			
Iv	Course content (separate sheet may be used, if necessary)	Propositions and predicates, proofs and proof techniques. Sets, relations and functions, cardinality, basic counting, recurrence relations, discrete probability. Graphs: paths, cycles, trees, connectivity.			
V	Texts/References	1. Kenneth Rosen, Discrete Mathematics and its applications, 6th edition, Tata-McGraw Hill, 2006.			

I	Title of the Course	**CS 213 Data Structures and Algorithms**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	CS 101			
Iv	Course content (separate sheet may be used, if necessary)	Introduction to data structures, abstract data types, analysis of algorithms. Creation and manipulation of data structures: arrays, lists, stacks, queues, trees, heaps, hash tables, balanced trees, tries, graphs. Algorithms for sorting and searching, order statistics, depth-first and breadth-first search, shortest paths and minimum spanning tree.			
V	Texts/References	1. T. Cormen, C. Leiserson, R. Rivest, C. Stein, Introduction to Algorithms, 2nd edition, Prentice-Hall India, 2001. 2. S. Sahni, Data Structures, Algorithms and Applications in C++, 2nd edition, Universities Press, 2005.			

I	Title of the Course	**CS 293 Data Structures and Algorithms Lab.**			
Ii	Credit Structure	L 0	T 0	P 3	C 3
Iii	Prerequisite, if any (for the student)	CS 101			
Iv	Course content (separate sheet may be used, if necessary)	Experiments based on creating and manipulating various data structures.			
V	Texts/References				

I	Title of the Course	**(New Course) Automata Theory and Logic**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	Discrete Structures (CS 207)			
Iv	Course content (separate sheet may be used, if necessary)	Propositional logic: Review and SAT solving, some puzzle solving Predicate Logic: Syntax, semantics, quantifier equivalences, notion of undecidability of predicate logic. Rudiments of formal languages. Finite state machines (DFA/NFA/epsilon NFAs), regular expressions. Properties of regular languages. My hill-Nerode Theorem. Non-regularity. Push down automata. Properties of context-free languages. Turing machines:Turing hypothesis, Turing computability, Nondeterministic, multi tape and other versions of Turing machines. Church's thesis, recursively enumerable sets and Turing computability. Universal Turing machines. Unsolvability, The halting problem, partial solvability, Turing enumerability, acceptability and decidability, unsolvable problems about Turing Machines. Post`s correspondence problem.			
V	Texts/References	1. Introduction to Automata Theory, Languages and Computation, by John. E. Hopcroft, Rajeev Motwani, J. D. Ullman, published by Pearson Education Asia, 2006. 2. Elements of the Theory of Computation, by H.R. Lewis and C.H.Papadimitrou, published by Prentice Hall Inc, 1981. 3. Logic in Computer Science. Huth and Ryan. Cambridge University Press, 2004.			

I	Title of the Course	**CS 301 Design and Analysis of Algorithms**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	Data Structures Algorithms			
Iv	Course content (separate sheet may be used, if necessary)	Models of computation, algorithm analysis, time and space complexity, average and worst case analysis, lower bounds. Algorithm design techniques: divide and conquer, greedy, dynamic programming, amortization, randomization. Problem classes P, NP, PSPACE; reducibility, NP-hard and NP-complete problems. Approximation algorithms for some NP-hard problems.			
V	Texts/References	1. T.H.Cormen, C.E.Leiserson, R.L.Rivest, C. Stein, Introduction to Algorithms, 2nd edition, Prentice-Hall India, 2001. 2. J. Kleinberg and E. Tardos, Algorithm Design, Pearson International Edition, 2005.			

I	Title of the Course	**(New Course) Logic Design**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	EE 101			
Iv	Course content (separate sheet may be used, if necessary)	Switching theory: Introduction to number systems, Computer arithmetic, switching function and logic circuits Combinational Logic, Canonical Logic Forms, K-maps Standard logic (SSI, MSI) vs. programmable logic (PLD, PGA). Finite state machine design: logic, minimization and races. Arithmetic unit, Control unit design, Logic design applications in computer systems, Introduction to computer-aided design software, FPGA overview, Introduction to design automation and design through Higher level languages like VHDL.			
V	Texts/References	1. Digital Systems Design with VHDL , Mark Zwonlinski, Pearson Education, 2003. 2. Contemporary Logic Design, Randy H. Katz and Gaetano Borriello, Prentice Hall 2nd edition, 2004. 3. SH Unger, The essence of logic circuits, Englewood Cliffs, NJ, Prentice Hall, 1989 4. Foundations of digital logic design , World Scientific Singapore, 1998			

I	Title of the Course	**(New Course) Logic Design Lab.**			
Ii	Credit Structure	L 0	T 0	P 3	C 3
Iii	Prerequisite, if any (for the student)	EE 101			
Iv	Course content (separate sheet may be used, if necessary)	Experiments with Logic Building Blocks using SSI/MSI, Experiments on Design and/or use Minimization tools. Use of VHDL and simulation in Logic Design. A small project on design with the use of tools and MSI and/or PLDs. FPGA basics and programming.			
V	Texts/References				

I	Title of the Course	**CS 296 Software Systems Lab.**			
Ii	credit structure	L 2	T 0	P 2	C 6
Iii	Prerequisite, if any (for the student)	Data Structures and Algorithms Lab.			
Iv	Course content (separate sheet may be used, if necessary)	Introduction to the UNIX operating system (file system and directory structure,and processes). Unix tools (shell programming, grep, tar, compress, sed, find, sort etc). Programming in AWK. Introduction to World Wide Web (HTML, HTTP, CGI).Programming Using Java, Graphical User Interface Programming using Java. Socket programming in Java. Programming tools (make, source code control using RCS/CVS/SVN, debuggers). Document processing using Latex.			
V	Texts/References				

<table>
<tr><td>I</td><td>Title of the Course</td><td colspan="4">(New Course) Computer Architecture</td></tr>
<tr><td>Ii</td><td>Credit Structure</td><td>L
3</td><td>T
0</td><td>P
0</td><td>C
6</td></tr>
<tr><td>Iii</td><td>Prerequisite, if any (for the student)</td><td colspan="4">Logic Design</td></tr>
<tr><td>Iv</td><td>Course content (separate sheet may be used, if necessary)</td><td colspan="4">Assembly level organization: instruction formats, addressing mechanisms, Architecture and programming of 8085 and or x86 architectures, microprogramming, Arithmetic and Logic Unit, Memory systems: memory hierarchy, main memories, cache, virtual memory, Pipeline processing, Interfacing and communication: I/O, interrupts, buses. Multiprocessor and alternative architectures, Contemporary architectures

Computer organization and architecture Lab

Machine/Assembly programming, Design of basic computing units.</td></tr>
<tr><td>V</td><td>Texts/References</td><td colspan="4">1. Computer Architecture, Fourth Edition: A Quantitative Approach by John L. Hennessey, David A. Patterson, Morgan Koffman. 2006.
2. Computer Organization and Architecture, William Stallings, Prentice Hall, 7th edition 2006
3. Computer Architecture and Organizationby John P. Hayes, McGraw-Hill 3rd edition, 2002</td></tr>
</table>

<table>
<tr><td>I</td><td>Title of the Course</td><td colspan="4">(New Course) Computer Architecture Lab</td></tr>
<tr><td>Ii</td><td>Credit Structure</td><td>L
0</td><td>T
0</td><td>P
3</td><td>C
3</td></tr>
<tr><td>Iii</td><td>Prerequisite, if any (for the student)</td><td colspan="4"></td></tr>
<tr><td>Iv</td><td>Course content (separate sheet may be used, if necessary)</td><td colspan="4">Experiments to support the associated theory course. In particular, the experiments should involve the use of architectural simulation tools and interfacing peripherals.</td></tr>
<tr><td>V</td><td>Texts/References</td><td colspan="4"></td></tr>
</table>

<table>
<tr><td>I</td><td>Title of the Course</td><td colspan="4">CS 347 Operating Systems</td></tr>
<tr><td>Ii</td><td>Credit Structure</td><td>L
3</td><td>T
0</td><td>P
0</td><td>C
6</td></tr>
<tr><td>Iii</td><td>Prerequisite, if any (for the student)</td><td colspan="4">Data Structures and Algorithms</td></tr>
<tr><td>Iv</td><td>Course content (separate sheet may be used, if necessary)</td><td colspan="4">Fundamental goals of operating systems Overview of important features of computer architectures for OS operation. Issues in user service and system performance.
Overview of operating systems--multiprogramming, time sharing, real time and distributed operating systems. Concurrency and parallelism.
Processes and threads Process synchronization. Process deadlocks.
Memory management. Memory fragmentation and techniques for memory reuse.
Virtual memory using paging. Segmentation.
File systems. Implementation of file Operations .Protection of files.
Case studies of contemporary operating systems.</td></tr>
<tr><td>V</td><td>Texts/References</td><td colspan="4">1. Dhamdhere, D. M., Operating Systems--A concept-based approach, Second edition, McGraw-Hill Education India, New Delhi, 2006.
2. Silberschatz, A., P. B. Galvin, and G. Gagne, Operating System Principles, Seventh edition, John Wiley, New York, 2005.
3. Stallings, W., Operating Systems--Internals and Design Principles,Fifth edition, Pearson Education, New York, 2005.</td></tr>
</table>

<table>
<tr><td>I</td><td>Title of the Course</td><td colspan="4">CS 377 Operating Systems Lab</td></tr>
<tr><td>Ii</td><td>Credit Structure</td><td>L
0</td><td>T
0</td><td>P
3</td><td>C
3</td></tr>
<tr><td>Iii</td><td>Prerequisite, if any (for the student)</td><td colspan="4"></td></tr>
<tr><td>Iv</td><td>Course content (separate sheet may be used, if necessary)</td><td colspan="4">Experiments to support the associated theory course.</td></tr>
<tr><td>V</td><td>Texts/References</td><td colspan="4"></td></tr>
</table>

I	Title of the Course	**CS 317 Database and Information Systems**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	Data Structures Algorithms			
Iv	Course content (separate sheet may be used, if necessary)	Nature of Business Systems and Data Processing. Data Models, ER Model, ER Diagrams, UML Class Diagrams. Relational model and query languages (relational algebra and calculus, SQL). Integrity and Security. Database design and normalization. XML and x query. Storage structures. Indexing and Hashing Techniques. Query processing and optimization, transactions, concurrency control and recovery. Introduction to decision support and data analysis, data warehousing and data mining. Information Retrieval.			
V	Texts/References	1. Abraham Silberschatz, Henry F. Korth and S. Sudarshan, Database System Concepts 4th Ed, McGraw Hill, 2002. 2. Raghu Ramakrishnan and Johannes Gehrke, Database Management Systems, 3rd Ed, 2002 3. Ramez Elmasri and Shamkant Navathe, Fundamentals of Database Systems 3rd Ed, Benjamin Cummings, 1999.			

I	Title of the Course	**CS 387 Database Information Systems Lab**			
Ii	Credit Structure	L 0	T 0	P 3	C 3
Iii	Prerequisite, if any (for the student)	Data Structures and Algorithms			
Iv	Course content (separate sheet may be used, if necessary)	Use of database systems supporting interactive SQL. Two-tier client-server applications using JDBC or ODBC, Three-tier web applications using Java servlets/JDBC or equivalent. Design of applications and user interfaces using these systems. Data analysis tools. Laboratory project.			
V	Texts/References				

I	Title of the Course	**CS 344 Artificial Intelligence**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	Data Structures and Algorithms			
Iv	Course content (separate sheet may be used, if necessary)	Basics of problem-solving : problem representation paradigms, state space, satisfiability vs optimality, pattern classification problems, example domains. Search Techniques : Problem size, complexity, approximation and search; depth, breadth and best search; knowledge based problem solving, artificial neural networks. Knowledge representation : First order and non-monotonic logic; rule based, frame and semanticnetwork approaches. Knowledge Acquisition : Learnability theory, approaches to learning. Uncertainty Treatment : formal and empirical approaches including Bayesian theory, belief functions, certainty factors, and fuzzy sets. Detailed Discussion from Example Domains : Industry, Language, Medicine, Verification, Vision, Knowledge Based Systems. Languages and Machines : AI languages and systems, special purpose architectures.			
V	Texts/References	1. George F.Luger and William A. Stubblefield, AI: Strcutures and Strategies for Complex problem solving, 2nd edition, Benjamin Cummings Publishers, 1997. 2. Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall Series in AI, 1995. 3. Mark Stefik, Introduction to Knowledge Systems, Morgan Kaufman, 1995. 4. Winston P.H., Artificial Intelligence, 3rd edition, Addison Wesley, 1995. 5. E. Rich and K.Knight, Artificial Intelligence, Tata McGraw Hill, 1992. 6. E. Charniack and D. McDermott, Artificial Intelligence, Addison Wesley, 1987. 7. N.J.Nilsson, Principles of Artificial Intelligence, Morgan Kaufman, 1985.			

I	Title of the Course	**CS 386 Artificial Intelligence Lab**			
Ii	Credit Structure	L 0	T 0	P 3	C 3
Iii	Prerequisite, if any (for the student)				
Iv	Course content (separate sheet may be used, if necessary)	The laboratory will emphasize the use of PROLOG, LISP, CLOS (Common Lisp Object Systems), Expert System Shells, tools from public domain, and in-house work.			
V	Texts/References				

I	Title of the Course	**(New Course) Implementation of Programming Languages**			
Ii	Credit Structure	L 3	T 1	P 0	C 8
Iii	Prerequisite, if any (for the student)	Abstraction and Paradigms in Programming, Data structures and Algorithms, Automata Theory and Logic			
Iv	Course content (separate sheet may be used, if necessary)	The compiled and interpreted execution models. Lexical analysis and parsing using lex and yacc. Scope and visibility analysis. The role of types. Type analysis of a language with basic types,derived types, parametric polymorphism and subtypes. Binding times. Data layout and lifetime management of data. Stack and heap as storage structures. Implementation of function calls. Activation records structures. Dynamic memory allocation and Garbage collection. Implementation of higher order functions - closures. Implementation of control structures, exception handling. Implementation of object oriented concepts -- objects, inheritance and dynamic dispatch. Implementation of a naive code generator for a virtual machine. Security checking of virtual machine code.			
V	Texts/References	1. Alfred V. Aho, Monica S. Lam, Ravi Sethi and Jeffrey D.Ullman: Compilers: Principles, Techniques, and Tools, 2/E, Addison-Wesley 2007. 2. Andrew Appel : Modern Compiler Implementation in C/ML/Java, Cambridge University Press, 2004 3. Dick Grune, Henri E. Bal, Cerial J.H. Jacobs and Koen G. Langendoen: Modern Compiler Design, John Wiley & Sons, Inc. 2000. 4. Michael L. Scott: Programming Language Pragmatics, Morgan Kaufman Publishers, 2006.			

I	Title of the Course	**Implementation of Programming Languages Lab**			
Ii	Credit Structure	L 0	T 0	P 3	C 3
Iii	Prerequisite, if any (for the student)				
Iv	Course content (separate sheet may be used, if necessary)	Design and implementation of a compiler for a sufficiently rich subset of a real programming language. The compiler will be automatically generated through use of tools such as LEX , YACC and IBURG.			
V	Texts/References	1. Levine, J.R., T. Mason and D. Brown, Lex and Yacc, edition, O'Reilly & Associates, 1990			

I	Title of the Course	**CS 348 Computer Networks**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	Data Structures and Algorithms			
Iv	Course content (separate sheet may be used, if necessary)	Design of Computer Networking protocols at all layers: transmission media, data link protocols, media access control, routing and congestion control, admission control, traffic shaping and policing, Internet working (IP) and transport layer protocols (TCP). Performance analysis of networks.			
V	Texts/References	1. W. Stallings, Data and Computer Communications, 6th edition, Prentice Hall, 2000. 2. A. S. Tannenbaum, Computer Networks, 4th edition, Prentice Hall, 2003. 3. F. Halsall, Data Communications, Computer Networks and Open Systems, 4th edition, Addison-Wesley, 1996. 4. Walrand and Varaiya, High Performance Communication Networks, Morgan Kaufman, 1996. 5. D. E. Comer, Internet working with TCP/IP: Principles, Protocols, Architecture, 3rd edition, Prentice Hall, 2000. 6. W. R. Stevens, TCP/IP Illustrated Vol. I, Addison Wesley, 1994.			

I	Title of the Course	**CS 378 Computer Networks Lab**			
Ii	Credit Structure	L 0	T 0	P 3	C 3
Iii	Prerequisite, if any (for the student)				
Iv	Course content (separate sheet may be used, if necessary)	Experiments to support study of the Internet protocol stack: a) Experimental study of application protocols such as HTTP, FTP, SMTP, using network packet sniffers and analyzers such as Ethereal. Small exercises in socket programming in C/C++/Java. b) Experiments with packet sniffers to study the TCP protocol. Using OS (netstat, etc) tools to understand TCP protocol FSM, retransmission timer behavior, congestion control behaviour. c) Introduction to ns2 (network simulator) - small simulation exercises to study TCP behavior under different scenarios. d) Setting up a small IP network - configure interfaces, IP addresses and routing protocols to set up a small IP network. Study dynamic behaviour using packet sniffers e) Experiments with ns2 to study behaviour (especially performance of) link layer protocols such as Ethernet and 802.11 wireless LAN.			
V	Texts/References				

I	Title of the Course	**CS 336 Computer Aided Geometric Design**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)				
Iv	Course content (separate sheet may be used, if necessary)	Introduction/motivation for CAGD. Explicit and Implicit definitions. Problems of surface and curve design. Interpolation and Approximation. Polynomial bases such as Bezier and B-Spline. Design using polynomial bases. Geometry of Curves and Surfaces. User requirements such as convexity, smoothness and fairness. The Solid-Modeler paradigm. Definition and Operations on solids. The manufacturing cycle. The ACIS C++ CAD kernel library. The scheme language test-harness for the ACIS kernel. Manipulation of solids and elementary application programs on this kernel.			
V	Texts/References	1. Gerald Farin, Curves and Surfaces in CAGD, 3rd edition, Academic Press, 1993. 2. D.Rogers and J A Adams, Mathematical Elements of Computer Graphics, 2nd edition, McGraw-Hill, 1990.			

I	Title of the Course	**CS 346 Software Engineering**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	Nil			
Iv	Course content (separate sheet may be used, if necessary)	Software engineering backdrop; Software development orientation, software development practices, processes, and architecture , software development and life cycle, software project management, estimation techniques, quality management systems,quality control:reviews, configuration management, software requirements phase, process modelling, data modeling, time frame modeling, software design phase, user interface design, computer aided software engineering, software construction phase, quality control: testing.			
V	Texts/References	1. Software Engineering, S.A.Kelkar, Prentice Hall of India, 2007 2. Software Engineering, 6th Edition, Pressman, Tata McGraw Hill, 2006 3. Software Engineering, 3rd Edition, Pankaj Jalote, Narosa Publishers, 2006.			

I	Title of the Course	**CS 352 Machine Learning**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	Nil			
Iv	Course content (separate sheet may be used, if necessary)	Introductory Material: Machine Learning and AI, Motivations for Studying ML, Supervised and Unsupervised learning, Machine Learning in the Large Classical ML Topics: Concept Learning (also called Learning from Examples), Learning from Analogy, Explanation Based Learning, Structure Learning, Reinforcement Learning, Decision Tree Learning, Decision List Learning Theoretical ML: Identification in the Limit, Oracle Based Learning, Probably Approximately Correct (PAC) Model, Boosting Bayesian Learning: Maximum Likelihood Estimates, Parameter Estimation, Bayesian Belief Networks Introductory Graphical Models Based Learning: Expectation Maximization as a fundamental technique, Hidden Markov Models (HMM): Motivation for Generative Models, Forward-backward Algorithm, Baum Welch Iteration, Feature Enhanced HMM Maximum Entropy Markov Models (MEMM): Motivation for Discriminative Models, Training of MEMMs Introductory Optimization Based Methods: Neural Nets, Support Vector Machines, Genetic Algorithms Applications: Text Learning, Speech Processing, Data Mining, Bioinformatics			
V	Texts/References	1. Christopher M. Bishop, Pattern Recognition and Machine Learning,Springer, 2006 2. Tom Mitchell, Machine Learning, McGraw Hill, 1997 (new chapters on line, 2006) 3. Duda, Hart and Stork, Pattern Classification (2nd ed.), Wiley Interscience, 2000			

I	Title of the Course	**CS 406 Cryptography and Network Security**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	Computer Networks (CS 348)			
Iv	Course content (separate sheet may be used, if necessary)	Threats, vulnerabilities and attacks. Authentication, confidentiality, integrity and non-repudiation in data communication. Stream ciphers (ex. RC4) and block ciphers (DES, AES), modes of operation. Public key cryptography using RSA and elliptic curve cryto-schemes (on prime and binary fields). Linear and differential cryptanalysis, side-channel attacks. Discrete Logarithm Problem and Duffie-Hellman key exchange. Attacks (based on Pollard-rho algorithm, for example) on factorization and the discrete log. Properties, applications and attacks on the cryptographic hash. Message authentication Code (MAC) and the digital signature. Key management, digital certificates and the Public Key Infrastructure (PKI), hardware/software implementation issues for various crypto schemes including performance. Basic authentication protocols, Needham-Schroeder protocol and Kerberos. Security at the network layer (IPSec), transport layer (SSL/TLS) and in applications such as electronic payment. Introduction to network security (firewalls, worms/viruses, Trojans and spy ware, intrusion detection/prevention systems, virtual private networks).			
V	Texts/References	1. Cryptography and Network Security - Principles and Practices, W. Stallings, Pearson Education Publishers, (3rd Edition), 2005. 2. Network Security - Private Communication in a Public World, Kaufman, Perlman, Speciner, Prentice Hall, (2nd Edition), 2002. 3. Cryptography and Network Security B.Forouzan. Tata McGraw Hill.(1st Edition), 2007. 4. Several papers from conferences, magazines and journals (principally IEEE and ACM).			

I	Title of the Course	**CS 407 Digital Signal Processing**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	Nil			
Iv	Course content (separate sheet may be used, if necessary)	Signals as sequences. Linear time invariant operators. The impulse response. The discrete fourier transform. The z-transform. Aliasing in frequency and periodicity in time. Band-limited-ness and the sampling theorem. Filters, FIR and IIR. Various properties of the transforms and their use in filter design. The fast fourier transform and its uses. Rudiments of Estimation. Applied topics.			
V	Texts/References	1. R. Oppenheim and S. Schafer, Digital Signal Processing, Prentice-Hall India, 1975.			

I	Title of the Course	**CS 414 Introduction to Wireless Networks**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	Computer Networks (CS 348)			
Iv	Course content (separate sheet may be used, if necessary)	This course examines common and different aspects of wired and wireless networks. The topics covered are: antenna basics, radio propagation, coding and error control, MAC protocols, network layer protocols to address mobility, TCP and wireless, wireless LANs and ad-hoc networks, cellular communication concepts, wireless mesh networks, long-distance and last-hop wireless technologies, and security in wireless systems.			
V	Texts/References	1. Principles of Wireless Networks, K. Pahlavan and P. Krishnamurthy, Pearson Education, 2002. 2. Wireless Communication and Networks, W. Stallings, Pearson Education, 2002. 3. Mobile Communications, Jochen Schiller, Addison Wesley, 2003.			

I	Title of the Course	**CS 415 Numerical Computation**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	Nil			
Iv	Course content (separate sheet may be used, if necessary)	Representation of Numbers. Sources of errors and their propagation. Error analysis and the idea of conditioning. Linear systems of equations and their solutions. Gauss elimination and its complexity and robustness. Well-conditioning and matrix inversion. Gram-Schmitt orthogonalization. Interpolation and approximation. Divided differences. Interpolation at increasing number of points. Best approximation and orthogonal polynomials. Iterative methods for root finding. Rates of convergence. Fixed points. Iterative methods for linear systems. Numerical Differentiation. Ordinary Differential Equations and numerical Integration.			
V	Texts/References	1. P. Davis, Interpolation and Approximation, Dover, 1975. 2. S.D.Conte and C. deBoor, Elementary Numerical Analysis 3ed, McGraw Hill, 1981.			

I	Title of the Course	**CS 444 Database Management Systems**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)				
Iv	Course content (separate sheet may be used, if necessary)	Architecture of a typical DBMS. Advanced storage devices including RAID. Advanced index structures. Buffer Management. Implementation of basic and extended relational algebra operations. Cost estimation. Cost based query optimization. Transaction processing including ACID properties and serializability, concurrency control techniques and recovery techniques. Database tuning. Introduction to selected advanced topics.			
V	Texts/References	1. Abraham Silberschatz, Henry F. Korth and S. Sudarshan, Database System Concepts 4th Ed, McGraw Hill, 2002. 2. Ramez Elmasri and Shamkant Navathe, Fundamentals of Database Systems 2nd Ed, Benjamin Cummings, 1994.			

I	Title of the Course	**CS 449 Topics in Artificial Intelligence Programming**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)				
Iv	Course content (separate sheet may be used, if necessary)	Pattern recognition and analysis, supervised and unsupervised learning, learning in neural nets, associative memories. Natural Language Processing: Stages of processing, parsing Techniques. GPSG, LFG and such other Chomsky approaches. Case Grammar based methods, Morphological processing, Lexicon building and Organization, knowledge representation for NLP. Intelligent Systems in Management: strategic-operational problems, effectiveness measures, heuristics in planning and scheduling problems, learning and process redesign. Information Management: intelligent retrieval on the web, semantic distance measures. Intelligent Systems in Medicine: diagnostic and prognostic approaches, case based learning and system evolution. Rewrite Systems, Automated Theorem Proving, Higher order and Non-monotonic Logic.			
V	Texts/References	1. Carl G. Looney, Pattern Recognition Using Neural Networks, Oxford University Press, 1997. 2. Gazder M., Natural Language Processing, MIT Press, 1994. 3. Handbook of Logic in AI (vol. 1), Oxford University Press, 1997.			

I	Title of the Course	**CS 451 Distributed Systems**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)	Nil			
Iv	Course content (separate sheet may be used, if necessary)	Introduction to Distributed Computing: absence of global states;causal ordering of events. Distributed architectures: shared memory and message passing, Programming Models such as PVM; MPI; Linda; ORCA, Distributed algorithms: mutual exclusion, consensus, leader election. Clock synchronization, distributed termination. Fault Tolerance:fail-stop and byzantine models.			
V	Texts/References	1. H. Attiya and J. Welch, Distributed Computing: Fundamentals, Simulations, and Advanced Topics, McGraw-Hill, 1998. 2. G. F. Colouris, and J. Dollimore, Distributed Systems: Concepts and design, Addison Wesley, 1988. 3. N. Lynch, Distributed Algorithms, Morgan Kaufman, 1996. 4. S. Mullender (Ed.), Distributed Systems, 2nd Edition, Addison Wesley, 1993. 5. Tel Gerard, Introduction to distributed algorithms Cambridge University Press, Cambridge, 1994. 6. Michel Raynal, Distributed algorithms and protocols Wiley, Chichester, 1988. 7. Valmir C. Barbosa, An Introduction to Distributed Algorithms, MIT Press, 1996.			

I	Title of the Course	**CS 460 Natural Language Processing**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)				
Iv	Course content (separate sheet may be used, if necessary)	Introductory Material: Motivation for studying NLP; Natural Language Processing as the forcing function of AI; Classical approaches to NLP with knowledge bases and linguistic rules; Data Driven and Machine Learning Approaches to NLP; Efficient, Robust and Scalable NLP Classical NLP: Linguistics Fundamentals: Syntax and Parsing: Meaning: Empirical or Statistical NLP: Probabilistic Methods on Introductory Graphical Models for NLP: Shallow Parsing: Probabilistic Parsing Applications: Machine Translation, Information Retrieval, Question Answering, Summarization, Information Extraction Biology and Sociology of NLP: Neurolinguistics, Child Language Acquisition			
V	Texts/References	1. Jurafsky, Daniel, and James H. Martin, Speech and Language Processing:An Introduction to Natural Language Processing, Speech Recognition,and Computational Linguistics, Prentice-Hall, 2000. 2. Christopher D. Manning and Hinrich Schütze, Foundations of Statistical Natural Language Processing. Cambridge, MIT Press, 1999. 3. James Allen, Natural Language Understanding, Benjamin/Cummings, 2ed, 1995. 4. Eugene Charniak, Statistical Language Learning, MIT Press, 1996. 5. Martin Atkinson, David Britain, Harald Clahsen, Andrew Redford, Linguistics, Cambridge University Press, 1999. 6. P. Lieberman, Toward an evolutionarybiology of language, Harvard university Press, 2006.			

I	Title of the Course	**CS 462 Analytical Models of Computing Systems**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)				
Iv	Course content (separate sheet may be used, if necessary)	Queuing models of scheduling in batch and time sharing systems, Priority scheduling. Queuing models of a disc system. Network of Queues. Queuing models for multiprogrammed systems.			
V	Texts/References	1. E.G.Coffman, P.J.Denning, Operating Systems Theory, Prentice Hall, 1983. 2. P.B.Hansen, Operating SystemPrinciples, Prentice Hall, 1973. 3. L.Kleinrock, Queuing Systems, Vol.I and II, Wiley, 1976.			

I	Title of the Course	**CS 467 Functional and Logic Programming**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)				
Iv	Course content (separate sheet may be used, if necessary)	Functional Programming :Introduction, Lambda Calculus, translating high level functional language into the lambda calculus, structured types, semantics of pattern matching and efficient compilation, list comprehension, polymorphic type checking, graph reduction of lambda expressions, lazy evaluation, Super combinators, SK combinators, G-code, strictness analysis, SASL, Examples of functional languages - ML, Haskell. Logic Programming : Logic and Reasoning, Logic programs, Prolog syntax and its principal primitives. Some important techniques: tail recursion, accumulators, difference lists. Some applications such as simple theorem proving, Natural Language Processing, Expert Systems. Implementation of logic programs. Constraint Logic Programming: constraint satisfaction, constraint propagation- rationale, methodology and examples.			
V	Texts/References	1. Simon Thompson, Haskell: The craft of Functional Programming, 2ndEdition, Addison-Wesley, 1999. 2. Simon L. Peyton Jones, The Implementation of Functional Programming Languages, Prentice Hall, International series in Computer Science, 1987. 3. H. Barendregt. The Lambda Calculus: Its Syntax and Semantics. North Holland, 1984. 4. Christopher John Hogger, Introduction to Logic Programming, Academic Press, 1984. 5. L. Stirling and E. Shapiro, The Art of Prolog :Advanced Programming Techniques, MIT Press, 2nd edition, 1994; 6. W.F. Clocksin and C.S. Mellish, Programming in Prolog, Springer-Verlag, 1987. 7. JW Lloyd. Foundations of Logic Programming. Springer Verlag, 1987			

I	Title of the Course	**CS 468 Computational Models in Pattern Recognition and Learning**			
Ii	Credit Structure	L 3	T 0	P 0	C 6
Iii	Prerequisite, if any (for the student)				
Iv	Course content (separate sheet may be used, if necessary)	Introduction to Learning, different approaches to machine learning. Learning Algorithms, Complexity of inductive inference. Review of formal languages, finite automata and regular languages, context-free languages and parsing. Language identification in the limit. Gold's basic results. Polynomial learning, PAC learnability, Valiant's results. VC-dimension. Examples from language identification. Sample Complexity for finite hypothesis spaces, Learnability of CNF formulas. Sample Complexity for infinite hypothesis spaces, VC dimension for neural networks. Mistake Bound Model of Learning.			
V	Texts/References	1. Mitchell T.M., Machine Learning, Mc-Graw Hill International, 1984. 2. Anthony M. and Biggs N., Computational Learning Theory : An Introduction, Cambridge University Press, England, 1992. 3. Natarajan B.K., Machine Learning : A Theoretical Approach, Morgan Kaufman, 1991. 4. Kearns M.J. and Vazirani U.V., An Introduction to Computational Learning Theory, Cambridge, Ma., MIT Press, 1994.			

<table>
<tr><td>I</td><td>Title of the Course</td><td colspan="4">CS 470 Modelling and Simulation</td></tr>
<tr><td>Ii</td><td>Credit Structure</td><td>L
3</td><td>T
0</td><td>P
0</td><td>C
6</td></tr>
<tr><td>Iii</td><td>Prerequisite, if any (for the student)</td><td colspan="4"></td></tr>
<tr><td>Iv</td><td>Course content (separate sheet may be used, if necessary)</td><td colspan="4">Selected illustrative examples of simulation applications.
Models: Structural, Process, Continuous, Discrete, Deterministic, Random, input/output, static, dynamic, multilevel.
Simulation: Analog/Digital/Hybrid, verification, validation.
Data Modelling and Analysis : Population parameters, hypotheses testing, confidence intervals, goodness of fit, estimating transient/steady-state characteristics, variance reduction.

Simulation Process : Problem formulating, model building, data acquisition, model translation, verification, validation, strategic and tactical planning, experimentation, analysis of results, implementation and documentation.
Simulation Languages: Examples from SIMSCRIPT, GPSS, GASP, SIMULA, etc.</td></tr>
<tr><td>V</td><td>Texts/References</td><td colspan="4">1. G.Gordon, System Simulation, 2nd ed., Prentice Hall, 1978.
2. Narsing Deo, System Simulation with Digital Computers, Prentice Hall, 1976.
3. J.R. Leigh, Modelling and Simulation, Peter Peregrims Ltd., 1983.
4. A.M.Law, W.D.Kelton, Simulation Modelling and Analysis, McGraw Hill, 1982.</td></tr>
</table>

I	Title of the Course	**CS 329 Principles of Programming Languages**			
Ii	Credit Structure	L	T	P	C
Iii	Prerequisite, if any (for the student)	None			
Iv	Course content (separate sheet may be used, if necessary)	Introduction: Brief history of evolution of programming languages. Programming abstractions for control and data. Classification of Programming Languages. Types in Programming Languages: Types and values. Primitive types and composite types, recursive types. Sub typing and presumption rules, Forms of Polymorphism. Static vs. dynamic typing, type-safety and type checking. Lambda Calculus. Foundations of Imperative, Object Oriented, Applicative and Declarative Programming Languages. Programming Environments and virtual machines, A study of some runtime structures. Threads and concurrency primitives in modern programming languages, event handling.			
V	Texts/References	1. Benjamin Pierce, Types and Programming Languages, MIT Press, 2002. 2. David Watt, Programming Language Design Concepts, Willey, 2004. 3. Kenneth Louden, Programming Languages: Principles & Practice,Thomson, 2003 4. Relevant Language Specifications and Reports			

i	Title of the Course	**CS 435 Linear Optimization**
ii	Credit structure	3 0 0 6
iii	Prerequisite, if any (for the students)	
iv	Course Content (seperate sheet may be used, if necessary)	Vector Spaces: bases, echelon forms, rank and determinants. Gauss elimination and its complexity, Inner products, Gram-Schmidt orthogonalization. Linear transformations. Optimization: Modelling and formulation of optimization problems. Linear costs and convex domains. Mean-square (distance) minimizations. Linear programming and the Simplex algorithm. Duality and the primal dual method. Examples from combinatorial optimization. Shortest paths, network flows and matchings. Approximation and randomized algorithms. Matrix Games.
v	Texts/References (seperate sheet may be used, if necessary)	1. C.Papadimitriou and K. Steiglitz,Combinatiorial Optimization, Prentice-Hall India, 1996. 2. Gibert Strang, Linear Algebra and its Applications, Harcourt Brace Jovanovitch, 1988. 3. V. Chvatal, Linear Programming and Applications, 1982. 4. K. Hoffman and R Kunze, Linear Algebra, Prentice-Hall India, 1971. 5. E.D. Nering and A.W.Tucker, Linear Programs and Related Problems, Academic Press, 1993.

i	Title of the Course	**CS 475 Computer Graphics**
ii	Credit structure	3 0 0 6
iii	Prerequisite, if any(for the students)	CS 213(exposure).
iv	Course Content (seperate sheet may be used, if necessary)	Introduction: What is Computer Graphics? Geometric Manipulation: Transformations, Matrices, Homogeneous Coordinates. Elementary 3D Graphics: Plane projections, Vanishing points, Specification of a 3D view. Visibility: Image and object precision, z-buffer algorithms, area based algorithms. Basic Raster Graphics: Scan conversion, filling, and clipping. Rendering: Lighting, Radiosity, Raytracing
v	Texts/References (seperate sheet may be used, if necessary)	1. F.S. Hill. Computer Graphics Using Open GL. Prentice Hall. 2001 2. S. Feiner, J. Foley, A. Van Dam, R Hughes, Computer Graphics, Principles and Practice. Addison Wesley, 1990.