

M.Sc.(Computer Science)/ IT Course Curriculum

III- Semester

Code	Subject	Practical /Project	End Sem	Total
CS-5011	Embedded Systems K	-	50	50
CS-4408	Database Application and Tools S	50	50	100
CS-5123	Theory of Computation S	-	50	50
CS-4508	Computer Graphics and Multimedia K K	50	50	100
CS-4211	Object Oriented Programming using JAVA A	50	50	100
		150	250	400

IV- Semester

SUB_CODE	NAME	Practical /Project	End Sem	Total
CS-4517	Linux/UNIX Administration	50	50	100
CS-5512	Compiler Design	-	50	50
CS-5178	Internet & Web Technology	50	50	100
CS - 5216	Design and Analysis of Algorithms	50	50	100
CS-5802B	Project Viva	50	-	50
Total		200	200	400

Scheme for award of degree shall be same as for MBA(CM).

Baruwani
28/01/09

Am 17/01/09

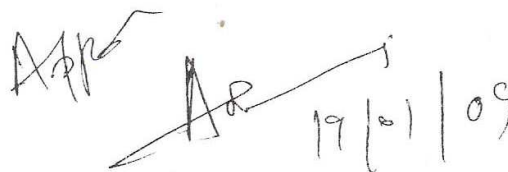
Social
IT

Course Specification

CS-506 Embedded Systems

Course Description

DATE	TOPIC	READING
Week 1-2	Introduction To Embedded Systems : Definition and Classification – Overview of Processors and hardware units in an embedded system – Software embedded into the system – Exemplary Embedded Systems – Embedded Systems on a Chip (SoC) and the use of VLSI designed circuits. Assignment Set 1	Chapter 1 Embedded System by Dr. Rajkamal
Week 3-4	Devices And Buses For Devices Network : I/O Devices - Device I/O Types and Examples – Synchronous - Iso-synchronous and Asynchronous Communications from Serial Devices - Examples of Internal Serial-Communication Devices - UART and HDLC - Parallel Port Devices - Sophisticated interfacing features in Devices/Ports- Timer and Counting Devices - 'I ² C', 'USB', 'CAN' and advanced I/O Serial high speed buses- ISA, PCI, PCI-X, cPCI and advanced buses. Assignment Set 2	Chapter 3 Embedded System by Dr. Rajkamal
Week 5-6	Programming Concepts And Embedded Programming in C, C++ : Programming in assembly language (ALP) vs. High Level Language - C Program Elements, Macros and functions -Use of Pointers - NULL Pointers - Use of Function Calls – Multiple function calls in a Cyclic Order in the Main Function Pointers.	Chapter 5 Embedded System by Dr. Rajkamal
Week 7	Concepts of EMBEDDED PROGRAMMING in C++ - Objected Oriented Programming – Embedded Programming in C++, 'C' Program compilers – Cross compiler – Optimisation of memory codes. Assignment Set 3	Chapter 5 Embedded System by Dr. Rajkamal
Week 8-9	Real Time Operating Systems – Part 1 Definitions of process, tasks and threads – Clear cut distinction between functions – ISRs and tasks by their characteristics – Operating System Services- Goals – Structures- Kernel - Process Management – Memory Management – Device Management – File System Organisation and Implementation – I/O Subsystems – Interrupt Routines Handling in RTOS. Assignment Set 4	Chapter 8,9 Embedded System by Dr. Rajkamal
Week 9-10	Real Time Operating Systems –Part 2 : RTOS Task scheduling models - Handling of task scheduling and latency and deadlines as performance metrics – Co-operative Round Robin Scheduling – Cyclic Scheduling with Time Slicing (Rate Monotonics Co-operative Scheduling) – Preemptive Scheduling Model strategy by a Scheduler – Critical Section Service by a Preemptive Scheduler – Fixed	Chapter 9 Embedded System by Dr. Rajkamal



 AR 19/11/09

	(Static) Real time scheduling of tasks - Inter Process Communication And Synchronisation – Shared data problem - Use of Semaphore(s) –	
Week 11	Priority Inversion Problem and Deadlock Situations – Inter Process Communications using Signals – Semaphore Flag or mutex as Resource key – Message Queues – Mailboxes – Pipes – Virtual (Logical) Sockets – Remote Procedure Calls (RPCs). Assignment Set 5	Chapter 9 Embedded System by Dr. Rajkamal
Week 12-13	Real Time Operating Systems – Part 3: Study of Micro C/OS-II and Vx Works or Any other popular RTOS – RTOS System Level Functions – Task Service Functions – Time Delay Functions – Memory Allocation Related Functions – Semaphore Related Functions Assignment Set 6	Chapter 10 Embedded System by Dr. Rajkamal
Week 14	– Mailbox Related Functions – Queue Related Functions – Case Studies of Programming with RTOS – Understanding Case Definition – Multiple Tasks and their functions – Creating a list of tasks – Functions and IPCs – Exemplary Coding Steps.	Chapter 10 Embedded System by Dr. Rajkamal

Learning Resources

1. Required Text(s)
Text Books : 1. Embedded Systems – Architecture, Programming & Design by Dr. Rajkamal, First Edition, Third reprint 2004, TataMcGraw-Hill Publishing Company Limited, New Delhi.
2. Essential References:
1. An Embedded Software Primer by David E. Simon, Eleventh Indian Reprint 2004, Pearson Education (Singapore) Pte. Ltd., Indian Branch, Delhi.
3- Recommended Books and Reference Material (Journals, Reports, etc) (Attach List)
1. Fundamentals of Embedded Software – where C and assembly meet by Daniel W. Lewis, Fifth Indian Reprint 2004, Pearson Education (Singapore) Pte. Ltd., Indian Branch, Delhi.
4- Electronic Materials, Web Sites etc
1. http://en.wikipedia.org/wiki/Embedded_system
2. http://www.embedded.com/
3. http://www.freertos.org/implementation/
5- Other learning material such as computer-based programs/CD, professional standards/regulations

Assignments:

Assignment Set 1:

1. Classify the embedded systems into small scale, medium scale and sophisticated systems. Now reclassify the embedded systems with and without real-time (response time constrained) systems. Give ten examples of each.
2. Search definitions of embedded systems in books and tabulate these with definitions in column 1 and reference in column 2.
3. Why does a CMOS IO circuit power dissipation reduces by compared to 5 V, factor of half, $\sim (3.3/5)^2$, in IO 3.3 V operation ?
4. How small shall be reduction in power dissipation for a processor CMOS circuit when V reduces from 5 V to 1.8 V operation?
5. Tabulate advantages and disadvantages of using coding language as following: a) Final Machine Implementable b) Assembly language programming c) C d) C++ e) Java.
6. Justify the importance of device drivers in an embedded system.
7. Cost of designing an embedded system may be thousands of times the cost of its processor and hardware units. Explain this statement.

Assignment Set 2:

1. A generation automobile has about 100 embedded systems. How do the bus arbitration bits, control bits for address and data length, data bits, CRC check bits, acknowledgement bits and ending bits in CAN bus help the networking devices distributed in an automobile embedded system.
2. How does USB protocol provide for a device attachment, configuration, reset configuration, bandwidth sharing with other devices and device detachment (while others are in operation) and reattachment.
3. Design a table that compares the maximum operational speeds and bus lengths and give two examples of uses of each of the following serial devices a) UART b) 1-wire CAN c) Industrial I²C d) SM I²C bus e) SPI of 68 series Motorola microcontrollers f) fault tolerant CAN g) Standard Serial port h) Micro wire i) I²C j) High speed CAN k) IEEE 1284 l) High speed I²C m) USB 1.1 low speed channel and High Speed channel n) SCSI parallel o) Fast SCSI p) Ultra-SCSI-3 q) Firewire / IEEE 1394 r) High speed USB 2.0.
4. Design a table that compares the maximum operational speeds and bus lengths and give two examples of uses of each of the following parallel devices: a) ISA b) EISA c) PCI d) PCI-X e) COMPACT PCI f) GMII(Gigabit Ethernet MAC Interchange Interface) g) XGMI(10 Gigabit Ethernet MAC Interchange Interface) Rapid IOTM Interconnect specification v1.1 at 8 Gbps with 500 Mbps performance or 250 MHz dual direction registering performance using 8 bit LVDS (Low voltage data bus).

Assignment Set 3:

1. What are the criteria by which an appropriate programming language is chosen for embedded software of a given system?
2. What is the most important feature in C that makes it a popular high-level language for an embedded system ?

3. What is the most important feature in Java that makes it a highly useful high level language for an embedded system in many network related applications ?
4. Design a table to give the features of top-down and bottom-up design of a program.
5. Why do we need a cross compiler ?
6. Why do we use infinite loop in embedded system software ?
7. What are the advantages of reentrant functions in embedded system software ?
8. What are the advantages of multiple function calls in cyclic order in main ?
9. What are the advantages of building ISR queues ?
10. What are the advantages of having short ISRs that build function queues for processing at a later time ?
11. How are the queues used for a network ?
12. What are the commonly used preprocessor directives? Give four examples of each.
13. How does the use of a macro differ from a function ? Explain with codes.
14. How does combining two functions reduce the memory requirement? Explain with four examples.

Assignment set 4 :

1. How does a data output generated by a process transfer to another using an IPC(Inter process Communication) ?
2. What are the parameters of TCB of a task ? Why should each task have distinct TCB ?
3. What are the states of a task ? Which is the entity controlling (scheduling) the transitions from one state to another in a task ?
4. Define critical section of a task.
5. How does use of a counting semaphore differs from a mutex ? How is a counting semaphore used ?
6. Give an example of a deadlock situation during multiprocessing (multitasking) execution.
7. What are the advantages and disadvantages of disabling interrupts during the running of a critical section of a process ?
8. Each process or task has an endless (infinite) loop in a preemptive scheduler. How does the control of resources transfer from one task to another ?
9. How do the functions differ from ISRs, tasks, threads and processes?
10. List the features of P and V Semaphores. How they are used as a resource key, counting semaphore and as a mutex ?
11. What are the situations that lead to priority inversion problems? How does the OS solve this problem by a priority inheritance mechanism?
12. What is meant by a pipe? How does a pipe may differ from a queue ?
13. What are the analogies between process, task and thread ? Also list differences among them.
14. What is the advantage of using a signal as an IPC ?
15. Can different IPCs be used ? Given the choice, how will you select an IPC from signal, semaphore, queue and mailbox ?

Assignment Set 5:

1. List the layers between application and hardware.
2. Explain the term process descriptor and process control block(PCB). What are the analogies between a PCB and TCB?
3. Define a network operating system. How does a network OS differ from a conventional OS?
4. Give examples of IO subsystems.

5- Other learning material such as computer-based programs/CD, professional standards/regulations:

CS-512 Theory of Computation

CLASS SCHEDULE:

DATE	TOPIC	UNIT
Week 1	The Theory Of Automata: String, Alphabets and Languages, Finite Automata, Finite State Machine, Basic Definition. Description of a Finite Automaton,	Chapter2
Week 2	Deterministic Finite Acceptors- Transition Graphs, Languages, Non-Deterministic Finite Acceptors- Definition, Finite Automata with ϵ -moves	Chapter2
Week 3	Equivalence of Deterministic and Nondeterministic Finite Acceptors, Mealy and Moore models-Definitions, Transformation of Mealy Machine into Moore Machine and vice-versa	Chapter2
Week 4	Conversion of NDFA to DFA Removal of ϵ transition from ϵ – NDFA.	Chapter2
Week 5	The Myhill-Nerode theorem and Minimization of Finite Automata – Definition and Construction.	Chapter3
Week 6	Properties of Regular Sets: Pumping lemma for regular set, Closure properties of regular set.	Chapter3-4
MID TERM EXAM		
Week 7	Formal Language: Basic Definition, Chomsky Classification of languages, Initialization of Finite Automata Regular Expression and Language Regular Expressions, Connection between Regular Expressions and Regular Languages	Chapter3-4
Week 8	Regular Grammars – Right and Left Linear Grammars, Equivalence between Regular Languages and Regular Grammars	Chapter4
Week 9	Context-Free Grammars: Leftmost and Rightmost Derivations, Derivation Trees, Parsing and Ambiguity, Simplification of CFGs	Chapter5
Week 10	Chomsky Normal Form, Greibach Normal Form, Cocke-Kasami-Younger Algorithm, Properties of Context-Free Languages	Chapter5
Week 11	Pushdown Automata: Definition, Non deterministic Pushdown Automata, Pushdown Automata for Context Free Languages.	Chapter6
MID TERM EXAM		
Week 12	Context-Free Grammars for Pushdown Automata. Deterministic Pushdown Automata and Deterministic Context-Free Languages.	Chapter6
Week 13	Turing Machine: Definition of Standard Turing Machine, Turing Machine as Language Acceptors and Transducers.	Chapter7

Learning Resources

1. Required Text(s)

1. Mishra and Chandrasekaran, *Theory of Computer Science (Automata, language and Computation)*, 2nd Ed. Prentice Hall of India.
2. J. E. Hopcroft, R. Motwani and J.D Ullman, *Introduction to Theory, Languages and Computation; Second Edition*, Addison-Wesley, 2001 Narosa Publishing House.

2. Essential References

3. Moll, Arbib and Kfoury, *An Introduction to Formal Language Theory*, Springer-Verlag.
4. Martin, J.C.: *Introduction to Languages and the Theory of Computation*, McGraw-Hill, Inc., 3rd ed., 2002. ISBN 0-072-32200-4.
5. Brookshear, J.G.: *Theory of Computation: Formal Languages, Automata, and Complexity*, The Benjamin/Cummings Publishing Company, Inc, Redwood City, California, 1989. ISBN 0-805-30143-7
6. Peter Linz, *An Introduction to Formal Languages and Automata*, Narosa Publishing House.

3. Electronic Materials, Web Sites etc: www.scs.dauniv.ac.in for information about the course

4. Other learning material such as computer based programs/CD, professional standards/regulations

Assignments

1. What are various operations other than concatenation over string? Highlight your ideas.
2. When will you say that the strings are ordered? What are the relationships for ordering that may exist for string? Explain them with example.
3. Write a grammar that generates matched parenthesis i.e. legally balanced strings of left and right parenthesis.
4. What languages do the following grammars generate:
 - i) $S \rightarrow ()$
 $S \rightarrow)($
 $S \rightarrow SS$
 $S \rightarrow (S)$
 $S \rightarrow)S($
 - ii) $S \rightarrow 0$
 $S \rightarrow 1$
 $S \rightarrow S0$
 - iii) $S \rightarrow aSa \mid bSb \mid a \mid b \mid \epsilon$
 - iv) $S \rightarrow 0B \mid 1A$

$A \rightarrow 0 \mid 0S \mid 1AA$
 $B \rightarrow 1 \mid 1S \mid 0BB$

5. Give all prefixes, suffixes and substrings of abcd
6. Prove or disprove $L^+ = L^* - \{e\}$
7. Let h be the homomorphism defined by $h(0) = a$ and $h(1) = bb$ and $h(2) = \{e\}$.
What is $h(L)$ where $L = \{012\}^*$.
8. Find Context-free grammars for the following languages (with $n \geq 0, m \geq 0$):
 - i) $L = \{a^n b^m : n \leq m+3\}$
 - ii) $L = \{a^n b^m : n \neq m-1\}$
 - iii) $L = \{a^n b^m : n \neq 2m\}$
 - iv) $L = \{a^n b^m : 2n \leq m \leq 3n\}$
9. Find the reduced grammar to the grammar G whose productions are:

$S \rightarrow AB$
 $S \rightarrow CA$
 $B \rightarrow BC$
 $B \rightarrow AB$
 $A \rightarrow a$
 $C \rightarrow aB$
 $C \rightarrow b$
10. Construct a grammar equivalent to following grammar:

$S \rightarrow aAb$
 $A \rightarrow Sb \mid bcc \mid DaA$
 $C \rightarrow abb \mid DD$
 $E \rightarrow aC$
 $D \rightarrow aDA$
11. Find a derivation tree of a^*b+a^*b given that a^*b+a^*b is in $L(G)$ where G is given by

$S \rightarrow S+S \mid S^*S$
 $S \rightarrow a \mid b$
12. A context free grammar G has the following productions:

$S \rightarrow 0S0 \mid 1S1 \mid A$
 $A \rightarrow 2B3$
 $B \rightarrow 2B3 \mid 3$

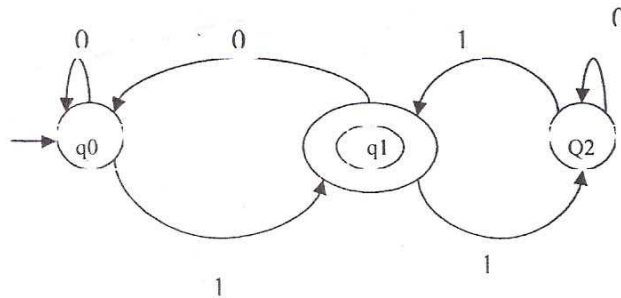
 Describe the language generated by the parameters.
13. Show that for getting an equivalent grammar in the simplest form, we have to eliminate unit productions first and then redundant symbols.
14. Reduce the following grammars to Chomsky Normal Form:
 - i)

$S \rightarrow 1A \mid 0B$
 $A \rightarrow 1AA \mid 0S \mid 0$
 $B \rightarrow 0BB \mid 1S \mid 1$
 - ii)

$S \rightarrow a \mid b \mid cSS$
 - iii)

$S \rightarrow abSb \mid a \mid aAb$
 $A \rightarrow bS \mid aAAb$
15. Is it always possible to reduce the grammar? If no verify it by an example.

16. Which of the strings 0001, 01001, 0000110 are accepted by the DFA in the following figure:



17. For $\Sigma = \{a, b\}$, construct DFAs that accept the sets consisting of:

- all strings with exactly one **a**
- all strings with at least one **a**
- all strings with not more than three **a**'s
- all strings with at least one **a** and exactly two **b**'s.
- all the strings with exactly two **a**'s and more than two **b**'s.

18. Give DFAs for the languages:

- $L = \{ ab^5wb^4 : w \in \{a, b\}^* \}$
- $L = \{ w_1abw_2 : w_1 \in \{a, b\}^*, w_2 \in \{a, b\}^* \}$

19. A run in a string is a substring of length at least two, as long as possible and consistent and entirely of the same symbol. For instance, the string **abbbaab** contains a run of **b**'s of length three and a run of **a**'s of length two. Find DFA's for the following languages on $\{a, b\}$

- $L = \{w : w \text{ contains no runs of length less than four}\}$
- $L = \{w : \text{every run of a's has length either two or three}\}$
- $L = \{w : \text{there are at most two runs of a's of length three}\}$
- $L = \{w : \text{there are exactly two runs of a's of length three}\}$

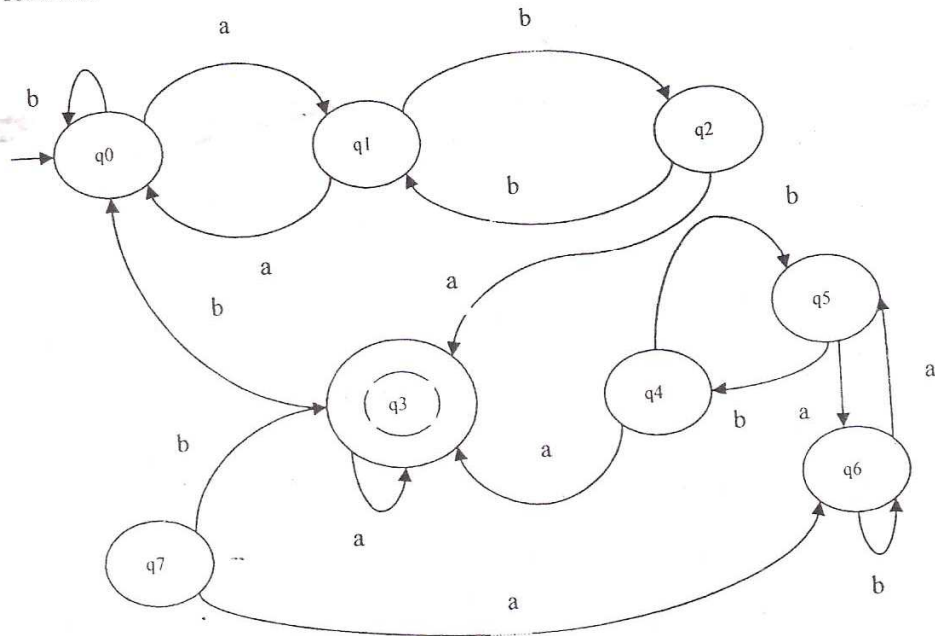
20. Construct a nondeterministic finite automaton accepting the set of all strings over $\{a, b\}$ ending in **aba**. Use it to construct a DFA accepting the same set of strings.

21. The transition table of a nondeterministic finite automaton M is given in following table:-

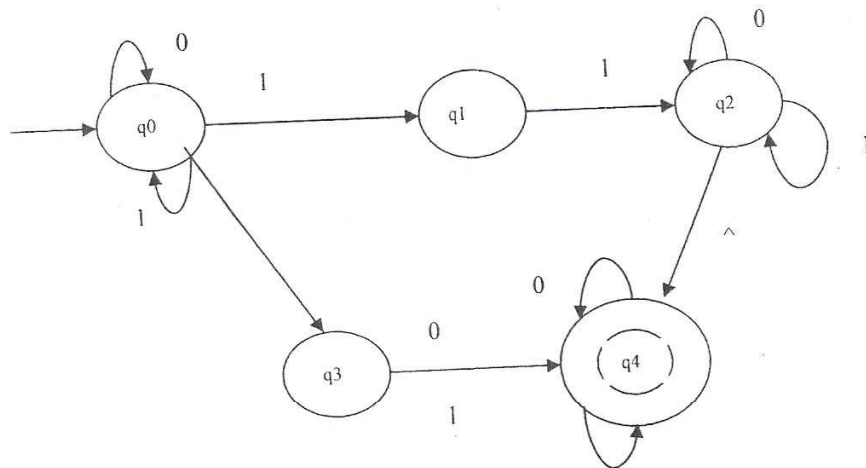
Input State	0	1	2
→ q0	q1 q4	q4	q2q3
q1		q4	
q2			q2 q3
q3		q4	
q4			

Construct an equivalent DFA to M

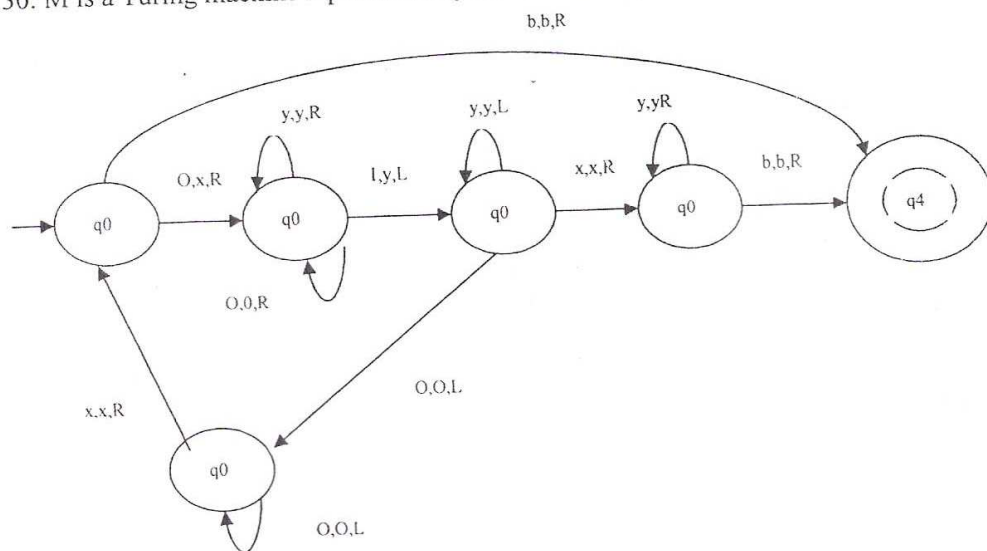
22. Construct the minimum state automaton equivalent to the transition diagram as follows:



23. Construct a DFA equivalent to the NFA in the following figure:



24. Construct a transition system which can accept strings over the alphabet a, b, \dots Containing either **cat** or **rat**.
25. Construct a Mealy machine which can output EVEN, ODD according to the total number of 1's encountered in the given string. The input symbols are 0 and 1.
26. Construct a DFA with reduced states equivalent to the regular expression: $10+(0+11+)^0*1$
27. Construct transition system equivalent to the regular expression:
- $(ab+a)^*(aa+b)$
 - $(a*b+b*a)*a$
 - $a^*+(ab+a)^*$
 - $a(a+b)^*ab$
 - $a*b+b*a$
 - $(aa+b)^*(bb+a)^*$
28. A student walks into a class room and sees on the blackboard a diagram of a TG with two states that accept only the word λ . The student reverses the direction of exactly one edge leaving all other edges and all labels all '+'s and '-'s are the same. But now the new TG accepts the language a^* . What was the original machine.
29. Construct a pda **A** accepting $L = \{WCW^R \mid W \text{ in } (a+b)^*\}$ by final state
30. **M** is a Turing machine represented by the transition system:



Lab Assignments:-

- Write a C program to generate strings up to length 15 with the indivisible symbols like $\{0,1\}$, $\{a, b\}$, $\{+,-\}$.
- Write a C program that accepts any Context Free Grammar (use appropriate data structure to store CFG) and then print it in the proper format.
- Write a program for recognition of a given string using CKY algorithm. Use a subprogram for generation of string (*) of various length. (length u to 8 and * of 0,1).

5. How do you choose scheduling strategy for the periodic, aperiodic and sporadic tasks?
6. What is the unique feature of Linux device driver ?
7. What are the OS units in an RTOS kernel ?
8. When do we use cooperative scheduling and when preemptive?
9. Compare scheduling strategies, real time scheduling, round robin mode and time slicing scheduling?
10. What are the advantages of time slice scheduling by an RTOS?
11. How does an preemption event occur?
12. Real time system performance metrics are throughput, interrupt latencies, average response times and deadline misses. Explain the importance of each of these metrics.
13. Why should we estimate worst case latency?
14. What do you mean by hierarchical RTOS?
15. Why are mobile OS popular in embedded systems for telephones and pocket PCs?
16. How is precedence assignment done for the tasks?
17. List the best strategies for synchronization between the tasks and ISRs.
18. What is dynamic program scheduling?

CS 447 Database Applications and Tools

Course Description

I Topics to be Covered		
Topics	No of Weeks	Hours
<p>Database Environment: Data versus information, traditional file processing, disadvantages, database approach, range of database application, advantages of database approach</p> <p>Exercise 1: Make a team of three or four. Choose any of the projects and submit a well documented short description. Also give the plan for making the project successful.</p>	Week 1 Chapter 1	4 hrs
Cost and risk factors, components of database environment, evolution of database system.	Week 2 Chapter 1	4 hrs
<p>Database Development Process: Information engineering, information architecture, enterprise data model, planning, SDLC, CASE etc. Steps of planning, strategic planning factors, corporate planning objects.</p>	Week-3 Chapter 2	4 hrs

<p>Developing preliminary data model, and use of planning matrices, SDLC steps, CASE role, people in database development, three-schema architecture for database development. Examples to demonstrate the development process.</p> <p>Exercise-2: For any project of your choice, develop the data development approach and prepare a report for the same, presentation will be necessary.</p>	<p>Week 4 Chapter 2</p>	<p>4 hrs</p>
<p>Modeling Data in the Organisation: Modeling of the rules of organization, data names and definitions, ER model constructs entities and its types, attributes, relationships, degree, unary, binary, ternary, n-ary, cardinalities constraints, ER modeling examples.</p>	<p>Week-5 Chapter 3</p>	<p>4 hrs</p>
<p>Mid Term Exam</p>		
<p>Enhanced ER modeling: supertype, subtypes, specialization, generalization, specifying constraints in EER models, completeness,</p>	<p>Week 6 Chapter 4</p>	<p>4 hrs</p>
<p>Disjointness, discriminators, defining super/sub type hierarchies, EER modeling examples, live demos modeling for few scenarios.</p>	<p>Week 7 Chapter 4</p>	<p>4 hrs</p>
<p>Exercise-3: For the project of your choice, describe the development of ER/EER model and document the complete conceptual design along with the presentation.</p>	<p>Week 8</p>	<p>4 hrs</p>
<p>Logical database design and relational model development, Relational model properties, keys, primary, secondary, composite, properties of relations.</p>	<p>Week-9 Chapter 5</p>	<p>4hrs</p>
<p>Codd's rules, integrity constraints, creating relational tables, Transform EER diagrams into relations, seven different steps for mapping EER model into relations,</p>	<p>Week 10 Chapter 5</p>	<p>4 hrs</p>
<p>Mid Term Exam</p>		
<p>Introduction to normalization, steps, functional dependencies, basic normal forms, definition of first, second, third normal form and removing anomalies from the relations. De-normalization and merging relations.</p>	<p>Week 11 Chapter 5</p>	<p>4 hrs</p>

Exercise-4: For the project of your choice, describe the development of normalized relational model from the ER/EER model and document the complete relational design	Week 12	4 hrs
Exercise-5: For the project of your choice, describe the SQL commands for creating all the tables, and sort and search of I/O information, using the knowledge gained from this chapter and the previous design.	Week-11 Chapter 7	4 hrs
Special Topics (Overview) : Data Warehousing, Data Mining, Distributed Databases,	Week-12 Chapter 11, 13	4 hrs
Object oriented modeling, definitions, activities in phases of model development, advantages of OOM, UML class diagrams, Example of a model development.	Week-13 Chapter 14	4 hrs
Exercise 6: Full project report preparation and presentation of the project.	Week 14	4 hrs

Learning Resources

1. Required Text(s) “Modern Database Management” Seventh Edition, Hoffer, Prescott, McFadden Pearson Education
2. Essential References <ul style="list-style-type: none"> • Database Systems ”Thomas M. Connolly, Carolyn E. Begg Pearson Education • Raghu R and Johannes G., “Database management Systems”, Mc Hill 3rd Ed 2002, ISBN 007246538 • Elmasri R, Navathe S, “Fundamentals of Database Systems”, Addison Wesley 4th Ed., ISBN 0321122267
3- Recommended Books and Reference Material <ul style="list-style-type: none"> • Database Projects 2007, Publication from the School of Computer Science
4-Electronic Materials, Web Sites etc: www.scs.dauniv.ac.in for information about the course and course material

4. Write a C/C++ program which reads a program written in any programming language (say C/C++/Java) and then perform lexical analysis. The output of program should contain the tokens i.e. classification as identifier, special symbol, delimiter, operator, keyword or string. It should also display the number of identifiers, special symbol, delimiter, operator, keyword, strings and statements.

Project Assignment

The students in the group of four members each will be given a particular programming language's assignment as:

- First phase: Find the alphabet set, set of terminals and non-terminals of the given language.
- Second phase: Derive the Context free grammar for every programming construct with parse tree

CS 423 Object Oriented Programming (Using JAVA)

CLASS SCHEDULE:

DATE	TOPIC	Chapter
Week 1	Introduction to java: Features of Java, Object-oriented programming overview, Introduction of Java Technologies, How to write simple Java programs, Data Types, Variables, Memory concepts, decision making operators, Naming Conventions	Chapter 1&2 of Text Book
Week 2	Introduction to Class, Objects, Methods and Instance Variables, Primitive type Vs Reference Type, Initializing Objects with Constructors.	Chapter 3 of text Book
Week 3	Type conversion & casting, Operators, Control Statements(if Single-Selection Statement, if-else Double Selection), while Repetition Statement, for Repetition Statement, do-while Repetition Statements, switch Multiple-Selection Statement, break and continue Statements.	Chapter 4&5 of text book
Week 4	Static Method, static field and Math Class, Method Call Stack and Activation Record, Argument Promotion and Casting, Scope of declaration and Method Overloading, String Handling: The String constructors, String operators, Character Exaction, String comparison, String Buffer.	Chpter 6 of text book chapter 13 of Java2 : The Complete Reference
Week5	Arrays: Declaring and Creating Arrays, Enhanced for Statement, Passing Arrays to Method, Multidimensional Arrays, Variable-Length Argument lists, Using Command-line Arguments.	Chapter 7 of text book

Week 6	final Instance Variables, this reference, static import, overloaded Constructors, Garbage collection and method finalize , Overloading methods, Parameter passing.	Chapter 8 of text Book
Week 7	Inheritance: Extending classes, protected Members, relationship between Superclasses and Subclasses, Using super, Constructor in Subclasses, The Object Class, Object Copying in Java	Chapter9 of text book and chapter6 of Core Java 1.2 Volume I
Week8	Polymorphism: Method overriding, upcasting, Dynamic Method Dispatch, final Method and classes, Abstract classes and Methods, instanceof operator, Downcasting, Class class, Runtime type Identification	Chapter 10 of Text book and chapter 5 of Core Java 1.2 Volume
Week 9	Packages and Interfaces: Defining a Package, Understanding CLASSPATH, Access Protection, Importing packages, creating own packages. Defining an Interface, Properties of interface, advantages of interface Achieving multiple inheritance through interfaces, Variables in Interfaces, Comparable interface.	Chapter 9 of Java2 : The Complete Reference and Chapter 6 Core Java 1.2 Volume I
Week 10	Exception Handling: Introduction, overview of doing it and keywords used, when to use it, Java Exception Hierarchy, finally block, chained exceptions, declaring new exception types, preconditions and postconditions.	Chapter 13 of text book
Week11	Streams and Files: Introduction, Data Hierarchy, Files and Streams, Sequential-access Text Files, Object Serialization, Random-Access files, Java Stream class Hierarchy.	Chapter 14 of text book
Week 12	Multithreading: What are threads, The java thread model, Thread priorities, Thread life cycle, Creating thread and executing thread, Thread Synchronization, producer-consumer problem without Synchronization	Chapter 23 of text book
Week 13	Multithreading cont....: Producer-consumer problem with Synchronization, Other class and Interfaces in java.util.concurrent, Monitor and Monitor Locks, Thread Groups, Synchronization, Inter-thread Communication.	Chapter 23 of text book
Week 14	Introduction To GUI : Introduction, Overview of swing Components, Displaying text and Images in a window, Introduction to Event Handling, Common GUI Event Type and Listener Interfaces, How Event Handling Works, Adapter Classes, Layout Managers	Chapter 11 of text book
Week15	Applets: Applet basics, Applet Architecture, Applet life cycle methods, Applet HTML Tag and attributes,	Chapter 20 of text book

	Executing applet in web browser and in the appletviewer, in Passing parameters to Applets, doing GUI programming in applet	
Week16	Generic and Collection API: Introduction, Motivation for Generic Methods, Generic Methods : Implementation and Compile- time Translation Issues, Overloading Generic Methods, Generic Classes, Raw Types, Generic and Inheritance	Chapter 18 of text book
Week 17	Database connectivity: JDBC, The design of JDBC, Typical uses of JDBC, The Structured Query language, Basic JDBC Programming concepts, Executing Queries.	Lecture Notes

Learning Resources

Required Text(s)	<ul style="list-style-type: none"> JAVA How to Program by Deitel & Deitel, Pearson Education, Seventh Edition, 2007
1. Essential References	<ul style="list-style-type: none"> Java2 : The Complete Reference by Herbert Schildt, Tata McGraw- Hill, fourth Edition, 2005 Thinking in Java by Bruce Eckel , Prentice Hall, Third Edition, 2005 Core Java 1.2: Volume 1 Fundamentals by Gary Cornell, Cay Horstmann, Prentice Hall, Seventh Edition, 2007
3- Recommended Books and Reference Material (Journals, Reports, etc) (Attach List)	JAVA How to Program by Deitel & Deitel, Pearson Education, Seventh Edition, 2007
4-.Electronic Materials, Web Sites etc	www.sun.java.com www.wrox.com math.hws.edu/javanotes/ http://www.apl.jhu.edu/~hall/java/

List of Experiment and Assignments:

S. No.	Excercises
Program List 1 :	
Submission : end of week 1	
1.	Write a program that produces the following output: Hello World!

Submission : end of week 3

1.	Write an application that uses <i>String</i> method <i>compareTo</i> to compare two strings defined by the user.
2.	Write an application that uses <i>String</i> method <i>equals</i> and <i>equalsIgnoreCase</i> to tests any two string objects for equality.
3.	Write an application that uses <i>String</i> method <i>indexOf</i> to determine the total number of occurrences of any given alphabet in a defined text.
4.	Write an application that uses <i>String</i> method <i>concat</i> to concatenate two defined strings.
5.	Write an application that finds the length of a given string.
6.	Write an application that uses <i>String</i> method <i>charAt</i> to reverse the string.
7.	Write an application that finds the substring from any given string using <i>substring</i> method and <i>startsWith</i> & <i>endsWith</i> methods.
8.	Write an application that changes any given string with uppercase letters, displays it , changes it back to lowercase letters and displays it.

Program List 4 :**Submission : end of week 4**

1.	Create a class called <i>Employee</i> that includes three pieces of information as instance variables – a first name (type <i>String</i>), a last name (type <i>String</i>) and a monthly salary (double)
2.	Create a constructor in above class to initialize the three instance variables. Provide a <i>get</i> method for each instance variable.
3.	Write a test application named <i>EmployeeTest</i> that demonstrates class <i>Employee</i> 's capabilities. Create two employee objects and display each object's yearly salary.
4.	Give each employee a 10% raise and display each <i>Employee</i> 's yearly salary again.
5.	Create a class <i>Account</i> with an instance variable <i>balance</i> (double). It should contain a constructor that initializes the <i>balance</i> , ensure that the initial balance is greater than 0.0.
6.	Create two methods namely <i>credit</i> and <i>getBalance</i> . The first one adds the amount (passed as parameter) to balance and does not return any data. The second method allows clients (i.e. the other classes that use this class) to obtain the value of a particular <i>Account</i> object's <i>balance</i> .
7.	Create class <i>AccountTest</i> to create and manipulate an <i>Account</i> object.
8.	Write another method <i>debit</i> in the above program that withdraws money from an <i>Account</i> . Ensure that the debit amount does not exceed the <i>Account</i> 's <i>balance</i> . In that case the <i>balance</i> should be left unchanged and the method should print a message indicating "Debit amount exceeded account balance". Modify class <i>AccountTest</i> to test method <i>debit</i> .
9.	Write an application that reads a five digit integer and determine whether it is a palindrome (digit that reads the same backward and forward eg. 12321, 45554 etc.) . display an error message, if the number is no5 five digits long and allow the user to enter a new value.
10.	Write an application that reads three nonzero value entered by the user and determines and prints sum, product, average, smallest & largest of three.

11.	Write an application that prompts the user for the radius of a circle and uses a method called circleArea to calculate the area of the circle.
12.	Add another method in the above program circlePerimeter to calculate the perimeter of the circle.
Program List 5 :	
Submission : end of week 6	
1.	Write an application to create a super class Employee with information first name & last name and methods getFirstName(), getLastName() derive the sub-classes ContractEmployee and RegularEmployee with the information about department, designation & method displayFullName() , getDepartment, getDesig() to print the salary and to set department name & designation of the corresponding sub-class objects respectively.
2.	Derive sub-classes of ContractEmployee namely HourlyEmployee & WeeklyEmployee with information number of hours & wages per hour, number of weeks & wages per week respectively & method calculateWages() to calculate their monthly salary. Also override getDesig() method depending on the type of contract employee.
3.	Write an application to create a super class Vehicle with information vehicle number,insurance number,color and methods getConsumption() and displayConsumption(). Derive the sub-classes TwoWheeler and FourWheeler with method maintenance() and average() to print the maintenance And average of vehicle.
4.	Extend the above TwoWheeler class with methods getType() and getName() which gives the information about the type and the name of the company.Create sub-classes Geared and NonGeared with method average() to print the average of a geared and non-geared two wheeler.
5.	Create a super class CommunityMember with the information of member i.e. name,address, contact, date_of_join through methods getName(),getAddress(),getContact(),getDate_of_Join() and derive sub-classes Employee and Student with method Qualification() to print the related information with his/her qualification.
6.	Create a super class Shape with methods getName() which gives the information about the type of the shape.derive its sub-classes TwoDim and ThreeDim with method area() and volume() respectively which prints the area and volume of a two-dimensional and three-dimensional shape.
7.	Extend the class TwoDim with methods getLength(),getBreadth() which displays the length and breadth of two dimensional shapes.Derive sub-classes rectangle,rhombus with method getArea() and getPerimeter() to calculate the area and perimeter of this two dimensional shapes. .
8.	Extend the class ThreeDim with methods getLength(),getBreadth(),getHeight() which displays the length , breadth and height of three dimensional shapes.Derive sub-classes cuboid,tetrahedron with method getArea() and getVolume() to calculate the area and volume of this three dimensional shapes. .

9.	Create a super class Student with methods getQual (), getFirstName(),getLastName(), getAddress(), getContat(), which gives basic details of student.derive sub-classes Faculty and Scholar with method salary(), Course() resp. which gives the additional information about the salary and course of faculty and scholar resp. .
----	---

Program List 6 :

Submission : end of week 8

1.	Create an abstract class Shape which calculate the area and volume of 2-d and 3-d shapes with methods getArea and getVolume. Reuse this class to calculate the area and volume of square ,circle ,cube and sphere.
2.	Create an abstract class Employee with methods getAmount() which displays the amount paid to employee. Reuse this class to calculate the amount to be paid to WeeklyEmployee and HourlyEmployee according to no. of hours and total hours for HourlyEmployee and no. of weeks and total weeks for WeeklyEmployee.
3.	Create an Interface payable with method getAmount ().Calculate the amount to be paid to Invoice and Employee by implementing Interface.
4.	Create an Interface Vehicle with method getColor(),getNumber(),getConsumption(). Calculate the fuel consumed, name and color for TwoWheeler and FourWheeler by implementing interface Vehicle.
5.	Create an Interface Fare with method getAmount() to get the amount paid for fare of travelling. Calculate the fare paid by bus and train implementing interface Fare.
6.	Create an Interface StudentFee with method getAmount(), getFirstName(),getLastName(), getAddress(), getContact(). Calculate the amount paid by the Hostler and NonHostler student by implementing interface StudentFee
7.	WAP to create your own package. Package should have more than two classes. Write a class that uses the package.
8.	Create a package named org.shapes. Create some classes in the package representing some common geometric shapes like Square, Triangle, Circle and so on.

EXCETION HANDLING QUESTIONS:

Program List 7 :

Submission : end of week 9

1.	Exception Handling program for division of two numbers that accepts numbers from user.
2.	Exception Handling program for storing values in array of <i>int</i> or <i>String</i> that results into buffer overflow
3.	Exception Handling program for calculating roots of quadratic equation that accepts coefficients from user.

4.	Exception Handling program for <i>NullPointerException</i> --thrown if the JVM attempts to perform an operation on an <i>Object</i> that points to no data, or <i>null</i>
5.	Exception Handling program for <i>NumberFormatException</i> --thrown if a program is attempting to convert a string to a numerical datatype, and the string contains inappropriate characters (i.e. 'z' or 'Q')
6.	Exception Handling program for <i>ClassNotFoundException</i> --thrown if a program can not find a class it depends at runtime (i.e., the class's ".class" file cannot be found or was removed from the CLASSPATH)
7.	Exception Handling program for <i>IOException</i> --actually contained in <i>java.io</i> , but it is thrown if the JVM failed to open an I/O stream
8.	Write a program that shows that the order of the catch blocks is important. If you try to catch a superclass exception type before a subclass type, the compiler should generate errors.
9.	Program for demonstrating the use of throw, throws & finally - Create a class with a main() that throws an object of class Exception inside a try block. Give the constructor for Exception a String argument. Catch the exception inside a catch clause and print the String argument. Add a finally clause and print a message to prove you were there.
10.	Create your own exception class using the extends keyword. Write a constructor for this class that takes a String argument and stores it inside the object with a String reference. Write a method that prints out the stored String . Create a try-catch clause to exercise your new exception.
11.	Write a program to rethrow an exception - Define methods one() & two(). Method two() should initially throw an exception. Method one() should call two(), catch the exception and rethrow it Call one() from main() and catch the rethrown exception.
Program List 8 :	
Submission : end of week 10	
1.	Write a program to change the priority of thread.
2.	WAP for producer consumer problem (w/o synchronization).
3.	WAP for producer consumer problem (with synchronization).
4.	Create an application of cash withdrawal from the bank account that have no. of users that are operating the accounts.(synchronization)
Program List 9 :	
Submission : end of week 11	
1.	Open a text file so that you can read the file one line at a time. Read each line as a String and send the results to System.out .
2.	Modify Exercise 1 so that the name of the file you read is provided as a command-line argument.
3.	Modify Exercise 2 to force all the lines in the results to upper case and send

	the results to System.out
4.	Modify Exercise 2 to also open a text file so you can write text into it.
5.	Implement a pair of classes, one <code>Reader</code> and one <code>Writer</code> , that count the number of times a particular character, such as <code>e</code> , is read or written. The character can be specified when the stream is created.
6	Construct a program <code>Wc</code> ("word count"), which counts number of chars, words and lines of the text file. Space is counted as a character. Empty rows are counted as lines. "Word" will represent a string.

Program List 10 :

Submission : end of week 12

1.	Create an application to display a frame with title <code>MyFrame</code> .
2.	Create an application to draw a horizontal line.
3.	Create an application to draw one line perpendicular to other. One line parallel to other.
4.	Create an application to display a circle within rectangle
5.	In the above application fill different colors in the circle & rectangle.
6.	Write an application that displays any string. Choose color from combo box to change the color of this displayed string and choose its size & type respectively from another two combo boxes.
7.	Write a small application with a default date 01/01/2000 and three combo boxes displaying valid days, months & year (1990 – 2050). Change the displayed date with the one chosen by user from these combo boxes.
8.	Create a GUI with a text field and three buttons. When you press each button, make some different text appear in the text field.
9.	Create a GUI application to take input of two numbers(text field) from user. When you press button it should display sum of the two numbers in a third text box.
10.	Create an applet with a <code>Button</code> and a <code>TextField</code> . Write a <code>referenceEvent()</code> so that if the button has the focus, characters typed into it will appear in the <code>TextField</code> .
11.	Write an application to create a GUI with two buttons such that clicking on the first displays the message "Welcome to SCS" on the window and clicking on the second changes the color of the message(<i>hint : toggle the color</i>)
12.	Create a GUI with title <code>STUDENT</code> which has labels roll no., name, class, address with textboxes for taking input from the user(without any functionality).
13.	Create a GUI application for fees receipt which contains checkboxes for selecting the course, radio buttons for selecting gender and labels and corresponding textboxes for name, class, date and amount paid.
14.	Create a GUI application to display a calculator using grid Layout (You do not have to provide functionality).

Program List 11 :

Submission : end of week 14

1.	WAP for string tokenizer.
2.	Create a program that will print every other argument given on the command line. (Use of string tokenizer), consider how your program will deal with no argument.
3.	WAP that generate a random number (1 – 10000). Let the user guess the correct number. User will enter the digit. Program should let the user that input is right or wrong. No of turns that user can make a choice for input is twice the number of digits in the system generated numbers.
4.	Convert the input date in words. Input format is dd mm yy.
5.	Find the frequency of each number in the array.

CS-4517 UNIX /LINUX ADMINISTRATION

Course Description (Note: General description in the form to be used for the Bulletin or Handbook should be attached)

Week	TOPIC	REA DIN G
Week 1	Background: Evolution of Unix OS. Unix implementations. Features of Unix operating system. Assignment 1	A: Ch2
Week 2	Linux operating system: Development of Linux. Applications of Linux operating system. Assignment 2	A: Ch2
Week 3	Basic unix environment: Basic commands, directory management, pipes, tee, I/O redirection and other utilities. Assignment 3	A: Ch3
Week 4	Advanced commands: File system and process management commands, Shell, Pattern matching, Navigating the File Systems. Assignment 4	A: Ch4,5
Week 5	Unix editor: VI editor, Creating new files. Text addition, deletion and changes. Dealing with sentences and paragraphs. Searching. Cut, paste and copy. Running C/C++ programs. Assignment 5	A: Ch8
Week 6	Shell programming: Features of shell. Shell variables. Control statements. Assignment 6	A: Ch8
Week 7	Advance shell programming: Command line arguments. Interactive shell scripts. Debugging of shell scripts. Communication facilities in Unix. Assignment 7	A: Ch 9,Ch15
Week 8	Structure of unix operating system: Structure of unix kernel, Unix system calls. Assignment 8	C: Ch4
Week 9	Unix system: File system calls, Process management calls. Assignment 9	C: Ch5
Week 10	Advance Filter Awk: Number processing, Interface with shell, functions. Assignment 10	A: Ch20

Week 11	Unix system administration: Adding and removing users. User accounting. Adding and removing hardware. Performing backups and restore. Disk space management. Assignment 11	B: Ch 6
Week 12	Unix system administration: Configuring the kernel. Network management in Unix. Performance analysis. Assignment 12	B: Ch13
Week 13	Documents Preparations: The ms macro package, Troff level, tbl & eqn expression. Assignment 13	A: Ch26
Week 14	Conclusion: Test, Laboratory Viva and Revision.	

A: **Unix Operating Systems:** Sumitabh Das, Tata McGraw Hills publication.

B: **Unix System Administration Handbook** (Second edition): Evi Nemeth, Garth Synder, Scott Seebass, Trent R Hein, Pearson Education - Asia, 2000.

C: **Design of Unix Operating System:** Maurice J. Back, Pearson Education - Asia.

Subject Learning Outcomes

<p>Development of Learning Outcomes in Domains of Learning:</p> <p>Developing applications in Unix environment and administrating Unix OS.</p> <ol style="list-style-type: none"> 1. Understanding will be devolved about various OS and usage. 2. Basic commands to use OS. 3. Understanding with file systems. 4. Learns working in editors. 5. Concepts of shell programming and system call will be developed. 6. Understanding of communication facilities used in UNIX. 7. Practicing administrative commands.

Learning Resources

<p>I-Required Text(s)</p> <p>A: Unix Operating Systems: Sumitabh Das, Tata McGraw Hills publication.</p> <p>B: Unix System Administration Handbook (Second edition): Evi Nemeth, Garth Synder, Scott Seebass, Trent R Hein, Pearson Education - Asia, 2000.</p> <p>C: Design of Unix Operating System: Maurice J. Back, Pearson</p>

Education - Asia.
2-Essential References(s) 1. UNIX: Concepts and Applications,4E By Sumitabh Das, Tata McGraw Hills publication 2. UNIX: Ultimate guide By Sumitabh Das, Tata McGraw Hills publication
3-Recommended Books and Reference Material (Journals, Reports, etc) (Attach List)
4-Electronic Materials, Web Sites etc www.iitk.ac.in ,
5- Other learning material such as computer-based programs/CD, professional standards/regulations

List of Assignments

1. Short description on various OS available and Comparison.
2. Security issues faced in using OS used by you.
3. Description on Windows OS compares the commands with Unix.
4. Describe installation process of Unix/Linux.
5. Explain the Unix & windows file system and compare.
6. Explain Editors and its Usage.
7. Write shell program to calculate interest, area, prime number.
8. Describe role of Administrator.
9. Describe backup & restore technique used for large data.
10. Design LAN for organization
11. Explain to enable DNS, DHCP server.
12. Explain to protect unpatched Samba server.
13. Write study report on IBM Storage Area Network (SAN).

CS – 5178 Internet and Web Technology

Aim and Objectives

Course Description (Note: General description in the form to be used for the Bulletin or Handbook should be attached)

1 Topics to be Covered		
Week	TOPIC	READING
Week 1	Introduction Dynamic Web Programming, HTML Forms, scripting languages, Introduction to HTTP, web Server and application Servers, Installation of Application servers, Configuration files, Web.xml.	Lecture notes

Week 2	Java Servlet, Servlet Development Process, Deployment Descriptors, The Generic Servlet Lifecycle.	Chap. 5
Week 3	Servlet Packages, Classes, Interfaces, and Methods, Handling Forms with Servlets.	Chap. 5
Week 4	Various methods of Session Handling. Various elements of deployment descriptors.	Chap. 5
Week 5	Java Database Connectivity: various steps in process of connection to the database, Various type of JDBC Driver.	Chap. 6
Week 6	Connection of JSP and Servlet with different database viz. Oracle, MS-SQL Server, MySQL. java.sql Package. Accessing metadata from the database.	Chap. 6 and lecture notes
Week 7	Type of Statement, Connection pooling: multiple users and need of connection pooling.	Chap. 7 and lecture notes
Week 8	JSP Basics: JSP lifecycle, Directives, scripting elements, standard actions, implicit objects. Writing JSPs.	Chap. 3
Week 9	Expression Language (EL), Separating Business Logic and Presentation Logic, Building and using JavaBean.	Chap. 3
Week 10	Session handling in JSP, Types of errors and exceptions handling, Standard Tag Library in JSP, Building Custom Tag Library	Chap. 3
Week 11	Hibernate, MVC Design pattern	Chap. 8 and lecture notes
Week 12	Struts framework	Chap. 8 and lecture notes

Subject Learning Outcomes

Development of Learning Outcomes in Domains of Learning

The course objectives are to:

To demonstrate the application of web programming.

To design and develop web application using Servlet and JSP.

To show the applicability of various J2EE frameworks to the web applications.

Based upon above objectives the course goals / learning outcomes are defined below:

1. Identify the need of dynamic website programming. Understand the concept of

Web servers and Application servers and Application of configuration files. Gain the idea of various dynamic web programming technology.

2. Understand the servlet model for the dynamic web programming. Developing and deploying the basic servlet application. Understand the lifecycle of the servlet and the various classes from servlet package provided in the API.
3. Understand HTML form handling using servlets. Understand the concept of session handling. To get detail understanding of the various session handling techniques. Understand the idea of deployment descriptor and detail about the various elements of deployment descriptor.
4. Understand the need of database programming for dynamic website designing. Develop programmes using various JDBC driver types and the SQL package from the JDBC API. Also develop programme to access metadata information.
5. Understand various types of Statement classes available in JDBC. Understand the concept and applicability of connection pooling.
6. Understand the basics of JSP viz. lifecycle of JSP, various scarping elements of JSP. Developing JSP program. Understand better designing concept of Web application using JavaBeans.
7. To get detail understanding of the various session handling techniques. Understand and use the standard tag library of JSP. Developing the custom Tag Library.
8. Understand the need and concept of Hibernate. Develop the J2EE application using the Hibernate. Understand the concept of MVC design pattern and develop the web application under the MVC design pattern using Struts.

Learning Resources

1. Required Text(s)

1. Kevin Mukhar, Chris Zelenak, James L Weaver, "Beginning Java EE 5: From Novice to Professional" Apress

2. Essential References:

1. Marty Hall, Larry Brown, "Core Servlets and Java Server Pages", 2nd edition, Pearson Education

2. JavaDoc for various technologies

<p>3- Recommended Books and Reference Material (Journals, Reports, etc):</p> <ol style="list-style-type: none"> 1. S. Allamaraju, "Professional Java Server Programming", Wrox Press 2. G. Franciscus, "Struts Recipes", Manning Press 3. C. Bauer, G. King, "Hibernate in Action", Manning Press
<p>4-.Electronic Materials, Web Sites etc</p> <p>www.sun.java.com</p> <p>www.coreservlets.com</p> <p>http://www.javaranch.com</p>
<p>5- Other learning material such as computer-based programs/CD, professional standards/regulations</p> <p>None</p>

Assignment

1 Topics to be Covered		
Week	TOPIC	READING
Week 1	Introduction Dynamic Web Programming, HTML Forms, scripting languages, Introduction to HTTP, web Server and application Servers, Installation of Application servers, Configuration files, Web.xml.	Lecture notes
Week 2	<p>Java Servlet, Servlet Development Process, Deployment Descriptors, The Generic Servlet Lifecycle.</p> <p>Lab Assignments:</p> <ol style="list-style-type: none"> 1. Write a servlet that prints "Hello World" 2. Write a servlet that knows to whom it's saying hello, This servlet must be called from an HTML page taking user name as input. (Use both get and post method) 3. Write a servlet that counts and displays the number of times it has been accessed since the last server reboot. 4. Write a servlet that counts the times it has been accessed, the number of instances created by the server, and the total times all of them have been accessed. 5. Write a servlet that counts and displays the number of times it has been accessed, and reads an init parameter to know what at what number to begin counting. 6. This servlet counts and displays the number of times it has been accessed, and saves the count to a file in its destroy() method to make the count persistent. 7. Write a servlet that searches for prime numbers above one 	Chap. 5

	<p>quadrillion. The algorithm it uses couldn't be simpler: it selects odd-numbered candidates and attempts to divide them by every odd integer between 3 and their square root. If none of the integers evenly divides the candidate, it is declared prime. It's disabled to let the server's CPU handle important tasks.</p> <ol style="list-style-type: none"> 8. Write a servlet that prints the name and value for all of its init parameters. 9. Write a servlet that displays information about its server (The process is called Snooping). 10. Write a servlet that snoops the server's servlet and Java version. 	
<p>Week 3</p>	<p>Servlet Packages, Classes, Interfaces, and Methods, Handling Forms with Servlets. Lab Assignments:</p> <ol style="list-style-type: none"> 11. Write a servlet to compute arithmetic operations on numbers and strings as follows: <div style="margin-left: 40px;"> <p>First val: <input type="text"/> Enter a name: <input type="text"/></p> <p>Second val: <input type="text"/> Length: <input type="text"/></p> <p>Result: <input type="text"/></p> <p> <input type="button" value="Add"/> <input type="button" value="Subtract"/> <input type="button" value="Multiply"/> <input type="button" value="Divide"/> <input type="button" value="Compute Length"/> </p> <p style="text-align: center;"><input type="button" value="Reset"/></p> </div> <ol style="list-style-type: none"> 12. Write a servlet that checks the client machine and only allows access if the client appears to be coming from somewhere other than the Terrorist 7 countries. A servlet like this could help restrict the export of strong encryption. 13. Write a servlet that prints its query string and then prints the name and value for all its parameters. 14. Write a servlet that prints the extra path information it receives and the resulting translation to a real path. 15. Write a servlet that serves files by using the <code>getPathTranslated()</code> and <code>getMimeType()</code> methods to return whatever file is given by the extra path information. 16. Write a servlet that prints information about its HTTP request headers. 17. Write a servlet that performs a random redirect, sending a client to a random site selected from its site list. Depending on the site list, a servlet like this could have many uses. As it stands now, it's just a jump-off point to a selectino of cool servlet sites. With a site list containing advertising images, it can be used to select the next ad banner. 	<p>Chap. 5</p>

	<p>18. Redirections can be used to learn where clients go when they leave your site. Assume you have several pages containing lists of links to other sites. Instead of linking directly to the external site, you can link to a redirecting servlet that can record each time an external link is selected. The HTML looks like this: <code>Servlets.com</code> Write a servlet that can be registered to handle the /goto/* path prefix where it will receive the selected URL as extra path info and redirect the client to that location after making a note in the server log. servlet shows a servlet that uses client pull to display the current time, updated every 10 seconds.</p> <p>19. Write a servlet that shows a servlet that uses client pull to display the current time, updated every 10 seconds.</p> <p>20. Write a servlet that redirects requests for one host to another host, giving an explanation to the client before the redirection.</p> <p>21. Write a servlet that demonstrates session tracking using hidden form fields by displaying the shopping cart for a booksite.</p>	
<p>Week 4</p>	<p>Various methods of Session Handling. Various elements of deployment descriptors. Lab Assignments:</p> <p>22. Write a servlet that uses session tracking to count the number of times a client has accessed it.</p> <p>23. Write a servlet that demonstrates how to programmatically alter the current timeout. On first execution, the current timeout displays the application-wide setting. On second execution, the current timeout displays two hours—because that's the timeout set during the first execution.</p> <p>24. Write a servlet that snoops all the information about the current session.</p> <p>25. Write a servlet that is protected by BASIC authentication as shown in web.xml and tomcat-users.xml. To see the salary information you'll need to login as a "manager" using names and passwords in tomcat-users.xml.</p> <p>26. Write a servlet that lets a user vote for his favorite food from a combo box or radio buttons (the user must be able to make multiple food selections per request).</p> <ul style="list-style-type: none"> • Store the favorite foods and the number of votes for each food. 	<p>Chap. 5</p>

	<ul style="list-style-type: none"> • Display all foods and their number of votes in alphabetical order back to the user. • Use an appropriate Collection class or Map class to store the data. 	
Week 5	Java Database Connectivity: various steps in process of connection to the database, Various type of JDBC Driver	Chap. 6
Week 6	<p>Connection of JSP and Servlet with different database viz. Oracle, MS-SQL Server, MySQL. java.sql Package. Accessing metadata from the database.</p> <p>Lab Assignments:</p> <p>27. Write a servlet that establishes a database connection using the values stored within the its sql.properties file having the following contents.</p> <pre> connection.driver=sun.jdbc.odbc.JdbcOdbcDriver connection.url=jdbc:odbc:somedb user=user password=passwd </pre> <p>28. Write a servlet that uses the Oracle JDBC driver to perform a simple query, printing names and phone numbers for all employees listed in a database table. Here assume that the database contains a table named EMPLOYEES, with at least two fields, NAME and PHONE.</p>	Chap. 6 and lecture notes
Week 7	Type of Statement, Connection pooling: multiple users and need of connection pooling.	Chap. 7 and lecture notes
Week 8	<p>JSP Basics: JSP lifecycle, Directives, scripting elements, standard actions, implicit objects. Writing JSPs.</p> <p>Lab Assignments:</p> <p>29. Write a JSP to output the values returned by <code>System.getProperty</code> for various system properties such as <code>java.version</code>, <code>java.home</code>, <code>os.name</code>, <code>user.name</code>, <code>user.home</code>, <code>user.dir</code></p> <p>30. Write a JSP to output the entire line, "Hello! The time is now ..." but use a scriptlet for the complete string, including the HTML tags.</p> <p>31. Write a JSP to output all the values returned by <code>System.getProperties</code> with "
" embedded after each property name and value. Do not output the "
" using the "out" variable.</p> <p>32. Modify exercise # 28-30 to import the <code>java.util</code> packages.</p> <p>33. Write a JSP to do either a <code>forward</code> or an <code>include</code>, depending upon a boolean variable.</p>	Chap. 3

	<p>34. Write a JSP/HTML set that allows a user to enter the name of a system property, and then displays the value returned by <code>System.getProperty</code> for that property name (handle errors appropriately.)</p> <p>35. Make a JSP page that randomly selects a background color for each request. Just choose at random among a small set of predefined colors. Be sure you do not use the <code>JSP-Styles.css</code> style sheet, since it overrides the colors given by <code><BODY BGCOLOR="..."></code>.</p> <p>36. Make a JSP page that lets the user supply a request parameter indicating the back-ground color. If no parameter is supplied, a background color should be selected at random.</p>	
<p>Week 9</p>	<p>Expression Language (EL), Separating Business Logic and Presentation Logic, Building and using JavaBean.</p> <p>Lab Assignments:</p> <p>37. Make a JSP page that lets the user supply a request parameter indicating the back-ground color. If no parameter is supplied, the most recently used background color (from a previous request by any user) should be used.</p> <p>38. The <code>java.math</code> package has a class called <code>BigInteger</code> that lets you create whole numbers with an arbitrary number of digits. Create a JSP page that makes a large <code>BigInteger</code> from a <code>String</code> you supply as a request parameter, squares it, and prints out the result. Use the online API at http://java.sun.com/j2se/1.5.0/docs/api/ to see the syntax for the <code>BigInteger</code> constructor and squaring operations.</p> <p>39. Make an HTML "signature" block with your name and email address. Include it in two JSP pages.</p> <p>40. The value of the page attribute of <code>jsp:include</code> is allowed to be a JSP expression. Use this idea to make a JSP page that includes a "good news" page or a "bad news" message at random.</p> <p>41. Suppose that you have two different JSP pages that do two different things. However, for both pages you want to let the user supply a <code>bgColor</code> attribute to set the background color of the page. Implement this, but use an include mechanism to avoid repeating code. For example:</p> <p style="padding-left: 40px;">White background: <code>http://host/path/page1.jsp</code> White background: <code>http://host/path/page2.jsp</code> Red background: <code>http://host/path/page1.jsp?bgColor=RED</code> Yellow background: <code>http://host/path/page2.jsp?bgColor=YELLOW</code></p> <p>For testing, I do not care if you write an HTML form to collect</p>	<p>Chap. 3</p>

	the bgColor parameter or if you simply attach it onto the end of the URL "by hand."	
Week 10	<p>Session handling in JSP, Types of errors and exceptions handling, Standard Tag Library in JSP, Building Custom Tag Library</p> <p>Lab Assignments:</p> <p>42. Make two separate JSP pages that have bulleted lists containing random integers in a certain range. Avoid repeating code unnecessarily by including a page that defines a randomInt method.</p> <p>43. Define a class called ColorBean that stores strings representing a foreground color and a background color. Compile and test it separately (i.e., without using a servlet or JSP page). Note: if your tester class (i.e., the one that has "public static void main(String[] args) {...}" in it) is in a package, remember that you have to use the package name when you run it from the command line. That is, you have to do "javac BeanTester.java" and then "java yourPackage.BeanTester".</p> <p>44. Make a "color preference" form that collects the user's preferred foreground and background colors. Send the data to a JSP page that displays some message using those colors. This JSP page should use a default value for any form value that the user fails to supply (but don't worry about empty strings). So, for example, if the user goes directly to the JSP page (bypassing the form), the JSP page should still work fine. For now, don't worry about the user sending you whitespace; just handle totally missing values.</p> <p>45. Redo the color preference example, but if the user fails to supply either of the colors, use whatever value they gave last time. If you have no previous value, use a default. (Hint: this problem is almost exactly the same difficulty as the previous one.)</p> <p>46. Redo the color preference example, but if the user fails to supply any of the parameters, use whatever color the most recent user gave last time. Why could this give bad results?</p> <p>47. Write a JSP that takes the user's name and age from a form.</p> <ul style="list-style-type: none"> • Echo back the name and age along with a message stating the price of movie tickets. • The price is determined by the age passed to the JSP. • If the age is greater than 62, the movie ticket price is 	Chap: 3

\$7.00.

- If the user is less than 10 years old, the price is \$5.00.
- For everyone else, the price is \$9.50.

48. Write a JSP that will allow a user to enter two values, select a type of mathematical operation to apply against them, and then upon clicking Submit, will display the result of the operation. An example of the default entry form:

49.

Enter two values, choose an operation, and click s

5
5
<input type="radio"/> Addition
<input type="radio"/> Subtraction
<input checked="" type="radio"/> Multiplication
<input type="radio"/> Division
<input type="submit" value="Submit"/>

And the page that displays upon Submit:

5 times 5 equals 25

A valid code must also have:

- one jsp *only* - Math.jsp - that submits to itself
- four types of operations - add, subtract, multiply, divide
- display some kind of error if the user attempts to divide by 0

50. Create a database table to store contact information, then write a web application to manage viewing, adding, and deleting contacts from that table. The view should look similar to:

A list of all contacts.

action	id	first name	last name	phone number	city
Delete	2	Laura	Juliano	555-1212	Urbanda
Delete	3	Bob	Jones	555-1212	Des Moi

Insert a new Contact

Some items of note from the view:

1. There is a delete link next to each record which triggers a delete servlet and will delete that record in the database before redirecting back to the view
 2. There is an 'Insert a new Contact' link at the bottom of the page that directs the user to ContactsInsert.html
- The insert page should look similar to:

Insert a contact

First Name:

Last Name:

Phone Number:

City:

State:

Zip:

An item to note about insert: once the user clicks submit, the form directs to a ContactsInsertServlet.java, which insert a record into the database then redirects to the view.

A description of each file needed for this exercise:

1. Write a sql script that creates a Contacts table in derby that holds text values for id, firstName, lastName, phoneNumber, city, state, and zip. Add to the script a couple of INSERT statements so that the table will not be empty. Run your script in Squirrel to confirm it works, and so you have the table on hand for the rest of the exercise. Save the script in the root of the web application as Contacts.sql.

2. ContactsViewServlet.java - displays an html table

	<p>containing all records in the table. Also displays links to ContactsDeleteServlet.java and ContactsInsert.html.</p> <p>3. ContactsInsert.html - have text fields for each field in the database. On submit, execution moves to ContactsInsertServlet.java.</p> <p>4. ContactsInsertServlet.java - uses an INSERT statement to add a record to the database. After the INSERT has been performed, redirect back to ContactsViewServlet.java.</p> <p>5. ContactsDeleteServlet.java - uses a DELETE statement to records from the database. After the DELETE has been performed, redirect back to ContactsViewServlet.java.</p> <p>6. web.xml - must have valid references to ContactsViewServlet.java, ContactsInsertServlet.java, and ContactsDeleteServlet.java.</p>	
Week 11	Hibernate, MVC Design pattern	lecture notes
Week 12	Struts framework	lecture notes

CS 5216 Design and Analysis of algorithms

Course Description

Week	TOPIC	READING
1	Order Analysis: Objectives of time analysis of algorithms; Big-oh and Theta notations. Assignments: all exercises of CLR ch1, CLR ch2, CLR ch3	CLR ch1, ch2, ch3
2	Master Theorem and its proof, solution of divide and conquer recurrence relations Assignments: all exercises of CLR ch4	CLR ch4
3	Searching, Sorting and Divide and Conquer Strategy: Linear Search, Binary Search Assignments: all exercises of CLR ch6, ch7	CLR ch6, ch7
4	Searching, Sorting and Divide and Conquer Strategy: Merge-sort; Quick-sort with average case analysis. Heaps and heap-sort. Lower bound on comparison-based sorting and Counting sort. Assignments: all exercises of CLR ch8, ch9	CLR ch 8, ch 9
5	Dynamic Programming: methodology and examples (Fibonacci numbers, Knapsack problem and some other simple examples) Assignments: all exercises of CLR ch 15	CLR ch 15
6	Dynamic Programming: Longest integer subsequence, Longest common subsequence, Weighted interval scheduling Assignments: Presentation on CLR ch 15	CLR ch 15

7	Greedy Method: Methodology, examples (lecture Scheduling, process scheduling) and comparison with DP (more examples to come later in graph algorithms) Assignments: all exercises of CLR ch 16	CLR ch 16
8	Greedy Method: Knapsack problem and some other simple examples Assignments: Presentation on CLR ch 16	CLR ch 16
9	Graph Algorithms: Basics of graphs and their representations. BFS. DFS. Topological sorting. Assignments: all exercises of CLR ch 22, ch 23	CLR ch 22, ch 23
10	Minimum spanning trees (Kruskal and Prim's algorithms and brief discussions of disjoint set and Fibonacci heap data structures). Shortest Paths (Dijkstra, Bellman-Ford, Floyd-Warshall). Assignments: all exercises of CLR ch 24, ch 25	CLR ch 24, ch 25
11	Hard problems and approximation algorithms. Problem classes P, NP, NP-hard and NP-complete, deterministic and nondeterministic polynomial-time algorithms. Assignments: all exercises of CLR ch 34	CLR ch 34
12	Approximation algorithms for some NP-complete problems. Assignments: all exercises of CLR ch 35	CLR ch 34
13	Backtracking, Branch and Bound technique	ES Ch 7, ch8
14	String Matching, Knave algorithm, KMP algorithm	SA Ch 11
15	Parallel Algorithms	SA Ch14

CLR: Cormen, Leiserson, and Rivest. *Algorithms*, MIT Press 2001

ES: Fundamentals of Computer Algorithms by Ellis Horowitz and Sartaj Sahni (Galgotia Publication 1998)

SA: Computer Algorithms Introduction to Design & Analysis by Sara Baase and Allen Van Gelder (Pearson Education 1999)

Subject Learning Outcomes

Development of Learning Outcomes in Domains of Learning: Algorithm design and analysis

The course objectives are to:

Present the concept of algorithm analysis

Study of Notations for Example O notation

Solution of Recurrence Relations

Comprehensive survey of sorting and Searching

Illustrate the idea of Design methods, Divide and conquer, Dynamic programming, Greedy method

Treatment of graph algorithms

Understanding of NP complete problems and approximation algorithms

Based upon above objectives the course goals / learning outcomes are defined below:

- 1 Define key concepts and key notations: Concept of algorithm analysis, O notation, Θ notation, Solution of divide and conquer recurrence relation using Master theorem, Solution of recurrence relation using generating function technique
- 2 Decent coverage of Searching : Sequential search, QuickSequential search, Binary Search, Interpolation Search, Worst case and average case analysis of above mentioned algorithms
- 3 Comprehensive Coverage of Sorting: Divide and conquer strategy, Merge Sort, Quick Sort Heap sort, Insertion Sort, Selection Sort, Bubble Sort, Shell Sort, Analysis of above mentioned algorithms
- 4 Understand Design Strategies: Dynamic Programming Methodology(Fibonacci numbers, Knapsack problem, Longest integer subsequence, Longest common subsequence ,Weighted interval scheduling
- 5 Fundamental ideas of Greedy method: Greedy method, Comparison with dynamic programming, Knapsack problem, Coin Problem, (more examples to come later in graph algorithms)
- 6Basics of Graph Algorithms: Breadth first search, Depth first search, Topological sorting, Minimum weight spanning trees, Prim algorithm, Kruskal algorithm, Dijkstra algorithm, Bellman Ford algorithm
- 7 Idea of different problem classes: Class P, Class NP, NP hard problems, NP complete problems, Deterministic and non deterministic polynomial time algorithms, Approximation algorithms for some NP-complete problems for example approximation algorithms for vertex cover problem and Set cover problem
- 8 Important Concepts: Backtracking, Branch and bound technique, solution of n queen problem and night tour problem using above mentioned concepts, String Matching
- 9 Advanced Ideas: Parallel Algorithms

Learning Resources

1. Required Text(s). Cormen, Leiserson, and Rivest. <i>Algorithms</i> , MIT Press 2001
2. Essential References ALGORITHMS IN C++ by Robert Sedgewick (Pearson Education)2008 Fundamentals of Computer Algorithms by Ellis Horowitz and Sartaj Sahni (Galgotia Publication 1998 Computer Algorithms Introduction to Design & Analysis by Sara Baase and Allen Van Gelder(Pearson Education 1999)
3- Recommended Books and Reference Material (Journals, Reports, etc) (Attach List) IEEE & ACM Journals
4-. Electronic Materials, Web Sites etc http://www.acm.org/ , http://www.ieeexplore.ieee.org/Xplore/dynhome.jsp
5- Other learning material such as computer-based programs/CD, professional standards/regulations

CS 5512 Compiler Design

Course Description

DATE	TOPIC	UNIT
Week 1	Translators, interpreters, assemblers, Compilers, Model of a compiler.	Unit 1
Week 2	Analysis of source program, The phases of a compiler, Cousins of the compilers.	Unit 1
Week 3	Finite automata, non-deterministic and deterministic finite automata, Acceptance of strings by NFA and DFA, Transforming NFA to DFA.	Unit 2
Week 4	Minimization/Optimization of a DFA, related algorithm, Regular sets and regular expression. Obtaining regular expression from finite automata, Lexical analyser design, The role of Lexical Analyser, Input Buffering, Specification of tokens, and Recognition of tokens.	Unit 2
Week 5	Syntax analysis, CFG, derivation of a parse tree, reduction of grammar, useless grammar symbols, Elimination of null and unit productions	Unit 3
Week 6	Elimination of left recursion Regular grammar, Right linear and left linear grammar . Parsing, Top-Down and Bottom Up parsing, general parsing strategies. Brute-force approach, recursive descent parser and algorithms.	Unit 3
Week 7	Simple LL(1) grammar, LL(1) with null and without null rules grammars, Bottom-up parsing- Handle of a right sentential form, Shift-reduce parsers, operator precedence parsing , LR, SLR, canonical LR and LALR grammar and parsers	Unit 3
Week 8	Symbol table contents, organization for non- block structured language-unordered, ordered, and tree-structured and hash Symbol tables.	Unit 4
Week 9	Student presentations	
Week 10	Organization for block structured languages-stack symbols table Stack-implemented tree structured stack implemented hash structured symbol tables.	Unit 4
Week 11	Student presentations	

Week 12	Specification of translations, implementation of translation specified by syntax-directed definition, L-attributed definitions, and syntax-directed translation schemes	Unit 5
Week13	Intermediates code generation, representing three-address statement, translation schemes for programming language constructs.	Unit 5
Week 14	Code Optimization:- Definition, loop Optimization, Elimination of local and global common sub Expressions, loop unrolling, Loop Jamming.	Unit 6
Week 14	Student presentations	
Week 15	Code Generation:- Definition, machine model, code generation methods, Peephole optimization. Error Handling :- Error y, recovery from various phase and parsing.	Unit 6
Week 15	Group presentations	
Week 16	Final exam	

2 Course components (total contact hours):

Lecture:42	Tutorial: 14	Practical/Fieldwork/Internship :	Other:
------------	--------------	-------------------------------------	--------

Learning Resources

Required Text(s) a. Compiler Design, Aho, Ullman, Sethi
2. Essential References 1. Compiler Construction Theory & Practice, Barrett, Bates, Gustafson, Couch 2. The Theory & Practice of Compiler Writing, Jean Paul Tremblay, Paul G. Sorenson
3- Recommended Books and Reference Material (Journals, Reports, etc) (Attach List)
4-.Electronic Materials, Web Sites etc

Computer Graphics

CS-4508

Aims and Objectives

<p>1. Aims of the Course</p> <p>Understand the need of developing graphic applications. Learn the hardware involves in building graphic applications. Learn algorithmic development of graphic primitives like: line, circle, curves, polygon etc. Learn the representation and transformation of graphical images and pictures. Learn the concept of Color Generation.</p>
<p>2. Briefly describe any course development objectives that are being implemented.</p>

Course Description

DATE	TOPIC	READING
Week 1-2	Introduction to Computer Graphics, Application of Graphics, Display Devices: Refresh Cathode -Ray Tubes, Raster Scan Displays, Random Scan Displays, Color CRT Monitors, Flat Panel Displays. Video cards/ display cards Input Devices: Mouse, Trackball, Space ball, Data Glove, Joystick, Light pen, Scanner, Digital Camera, Touch Panels, Voice Systems. Hardcopy Devices: Printers and Plotters	Chapter 2
Week 3-4	Graphics Primitives: Line Generation Algorithms: DDA algorithm, Bresenham's algorithm.	Chapter 3
Week 5-6	Graphics Primitives: Circle Generation Algorithms: Midpoint Circle algorithm, Bresenham's circle generation algorithm. Ellipse Generation algorithm.	Chapter 3
Week 7	Graphics Primitives: Polygon filling Algorithms: Scan Line Polygon fill algorithm, Inside - Outside Tests, Boundary-Fill algorithm, Flood - Fill algorithm. Fundamentals of aliasing and Antialiasing Techniques <i>Text clipping</i> <i>Exterior clipping</i>	Chapter 3-4
Week 8-9	Clipping: Clipping operations, Point clipping, Line clipping: Cohen Sutherland Algorithm, Liang Barsky Algorithm, Nicholl-Lee-Nicholl Algorithm. Polygon clipping: Sutherland- Hodgeman Algorithm, Weiler Atherton Algorithm. Text clipping, Exterior clipping.	Chapter 6
Week 9	Two Dimensional Transformations: Translation, Scaling, Rotation, Reflection, Shear, Homogenous coordinate system, composite transformations, raster method of transformation Two Dimensional Viewing: Window to Viewport coordinate transformation,	Chapter 5
Week 10-11	Three Dimensional: 3D Geometry, 3D display techniques, transformations. Projections: Parallel Projection, Perspective Projection.	Chapter 9, 11
Week 13-14	Color Models and Color Application: Color models: Properties of Light. Standard Primaries and the Chromaticity Diagram, XYZ Color Model, CIE Chromaticity Diagram. RGB Color Model, YIQ Color Model, CMY Color Model, HSV Color Model. Conversion between HSV and RGB Models. HLS Color Model, Color Selection and Application. Advancements in the technology in Computer Graphics.	Chapter 15

Learning Resources

1. Required Text(s)

Computer Graphics: Donald Hearn and M. Pauline Baker, Second Edition, Prentice Hall of India.

2. Essential References

3- Recommended Books and Reference Material (Journals, Reports, etc) (Attach List)

4- Electronic Materials, Web Sites etc

5- Other learning material such as computer-based programs/CD, professional standards/regulations