COURSE STRUCTURE AND SYLLABUS APPROVED IN THE BOARD OF STUDIES MEETING HELD AT ON 2001 TO BE EFFECTIVE FROM THE ACADEMIC YEAR 2000-2001.

M.Tech (Computer Science)
II Semester

				Scheme of Evaluation				
				Max. marks			Min. marks To pass	
Subject	L	P	C	Int.	Ext.	Total	Ext.	Total
MT2.1 Programming in Java	4	-	8	40	60	100	24	50
MT2.2 Software Engineering	4	-	8	40	60	100	24	50
MT2.3 Object Oriented Analysis And design (UML)	4	-	8	40	60	100	24	50
MT2.4 Advanced Unix Programming	4	-	8	40	60	100	24	50
MT2.5 Elective – I (a) Language Processors (b) Design and Analysis of Algorithms (c) Distributed database (d) Data mining and ware housing	4	-	8	40	60	100	24	50
MT2.6 Elective – II (a) Network security and E-Commerce (b) Software testing and methodology (c) Programming Languages (d) Image Processing & Patterns Recognition	4	-	8	40	60	100	24	50
MT2.7 Java Lab	-	4	4	40	60	100	24	50
MT2.8 OS Lab – through Unix	-	4	4	40	60	100	24	50

MT2.1 PROGRAMMING IN JAVA

1. introduction – Genesis of Java. Overview of Java, datatypes, variables, arrays, operators and control statements.

2. Classes and Objects: - Concepts of classes and objects, method overloading constructors, constructor overloading, usage of static with data and methods, usage of final with data, methods and classes, Garbage collector call by value, call by reference, access control recursion, nested classes, inner classes, anonymous inner classes.

3. Inheritence:- Concepts, composition, difference between inheritances in Java, usage of super keyword, method overloading, abstract classes, dynamic method dispatch.
   Packages:- Concepts, Package & import keywords, class path, defining, creating and accessing a package.

4. Interfaces:- Difference between classes and interfaces, application of interfaces, Multiple inheritence in Java, extending and initializing fields in interfaces.
   Exception handling:- concept of exception handling, types of exceptions, usage of try, catch, throw, throws, finally keywords.

5. Multithreading:- Concepts of multithreading, life cycle difference between process and thread, creating multiple threads using thread class of runnable interface, synchronization, thread priorities, interthread communication, daemon threads, deadlocks and thread groups.

6. Event Handling:- Event classes, event listeners, delegation event model, handling mouse & keyboard events, adapter classes.
   AWT:- concepts of components, container, panel, window, frame, canvas, font class, color class and graphics.

7. AWT Controls:- Buttons, labels, text field, text area, check boxes, check box groups (Radio buttons), lists, choice, scroll bars, layout managers viz, flow, border, grid, card gridbay.
   Applets:concept of applet, life cycle of applets, types of applets, creating applets, applet communication.

8. Java Library:- String handling, java.util, java.net, java.io
   Networking:-Basics of networking, inetaddress, TCP/IP sockets, data grams, URL, URL connection.

Suggested Books:-

1. Bruce Eckel : Thinking in Java, 1999 – Prentice Hall, OTK.
2. Herbert Schildt & Fatrick nanghton :The complete reference Java 2.0
3. Iver Harton: Beginning in Java 2.0 – Wrax publications.
4. Dietal & Dietal : Java 2.0 – How to programming.

MT2.2 SOFTWARE ENGINEERING

1. Software and Software Engineering
   The importance of software – software – software myths – software engineering paradigms – generic view of software engg.

2. Software metrics
   Measures and metrics – estimation – risk analysis – scheduling – size oriented metrics – function oriented metrics – metrics of software quality.

3. Software project estimation and Planning
   Decomposition techniques – LOC and FP estimation – effect estimation-risk analysis-identification-projection-assessment-management and monitoring-software re-engineering.

4. Requirement Analysis:
   Requirement analysis-tasks-analyst-software prototyping-specification principles-representation and the software requirements specification.

5. Object oriented analysis and data modeling
   Object oriented concepts – identifying objects – specifying attributes – defining operations – inter object communication – finalizing object definition – object oriented analysis modeling – data modeling – data objects, attributes and relationships – entity relationship diagrams.

6. Alternative analysis techniques
   Requirement analysis methods – data structure oriented methods – data structured system development – warner diagrams and the DSSD approach – Jackson system development.

7. Software Design fundamentals
   The design process – design fundamentals – effective modular design – dataflow oriented design – transform analysis – transaction analysis – design heuristics.

8. Object Oriented Design.
   Object oriented design concepts – object oriented design methods – refining operations – program components & interfaces – implementation detail design.

9. User interface design
   Human factors – Human computer interface design – interface design guidelines – Interface standards.

10. Software Quality Assurance
    Software quality factors – quality assurance, quality metrics, Halstead's S/W science.

11. Software testing techniques
    S/W Testing Fundamentals – white Box testing, Black box Testing, validation Testing, system testing, debugging.

12. Software Maintenance
    Maintainability – maintenance tasks – reverse engineering and Re-engineering.

Text Book: Roger S. Pressman – "Software Engineering", Mc Graw Hill.

OBJECT ORIENTED ANALYSIS AND DESIGN USING UML

1. Requirements Engineering: Requirements Engineering Process, Software requirement document: requirement validation, Requirements evolution.
   Requirements Analysis: viewpoint oriented analysis, Method-Based analysis system contexts, social and organizational factors.
   System models: data-flow models, semantic data models, object models.
   Requirements definition and specification: Requirements definition, requirements specification, Non-functional requirements.

2. Introduction to UML: The meaning of Object-Orientation, object identity, encapsulation, information hiding, polymorphism, genericity, importance of modeling, principles of modeling, object oriented modeling, conceptual model of the UML, Architecture.

3. Basic structural Modeling: classes, relationships, common mechanisms, diagrams, advanced structural modeling: advanced relationship interfaces, roles, packages, instances.

4. Class & object diagrams: Terms, concepts, examples, modeling techniques, class & Object diagrams.

5. Collaboration Diagrams: Terms, Concepts, depicting a message, polymorphism in collaboration diagrams, iterated messages, use of self in messages.
   Sequence diagrams: Terms, concepts, differences between collaboration and sequence diagrams, depicting synchronous messages with/without priority call back mechanism broadcast message.

6. Behavioral Modeling: Interactions, usecases, usecase diagrams, activity diagrams.

7. Advanced Behavioral Modeling: Events and signals, state machines, process and threads, time and space, state chart diagrams.

8. Architectural Modeling: Terms, concepts, examples, modeling techniques for component diagrams and deployment diagrams.

Suggested Reading

1. Grady Boach, James Rambaugh, Ivar Jacobson : The unified modeling language user guide, Addison Wesley, 1999.

2. Meilir page-jones: fundamentals of object oriented design in UML, Addison Wesley, 2000.

3. Ian Sommerville: Software Engineering, Fifth Edition, Addison Wesley.

MT.2.4 ADVANCED UNIX PROGRAMMING

1. Unix Utilities – I
   Introduction to Unix file system. Vi editor, File handling utilities, security by file permissions, process utilities, disk utilities, Networking commands, cp, mv, ln, rm, unlink, mkdir, rmdir, du, df, mount, umount, find, umask, ulimit, ps, who, w, finger, arp, ftp, telnet, rlogin.

2. Unix utilities – 2
   Text processing utilities and backup utilities detailed commands to be covered are: cat, tail, head, sort, nl, uniq, grep, egrep, fgrep, cut, paste, join, tee, more, pg, comm., cmp, diff, tr, awk, tar, cpio.

3. What is a shell, shell responsibilities, pipes and input redirection, Output redirection and here documents, the shell as programming Language shell variables, conditions, history and control structures and shell programming.

4. Unix Internals - 1
   Unix file structure, directories, files and devices, system calls and device drivers, library functions, low-level file access (write, read, open, close, ioctl, lseek, fstat, stat, dup and dup2), the standard I/O (fopen, fread, fclose,fflush, fseek, fgetc, getc, getchar, fputc, putc, putchar, fgets, gets), formatted, I/O stream errors, streams, and file descriptors, file and directory maintainance (chmod, chown, unlink, link symlink, mkdir, rmdir, chdir, getcwd).

5. Unix internals – 2
   Process and signals:- what is process, process structure, starting new process, waiting for a process, zombie process, process control, process identifiers, fork function, vfork, exit, wait, exec, system, functions, user identification, process times.
   Signal: Signal functions, reliable signals, interrupted system calls, kill and raise functions, alarm, pause functions, abort, system, sleep functions.

6. Unix Internals – 3
   Data Management:- Management Memory (simple memory allocation, freeing memory) file locking (creating lock files, locking regions, use of read/write locking, competing locks, other commands, deadlocks).

7. Unix Internals – 4
   Inter-process:- Pipe, process pipes, the pipe call, parent-child process, named pipes: FIFOs), Semaphores, message queues and shared memory applications of IPC.

References:

1. Advanced programming in Unix Environment (W. Richard Stevens).
2. Unix Network Programming (W. Richard Stevens).

MT 2.5(a) LANGUAGE PROCESSORS

1. Lexical Analysis, scanning process, regular grammars and regular expressions, state transition diagrams, minimization of number of states, scanning algorithm, LEX program.

2. Top down parsing, parse tree representation, brute force approach, recursive descent parsing, predicted parsers, LL(1) grammars.

3. Bottom up parsing, operator precedence grammar and corresponding parsing algorithm, simple precedence grammars, precedence functions, LR grammars, LR parsers, LALR(1) parsers, YACC program.

4. Symbol tables – data in symbol tables, symbol table organization hashing, tree structured; symbol table organization for block structured languages, representation PASCAL datatypes in symbol table storage allocation storage for arrays, strings, records etc. Run time storage Organization.

5. Semantic Analysis and code generation, Intermediate forms of source programs polish notation, N tuples, abstract syntax trees, transformation from infix to internal forms, sematic stacks, attributed translation grammars.

6. Code Optimization folding, redundant sub expression elimination, loop optimization unrolling, frequency reduction, strength reduction, global optimization using flow graph analysis.

7. Object code generation problems in object code generation, register allocation algorithms, object modules.

Text Books:

1. J. P. Tremblay & P. G. Sorenson The theory and practice of compiler writing Mc Graw 1985.

2. A. V. Aho & J. D. Ullman Principles of Compiler Design, Addison Wesley/Narosa 1985.

MT 2.5(b) DESIGN AND ANALYSIS OF ALGORITHMS

1. INTRODUCTION: Algorithm, pseudo code for expressing algorithms, analysis, time complexity and space complexity, O-notation, omega notation and theta notation, Heaps and Heap sort, sets and disjoint set, union and find algorithms.

2. DIVIDE AND CONQUER: General method, merge sort, quick sort, strassen's matrix multiplication.

3. GREEDY METHOD: General method, optimal storage on tapes, knapsack problem job sequencing with deadlines, minimum spanning tree, single source shortest paths.

4. DYNAMIC PROGRAMMING: General method, Multistage Graphs, optimal binary search trees, O/I knapsack problem, reliability design problem. Traveling sales person problem, floor shop scheduling.

5. SEARCHING AND TRAVERSAL TECHNIQUES: Efficient non recursive binary tree traversal algorithms, tree traversal, breadth first search and traversal, depth first search and traversal, AND/OR graphs, game tree, Biconnected components.

6. BACK TRACKING: General method, n-queen problem, sum of subsets problem, graph colouring, Hamiltonian cycles.

7. BRANCH AND BOUND: LC search, bounding, LC branch and bound, FIFO branch and bound, traveling sales person problem.

8. NP-HARD AND NP-COMPLETE PROBLEMS: Basic concepts, non-deterministic algorithms, NP-HARD and NP-COMPLETE classes, COOKS theorem.

BOOKS:

1. E. Howrowitz and Shani, Fundamentals of computer algorithms – GALGOTIA PUBLICATIONS.

2. ULLMAN, Design and analysis of algorithms, Addison Wesley-1994.

MT2.5(C) DISTRIBUTED DATABASES

1. Features of distributed databases, features of centralized databases, level of distributed transparency – Reference Architecture, types of Data Fragmentation, distribution Transparency, Access primitives, Integrity constraints.

2. Distributed Database design – A frame work, the design of database fragmentation, the allocation of fragments. Translation of global queries into fragment queries, query optimization.

3. Distributed Transaction Management – A framework, transaction atomicity, 2-phase commit, concurrency control: fundations, distributed deadlocks, timestamps.

4. Reliability: Basic concepts, commit protocols, consistent view of Network, Detection and Resolution of Inconsistencies, check points and cold restart.

5. Commercial Systems: Tranclem's ENCOMPASS
   Distributed database systems, IBM's Inter system communication, feature of distributed ingres and Oracle.

6. Heterogeneous databases: General problems – brief study of multibase.

Text Book:

1. Ceri S. Pelagatti. G, Distributed Database systems Principles and Systems, Mc Graw Hill.

2.6(B) SOFTWARE TESTING METHODOLOGY

1. Introduction:- Purpose of testing, Dichotomies, model for testing, consequences of bugs, taxonomy of bugs.

2. Flow graphs and Path testing:- Basics concepts of path testing, predicates, path predicates and achievable paths, path sensitizing, path instrumentation, application of path testing.

3. Transaction Flow Testing:- Transaction flows, transaction flow testing techniques. Dataflow testing:- Basics of dataflow testing, strategies in dataflow testing, application of dataflow testing.

4. Domain Testing: Domains and paths, Nice and Ugly domains, domain testing, domains and interfaces testing, domain and interface testing, domains and testability.

5. Paths, path products and Regular expressions:-
   Path products & Path expression, reduction procedure, applications, regular expressions and flow anomaly detection.

6. Logic Based Testing:- Overview, decision tables, path expressions, kv charts, specifications.

7. State, State Graphs and Transition Testing:- State Graphs, good and bad state graphs, state testing, testability tips.

8. Graph matrices and Application:- Motivational overview, matrix of graph relations, power of a matrix, node reduction algorithm, building tools.

REFERENCES:

1. Software Testing Techniques – Baris Beizer, International Thomson Computer Press, second edition.

2. The craft of software testing – Brian Marick, Prentice Hall series in innovative technology.

MT2.6(C) PROGRAMMING LANGUAGES

1. Criteria for programming language design, defining syntax, BNF, syntax graphs.

2. Variables, expressions and statements, variable binding and storage allocation, constants and initialization, expressions, Assignments, conditional statements, iterative statements, GO TO statements and labels.

3. Types Data types and typing, elementary data types, enumerated data types, pointer data types, structured data types, type coertion, type equivalence.

4. Scope, Extent and procedures basics, runtime storage allocation, parameter evaluation and passing.

5. Data abstraction, abstract data types, abstract specification, modularity, data obstraction in ADA exception handling.

6. Concurrency Basic concepts, semaphores, monitors and concurrent PASCAL, message passing, concurrency in ADA.

7. Functional programming and LISP, Basics of LISP, recursive interpreter of LISP, storage reclamation and LISP, PROLOG feature in LISP.

8. Logic Programming in PROLOG.

Text Books:

1. Ellis Horcuitz – Fundamentals of Programming Languages, Galgotia Publications, ND 1984.

2. Gheizi & Jezzayari – Programming Languages.

3. Clocksin & Mellish – Logic Programming.

4. Winston – LISP.