

ANNA UNIVERSITY CHENNAI : : CHENNAI – 600 025

AFFILIATED INSTITUTIONS

B.E. (8 SEMESTER) COMPUTER SCIENCE AND ENGINEERING

CURRICULUM – R 2008

(Common to all branches of B.E. / B.Tech Programmes)

**SEMESTER V**

(Applicable to the students admitted from the Academic year 2008–2009 onwards)

CODE NO.	COURSE TITLE	L	T	P	C
<b>THEORY</b>					
CS2301	<a href="#">Software Engineering</a>	3	0	0	3
MA2265	<a href="#">Discrete Mathematics</a>	3	1	0	4
CS2302	<a href="#">Computer Networks</a>	3	0	0	3
CS2303	<a href="#">Theory of Computation</a>	3	1	0	4
CS2304	<a href="#">System Software</a>	3	1	0	4
CS2305	<a href="#">Programming Paradigms</a>	3	0	0	3
<b>PRACTICAL</b>					
CS2307	<a href="#">Network Lab</a>	0	0	3	2
CS2308	<a href="#">System Software Lab</a>	0	0	3	2
CS2309	<a href="#">Java Lab</a>	0	0	3	2
<b>TOTAL</b>		<b>18</b>	<b>2</b>	<b>9</b>	<b>27</b>

**UNIT I SOFTWARE PRODUCT AND PROCESS 9**

Introduction – S/W Engineering Paradigm – Verification – Validation – Life Cycle Models – System Engineering – Computer Based System – Business Process Engineering Overview – Product Engineering Overview.

**UNIT II SOFTWARE REQUIREMENTS 9**

Functional and Non-Functional – Software Document – Requirement Engineering Process – Feasibility Studies – Software Prototyping – Prototyping in the Software Process – Data – Functional and Behavioral Models – Structured Analysis and Data Dictionary.

**UNIT III ANALYSIS, DESIGN CONCEPTS AND PRINCIPLES 9**

Systems Engineering - Analysis Concepts - Design Process And Concepts – Modular Design – Design Heuristic – Architectural Design – Data Design – User Interface Design – Real Time Software Design – System Design – Real Time Executives – Data Acquisition System – Monitoring And Control System.

**UNIT IV TESTING 9**

Taxonomy Of Software Testing – Types Of S/W Test – Black Box Testing – Testing Boundary Conditions – Structural Testing – Test Coverage Criteria Based On Data Flow Mechanisms – Regression Testing – Unit Testing – Integration Testing – Validation Testing – System Testing And Debugging – Software Implementation Techniques

**UNIT V SOFTWARE PROJECT MANAGEMENT 9**

Measures And Measurements – ZIPF's Law – Software Cost Estimation – Function Point Models – COCOMO Model – Delphi Method – Scheduling – Earned Value Analysis – Error Tracking – Software Configuration Management – Program Evolution Dynamics – Software Maintenance – Project Planning – Project Scheduling– Risk Management – CASE Tools

**TOTAL= 45 PERIODS****TEXT BOOKS:**

1. Ian Sommerville, "Software engineering", Seventh Edition, Pearson Education Asia, 2007.
2. Roger S. Pressman, "Software Engineering – A practitioner's Approach", Sixth Edition, McGraw-Hill International Edition, 2005.

**REFERENCES:**

1. Watts S.Humphrey,"A Discipline for Software Engineering", Pearson Education, 2007.
2. James F.Peters and Witold Pedrycz,"Software Engineering, An Engineering Approach", Wiley-India, 2007.
3. Stephen R.Schach, " Software Engineering", Tata McGraw-Hill Publishing Company Limited, 2007.
4. S.A.Kelkar,"Software Engineering", Prentice Hall of India Pvt, 2007.

**AIM**

To extend student's Logical and Mathematical maturity and ability to deal with abstraction and to introduce most of the basic terminologies used in computer science courses and application of ideas to solve practical problems.

**OBJECTIVES**

At the end of the course, students would

- Have knowledge of the concepts needed to test the logic of a program..
- Have an understanding in identifying structures on many levels.
- Be aware of a class of functions which transform a finite set into another finite set which relates to input output functions in computer science.
- Be aware of the counting principles
- Be exposed to concepts and properties of algebraic structures such as semi groups, monoids and groups.

**UNIT I LOGIC AND PROOFS****9 + 3**

Propositional Logic – Propositional equivalences-Predicates and quantifiers-Nested Quantifiers-Rules of inference-introduction to Proofs-Proof Methods and strategy

**UNIT II COMBINATORICS****9+3**

Mathematical inductions-Strong induction and well ordering-.The basics of counting-The pigeonhole principle –Permutations and combinations-Recurrence relations-Solving Linear recurrence relations-generating functions-inclusion and exclusion and applications.

**UNIT III GRAPHS****9 + 3**

Graphs and graph models-Graph terminology and special types of graphs-Representing graphs and graph isomorphism -connectivity-Euler and Hamilton paths

**UNIT IV ALGEBRAIC STRUCTURES****9 + 3**

Algebraic systems-Semi groups and monoids-Groups-Subgroups and homomorphisms-Cosets and Lagrange's theorem- Ring & Fields (Definitions and examples)

**UNIT V LATTICES AND BOOLEAN ALGEBRA****9 + 3**

Partial ordering-Posets-Lattices as Posets- Properties of lattices-Lattices as Algebraic systems –Sub lattices –direct product and Homomorphism-Some Special lattices-Boolean Algebra

**L: 45, T: 15, TOTAL= 60 PERIODS****TEXT BOOKS:**

1. Kenneth H.Rosen, "Discrete Mathematics and its Applications", Special Indian edition, Tata McGraw-Hill Pub. Co. Ltd., New Delhi, (2007). (For the units 1 to 3, Sections 1.1 to 1.7 , 4.1 & 4.2, 5.1 to 5.3, 6.1, 6.2, 6.4 to 6.6, 8.1 to 8.5)
2. Trembly J.P and Manohar R, "Discrete Mathematical Structures with Applications to Computer Science", Tata McGraw-Hill Pub. Co. Ltd, New Delhi, 30<sup>th</sup> Re-print (2007).(For units 4 & 5, Sections 2-3.8 & 2-3.9,3-1,3-2 & 3-5, 4-1 & 4-2)

**REFERENCES:**

1. Ralph. P. Grimaldi, "Discrete and Combinatorial Mathematics: An Applied Introduction", Fourth Edition, Pearson Education Asia, Delhi, (2002).
2. Thomas Koshy, "Discrete Mathematics with Applications", Elsevier Publications, (2006).
3. Seymour Lipschutz and Mark Lipson, "Discrete Mathematics", Schaum's Outlines, Tata McGraw-Hill Pub. Co. Ltd., New Delhi, Second edition, (2007).

**CS2302****COMPUTER NETWORKS****L T P C  
3 0 0 3****UNIT I****9**

Network architecture – layers – Physical links – Channel access on links – Hybrid multiple access techniques - Issues in the data link layer - Framing – Error correction and detection – Link-level Flow Control

**UNIT II****9**

Medium access – CSMA – Ethernet – Token ring – FDDI - Wireless LAN – Bridges and Switches

**UNIT III****9**

Circuit switching vs. packet switching / Packet switched networks – IP – ARP – RARP – DHCP – ICMP – Queueing discipline – Routing algorithms – RIP – OSPF – Subnetting – CIDR – Interdomain routing – BGP – Ipv6 – Multicasting – Congestion avoidance in network layer

**UNIT IV****9**

UDP – TCP – Adaptive Flow Control – Adaptive Retransmission - Congestion control – Congestion avoidance – QoS

**UNIT V****9**

Email (SMTP, MIME, IMAP, POP3) – HTTP – DNS- SNMP – Telnet – FTP – Security – PGP - SSH

**TOTAL= 45 PERIODS****TEXT BOOK:**

1. Larry L. Peterson, Bruce S. Davie, "Computer Networks: A Systems Approach", Fourth Edition, Morgan Kauffmann Publishers Inc., 2009, Elsevier.

**REFERENCES:**

1. James F. Kuross, Keith W. Ross, "Computer Networking, A Top-Down Approach Featuring the Internet", Third Edition, Addison Wesley, 2004.
2. Nader F. Mir, "Computer and Communication Networks", Pearson Education, 2007
3. Comer, "Computer Networks and Internets with Internet Applications", Fourth Edition, Pearson Education, 2005.
4. Andrew S. Tanenbaum, "Computer Networks", Sixth Edition, 2003, PHI Learning.
5. William Stallings, "Data and Computer Communication", Sixth Edition, Pearson Education, 2000

**UNIT I AUTOMATA 9**

Introduction to formal proof – Additional forms of proof – Inductive proofs – Finite Automata (FA) – Deterministic Finite Automata (DFA) – Non-deterministic Finite Automata (NFA) – Finite Automata with Epsilon transitions.

**UNIT II REGULAR EXPRESSIONS AND LANGUAGES 9**

Regular Expression – FA and Regular Expressions – Proving languages not to be regular – Closure properties of regular languages – Equivalence and minimization of Automata.

**UNIT III CONTEXT-FREE GRAMMARS AND LANGUAGES 9**

Context-Free Grammar (CFG) – Parse Trees – Ambiguity in grammars and languages – Definition of the Pushdown automata – Languages of a Pushdown Automata – Equivalence of Pushdown automata and CFG– Deterministic Pushdown Automata.

**UNIT IV PROPERTIES OF CONTEXT-FREE LANGUAGES 9**

Normal forms for CFG – Pumping Lemma for CFL – Closure Properties of CFL – Turing Machines – Programming Techniques for TM.

**UNIT V UNDECIDABILITY 9**

A language that is not Recursively Enumerable (RE) – An undecidable problem that is RE – Undecidable problems about Turing Machine – Post's Correspondence Problem – The classes P and NP.

**L: 45, T: 15, TOTAL= 60 PERIODS**

**TEXT BOOK:**

1. J.E. Hopcroft, R. Motwani and J.D. Ullman, "Introduction to Automata Theory, Languages and Computations", second Edition, Pearson Education, 2007.

**REFERENCES:**

1. H.R. Lewis and C.H. Papadimitriou, "Elements of the theory of Computation", Second Edition, Pearson Education, 2003.
2. Thomas A. Sudkamp, "An Introduction to the Theory of Computer Science, Languages and Machines", Third Edition, Pearson Education, 2007.
3. Raymond Greenlaw and H. James Hoover, "Fundamentals of Theory of Computation, Principles and Practice", Morgan Kaufmann Publishers, 1998.
4. Micheal Sipser, "Introduction of the Theory and Computation", Thomson Brokecole, 1997.
5. J. Martin, "Introduction to Languages and the Theory of computation" Third Edition, Tata Mc Graw Hill, 2007

**AIM**

To have an understanding of foundations of design of assemblers, loaders, linkers, and macro processors.

**OBJECTIVES**

- To understand the relationship between system software and machine architecture.
- To know the design and implementation of assemblers
- To know the design and implementation of linkers and loaders.
- To have an understanding of macroprocessors.
- To have an understanding of system software tools.

**UNIT I INTRODUCTION 8**

System software and machine architecture – The Simplified Instructional Computer (SIC) - Machine architecture - Data and instruction formats - addressing modes - instruction sets - I/O and programming.

**UNIT II ASSEMBLERS 10**

Basic assembler functions - A simple SIC assembler – Assembler algorithm and data structures - Machine dependent assembler features - Instruction formats and addressing modes – Program relocation - Machine independent assembler features - Literals – Symbol-defining statements – Expressions - One pass assemblers and Multi pass assemblers - Implementation example - MASM assembler.

**UNIT III LOADERS AND LINKERS 9**

Basic loader functions - Design of an Absolute Loader – A Simple Bootstrap Loader - Machine dependent loader features - Relocation – Program Linking – Algorithm and Data Structures for Linking Loader - Machine-independent loader features - Automatic Library Search – Loader Options - Loader design options - Linkage Editors – Dynamic Linking – Bootstrap Loaders - Implementation example - MSDOS linker.

**UNIT IV MACRO PROCESSORS 9**

Basic macro processor functions - Macro Definition and Expansion – Macro Processor Algorithm and data structures - Machine-independent macro processor features - Concatenation of Macro Parameters – Generation of Unique Labels – Conditional Macro Expansion – Keyword Macro Parameters-Macro within Macro-Implementation example - MASM Macro Processor – ANSI C Macro language.

**UNIT V SYSTEM SOFTWARE TOOLS 9**

Text editors - Overview of the Editing Process - User Interface – Editor Structure. - Interactive debugging systems - Debugging functions and capabilities – Relationship with other parts of the system – User-Interface Criteria.

**L: 45, T: 15, TOTAL= 60 PERIODS**

**TEXT BOOK:**

1. Leland L. Beck, “System Software – An Introduction to Systems Programming”, 3<sup>rd</sup> Edition, Pearson Education Asia, 2006.

**REFERENCES:**

1. D. M. Dhamdhare, "Systems Programming and Operating Systems", Second Revised Edition, Tata McGraw-Hill, 2000.
2. John J. Donovan "Systems Programming", Tata McGraw-Hill Edition, 2000.
3. John R. Levine, Linkers & Loaders – Harcourt India Pvt. Ltd., Morgan Kaufmann Publishers, 2000.

**CS2305****PROGRAMMING PARADIGMS****L T P C  
3 0 0 3****AIM:**

To understand the concepts of object-oriented, event driven, and concurrent programming paradigms and develop skills in using these paradigms using Java.

**UNIT I OBJECT-ORIENTED PROGRAMMING – FUNDAMENTALS 9**

Review of OOP - Objects and classes in Java – defining classes – methods - access specifiers – static members – constructors – finalize method – Arrays – Strings - Packages – JavaDoc comments

**UNIT II OBJECT-ORIENTED PROGRAMMING – INHERITANCE 10**

Inheritance – class hierarchy – polymorphism – dynamic binding – final keyword – abstract classes – the Object class – Reflection – interfaces – object cloning – inner classes – proxies

**UNIT III EVENT-DRIVEN PROGRAMMING 10**

Graphics programming – Frame – Components – working with 2D shapes – Using color, fonts, and images - Basics of event handling – event handlers – adapter classes – actions – mouse events – AWT event hierarchy – introduction to Swing – Model-View-Controller design pattern – buttons – layout management – Swing Components

**UNIT IV GENERIC PROGRAMMING 8**

Motivation for generic programming – generic classes – generic methods – generic code and virtual machine – inheritance and generics – reflection and generics – exceptions – exception hierarchy – throwing and catching exceptions – Stack Trace Elements - assertions - logging

**UNIT V CONCURRENT PROGRAMMING 8**

Multi-threaded programming – interrupting threads – thread states – thread properties – thread synchronization – thread-safe Collections – Executors – synchronizers – threads and event-driven programming

**TOTAL=45 PERIODS****TEXT BOOK:**

1. Cay S. Horstmann and Gary Cornell, "Core Java: Volume I – Fundamentals", Eighth Edition, Sun Microsystems Press, 2008.

**REFERENCES:**

1. K. Arnold and J. Gosling, "The JAVA programming language", Third edition, Pearson Education, 2000.
2. Timothy Budd, "Understanding Object-oriented programming with Java", Updated Edition, Pearson Education, 2000.
3. C. Thomas Wu, "An introduction to Object-oriented programming with Java", Fourth Edition, Tata McGraw-Hill Publishing company Ltd., 2006.

**CS2307****NETWORK LAB****L T P C  
0 0 3 2**

1. Programs using TCP Sockets (like date and time server & client, echo server & client, etc..)
2. Programs using UDP Sockets (like simple DNS)
3. Programs using Raw sockets (like packet capturing and filtering)
4. Programs using RPC
5. Simulation of sliding window protocols  
Experiments using simulators (like OPNET)
6. Performance comparison of MAC protocols
7. Implementing Routing Protocols
8. Performance comparison of Routing protocols
9. Study of UDP performance
10. Study of TCP performance.

**TOTAL = 45 PERIODS****Requirement for a batch of 30 students**

<b>S.No.</b>	<b>Description of Equipment</b>	<b>Quantity required</b>
1.	SOFTWARE <ul style="list-style-type: none"> <li>➤ C++ Compiler</li> <li>➤ J2SDK (freeware)</li> <li>➤ Linux</li> <li>➤ NS2/Glomosim/OPNET (Freeware)</li> </ul>	30
2.	Hardware <ul style="list-style-type: none"> <li>➤ PCs</li> </ul>	30 Nos

**(Using C)**

1. Implement a symbol table with functions to create, insert, modify, search, and display.
2. Implement pass one of a two pass assembler.
3. Implement pass two of a two pass assembler.
4. Implement a single pass assembler.
5. Implement a two pass macro processor
6. Implement a single pass macro processor.
7. Implement an absolute loader.
8. Implement a relocating loader.
9. Implement pass one of a direct-linking loader.
10. Implement pass two of a direct-linking loader.
11. Implement a simple text editor with features like insertion / deletion of a character, word, and sentence.
12. Implement a symbol table with suitable hashing

(For loader exercises, output the snap shot of the main memory as it would be, after the loading has taken place)

**TOTAL=45 PERIODS**

**Requirement for a batch of 30 students**

<b>S.No.</b>	<b>Description of Equipment</b>	<b>Quantity required</b>
1.	Hardware – Pentium PC Desktops	30 Nos.
2.	Software – Turbo C (Freely download)	Multiusers

1. Develop Rational number class in Java. Use JavaDoc comments for documentation. Your implementation should use efficient representation for a rational number, i.e. (500 / 1000) should be represented as ( $\frac{1}{2}$ ).
2. Develop Date class in Java similar to the one available in java.util package. Use JavaDoc comments.
3. Implement Lisp-like list in Java. Write basic operations such as 'car', 'cdr', and 'cons'. If L is a list [3, 0, 2, 5], L.car() returns 3, while L.cdr() returns [0,2,5].
4. Design a Java interface for ADT Stack. Develop two different classes that implement this interface, one using array and the other using linked-list. Provide necessary exception handling in both the implementations.
5. Design a Vehicle class hierarchy in Java. Write a test program to demonstrate polymorphism.
6. Design classes for Currency, Rupee, and Dollar. Write a program that randomly generates Rupee and Dollar objects and write them into a file using object serialization. Write another program to read that file, convert to Rupee if it reads a Dollar, while leave the value as it is if it reads a Rupee.
7. Design a scientific calculator using event-driven programming paradigm of Java.
8. Write a multi-threaded Java program to print all numbers below 100,000 that are both prime and fibonacci number (some examples are 2, 3, 5, 13, etc.). Design a thread that generates prime numbers below 100,000 and writes them into a pipe. Design another thread that generates fibonacci numbers and writes them to another pipe. The main thread should read both the pipes to identify numbers common to both.
9. Develop a simple OPAC system for library using even-driven and concurrent programming paradigms of Java. Use JDBC to connect to a back-end database.
10. Develop multi-threaded echo server and a corresponding GUI client in Java.
11. [Mini-Project] Develop a programmer's editor in Java that supports syntax-highlighting, compilation support, debugging support, etc.

**TOTAL= 45 PERIODS**

#### Requirement for a Batch of 30 Students

S. No.	Description of Equipment	Quantity Required
1.	PC's	30
2.	JUM & J2SE (Freeware)	30
3.	MYSQL or any other DB	30