**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY
HYDERABAD.**

B. TECH. COMPUTER SCIENCE AND ENGINEERING

III Year   **II Semester**
COURSE STRUCTURE

Code	Subject	T	P	C
	Operating Systems		4+1*	-	4
	Compiler Design		4+1*	-	4
	Computer Networks		4+1 *	-	4
	Information Security		4+1 *	-	4
	Artificial Intelligence and Neural Networks		4+1 *	-	4
	Object Oriented Analysis and Design		4+1 *	-	4
	Computer Networks and Case Tools Lab		-	3	2
	Operating systems and Compiler Design Lab		-	3	2
	Total		**30**	**6**	**28**

III Year B.Tech. CSE -II Sem

T	P	C
4+1*	0	4

OPERATING SYSTEMS

UNIT I :
Computer System and Operating System Overview: Overview of computer operating systems operating systems functions protection and security distributed systems special purpose systems operating systems structures and systems calls operating systems generation

UNIT II :
Process Management – Process concepts threads, scheduling-criteria algorithms, their evaluation, Thread scheduling, case studies UNIX, Linux, Windows

UNIT III :
Concurrency : Process synchronization, the critical- section problem, Peterson's Solution, synchronization Hardware, semaphores, classic problems of synchronization, monitors, Synchronization examples, atomic transactions. Case studies UNIX, Linux, Windows

UNIT IV :
Memory Management : Swapping, contiguous memory allocation, paging, structure of the page table , segmentation, virtual memory, demand paging, page-Replacement, algorithms, case studies   UNIX, Linux, Windows

UNIT V :
Principles of deadlock – system model, deadlock characterization, deadlock prevention, detection and avoidance, recovery form deadlock,
I/O systems, Hardware, application interface, kernel I/O subsystem, Transforming I/O requests Hardware operation, STREAMS, performance.

UNIT VI :
File system Interface- the concept of a file, Access Methods, Directory structure, File system mounting, file sharing, protection.
File System implementation- File system structure, file system implementation, directory implementation, directory implementation, allocation methods, free-space management, efficiency and performance, case studies. UNIX, Linux, Windows

UNIT VII :
Mass-storage structure overview of Mass-storage structure, Disk structure, disk attachment disk scheduling, swap-space management, RAID structure, stable-storage implementation, Tertiary storage structure.

UNIT VIII :
Protection : Protection, Goals of Protection, Principles of Protection, Domain of protection Access Matrix, Implementation of Access Matrix, Access control, Revocation of Access Rights, Capability-Based systems, Language – Based Protection,
Security- The Security problem, program threats, system and network threats cryptography as a security tool, user authentication, implementing security defenses, firewalling to protect systems and networks, computer –security classifications, case studies UNIX, Linux, Windows

TEXT BOOKS :
1. Operating System Concepts- Abraham Silberchatz, Peter B. Galvin, Greg Gagne 7th Edition, John Wiley.
2. Operating systems- A Concept based Approach-D.M.Dhamdhere, 2nd Edition, TMH

REFERENCES :
1. Operating Systems' – Internal and Design Principles Stallings, Fifth
   Edition–2005, Pearson education/PHI
2. Operating System A Design Approach-Crowley, TMH.
3. Modern Operating Systems, Andrew S Tanenbaum 2nd edition
   Pearson/PHI.

III Year B.Tech. CSE -II Sem

T	P	C
4+1*	0	4

COMPILER DESIGN

UNIT – I
Overview of Compilation: Phases of Compilation – Lexical Analysis, Regular Grammar and regular expression for common programming language features, pass and Phases of translation, interpretation, bootstrapping, data structures in compilation – LEX lexical analyzer generator.

UNIT – II
Top down Parsing : Context free grammars, Top down parsing – Backtracking, LL (1), recursive descent parsing, Predictive parsing, Preprocessing steps required for predictive parsing.

UNIT – III
Bottom up parsing : Shift Reduce parsing, LR and LALR parsing, Error recovery in parsing , handling ambiguous grammar, YACC – automatic parser generator.

UNIT – IV
Semantic analysis : Intermediate forms of source Programs – abstract syntax tree, polish notation and three address codes. Attributed grammars, Syntax directed translation, Conversion of popular Programming languages language Constructs into Intermediate code forms, Type checker.

UNIT – V
Symbol Tables : Symbol table format, organization for block structures languages, hashing, tree structures representation of scope information. Block structures and non block structure storage allocation: static, Runtime stack and heap storage allocation, storage allocation for arrays, strings and records.

UNIT – VI
Code optimization : Consideration for Optimization, Scope of Optimization, local optimization, loop optimization, frequency reduction, folding, DAG representation.

UNIT – VII
Data flow analysis : Flow graph, data flow equation, global optimization, redundant sub expression elimination, Induction variable elements, Live variable analysis, Copy propagation.

UNIT – VIII
Object code generation : Object code forms, machine dependent code optimization, register allocation and assignment generic code generation algorithms, DAG for register allocation.

TEXT BOOKS :
1. Principles of compiler design -A.V. Aho . J.D.Ullman; Pearson Education.
2. Modern Compiler Implementation in C- Andrew N. Appel, Cambridge
   University Press.

REFERENCES :
1. lex &yacc – John R. Levine, Tony Mason, Doug Brown, O'reilly
2. Modern Compiler Design- Dick Grune, Henry E. Bal, Cariel T. H. Jacobs,
   Wiley dreamtech.
3. Engineering a Compiler-Cooper & Linda, Elsevier.
4. Compiler Construction, Louden, Thomson.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY
HYDERABAD

III Year B.Tech. CSE -II Sem

T	P	C
4+1*	0	4

COMPUTER NETWORKS

UNIT – I
Introduction : OSI, TCP/IP and other networks models, Examples of Networks: Novell Networks ,Arpanet, Internet, Network Topologies WAN, LAN, MAN.

UNIT - II
Physical Layer : Transmission media copper, twisted pair wireless, switching and encoding asynchronous communications; Narrow band, broad band ISDN and ATM.

UNIT - III
Data link layer : Design issues, framing, error detection and correction, CRC, Elementary Protocol-stop and wait, Sliding Window, Slip, Data link layer in HDLC, Internet, ATM.

UNIT - IV
Medium Access sub layer : ALOHA, MAC addresses, Carrier sense multiple access. IEEE 802.X Standard Ethernet, wireless LANS. Bridges

UNIT - V
Network Layer : Virtual circuit and Datagram subnets-Routing algorithm shortest path routing, Flooding, Hierarchical routing, Broad cast, Multi cast, distance vector routing.

UNIT – VI
Dynamic routing – Broadcast routing. Rotary for mobility. Congestion, Control Algorithms – General Principles – of Congestion prevension policies. Internet working: The Network layer in the internet and in the ATM Networks.

UNIT –VII
Transport Layer: Transport Services, Connection management, TCP and UDP protocols; ATM AAL Layer Protocol.

UNIT – VIII
Application Layer – Network Security, Domain name system, SNMP, Electronic Mail; the World WEB, Multi Media.

TEXT BOOKS :
1. Computer Networks — Andrew S Tanenbaum, 4th Edition. Pearson Education/PHI
2. Data Communications and Networking – Behrouz A. Forouzan.Third Edition TMH.

REFERENCES :
1. An Engineering Approach to Computer Networks-S.Keshav, 2nd Edition, Pearson Education
2. Understanding communications and Networks, 3rd Edition, W.A. Shay, Thomson

III Year B.Tech. CSE -II Sem

T	P	C
4+1*	0	4

INFORMATION SECURITY

UNIT - I
Security Attacks (Interruption, Interception, Modification and Fabrication), Security Services (Confidentiality, Authentication, Integrity, Non-repudiation, access Control and Availability) and Mechanisms, A model for Internetwork security, Internet Standards and RFCs, Buffer overflow & format string vulnerabilities, TCP session hijacking, ARP attacks, route table modification, UDP hijacking, and man-in-the-middle attacks.

UNIT - II
Conventional Encryption Principles, Conventional encryption algorithms, cipher block modes of operation, location of encryption devices, key distribution Approaches of Message Authentication, Secure Hash Functions and HMAC.

UNIT - III
Public key cryptography principles, public key cryptography algorithms, digital signatures, digital Certificates, Certificate Authority and key management Kerberos, X.509 Directory Authentication Service.

UNIT - IV
Email privacy: Pretty Good Privacy (PGP) and S/MIME.

UNIT - V
IP Security Overview, IP Security Architecture, Authentication Header, Encapsulating Security Payload, Combining Security Associations and Key Management.

UNIT - VI
Web Security Requirements, Secure Socket Layer (SSL) and Transport Layer Security (TLS), Secure Electronic Transaction (SET).

UNIT - VII
Basic concepts of SNMP, SNMPv1 Community facility and SNMPv3.
Intruders, Viruses and related threats.

UNIT - VIII
Firewall Design principles, Trusted Systems. Intrusion Detection Systems.

TEXT BOOKS :
1. Network Security Essentials (Applications and Standards) by William Stallings Pearson Education.
2. Hack Proofing your network by Ryan Russell, Dan Kaminsky, Rain Forest Puppy, Joe Grand, David Ahmad, Hal Flynn Ido Dubrawsky, Steve W.Manzuik and Ryan Permeh, wiley Dreamtech

REFERENCES :
1. Fundamentals of Network Security by Eric Maiwald (Dreamtech press)
2. Network Security - Private Communication in a Public World by Charlie Kaufman, Radia Perlman and Mike Speciner, Pearson/PHI.
3. Cryptography and network Security, Third edition, Stallings, PHI/Pearson
4. Principles of Information Security, Whitman, Thomson.
5. Network Security: The complete reference, Robert Bragg, Mark Rhodes, TMH
6. Introduction to Cryptography, Buchmann, Springer.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY
HYDERABAD

III Year B.Tech. CSE -II Sem　　　　　　　　　　　　　　　　**T　　P　　C**
　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　4+1*　0　　4

ARTIFICIAL INTELLIGENCE AND NEURAL NETWORKS

UNIT - I
Introduction : AI problems, foundation of AI and history of AI intelligent agents: Agents and Environments,the concept of rationality, the nature of environments, structure of agents, problem solving agents, problemformulation.

UNIT - II
Searching : Searching for solutions, uniformed search strategies – Breadth first search, depth first Search. Search with partial information (Heuristic search) Greedy best first search, A* search　Game Playing: Adversial search, Games, minimax, algorithm, optimal decisions in multiplayer games, Alpha-Beta pruning, Evaluation functions, cutting of search.

UNIT - III
Knowledge Representation & Reasons logical Agents, Knowledge – Based Agents, the Wumpus world, logic, propositional logic, Resolution patterns in propos ional logic, Resolution, Forward & Backward. Chaining.

UNIT - IV
First order logic. Inference in first order logic, propositional Vs. first order inference, unification & lifts forward chaining, Backward chaining, Resolution.

UNIT - V
Characteristics of Neural Networks, Historical Development of Neural Networks Principles, Artificial Neural Networks: Terminology, Models of Neuron, Topology, Basic Learning Laws, Pattern Recognition Problem, Basic Functional Units, Pattern Recognition Tasks by the Functional Units.

UNIT - VI
Feedforward Neural Networks:
Introduction, Analysis of pattern Association Networks, Analysis of Pattern Classification Networks, Analysis of pattern storage Networks. Analysis of Pattern Mapping Networks.

UNIT - VII
Feedback Neural Networks
Introduction, Analysis of Linear Autoassociative FF Networks, Analysis of Pattern Storage　Networks.

UNIT - VIII
Competitive Learning Neural Networks & Complex pattern Recognition
Introduction, Analysis of Pattern Clustering Networks, Analysis of Feature
Mapping Networks, Associative Memory.

TEXT BOOKS :
1. Artificial Intelligence – A Modern Approach. Second Edition, Stuart Russel, Peter Norvig, PHI/ Pearson Education.
2. Artificial Neural Networks B. Yagna Narayana, PHI
REFERENCES :
1. Artificial Intelligence , 2nd Edition, E.Rich and K.Knight (TMH).
2. Artificial Intelligence and Expert Systems – Patterson PHI.
3. Expert Systems: Principles and Programming- Fourth Edn, Giarrantana/ Riley, Thomson.
4. PROLOG Programming for Artificial Intelligence. Ivan Bratka- Third Edition – Pearson Education.
5.Neural Networks Simon Haykin PHI
6. Artificial Intelligence, 3rd Edition, Patrick Henry Winston., Pearson Edition.

III Year B.Tech. CSE -II Sem	T	P	C
	4+1*	0	4

OBJECT ORIENTED ANALYSIS AND DESIGN

UNIT - I
Introduction to UML : Importance of modeling, principles of modeling, object oriented modeling, conceptual model of the UML, Architecture, Software Development Life Cycle.

UNIT - II
Basic Structural Modeling : Classes, Relationships, common Mechanisms, and diagrams.
Advanced Structural Modeling : Advanced classes, advanced relationships, Interfaces, Types and Roles, Packages.

UNIT - III
Class & Object Diagrams : Terms, concepts, modeling techniques for Class & Object Diagrams.

UNIT- IV
Basic Behavioral Modeling-I : Interactions, Interaction diagrams.

UNIT - V
Basic Behavioral Modeling-II : Use cases, Use case Diagrams, Activity
Diagrams.

UNIT - VI
Advanced Behavioral Modeling : Events and signals, state machines, processes and Threads, time and space, state chart diagrams.

UNIT-VII
Architectural Modeling : Component, Deployment, Component diagrams and Deployment diagrams.

UNIT - VIII
Case Study : The Unified Library application.

TEXT BOOKS :
1. Grady Booch, James Rumbaugh, Ivar Jacobson : The Unified Modeling
   Language User Guide, Pearson Education.
2. Hans-Erik Eriksson, Magnus Penker, Brian Lyons, David Fado: UML 2
   Toolkit, WILEY-Dreamtech India Pvt. Ltd.

REFERENCE BOOKS:
1. Meilir Page-Jones: Fundamentals of Object Oriented Design in UML,
   Pearson Education.
2. Pascal Roques: Modeling Software Systems Using UML2, WILEY-
   Dreamtech India Pvt. Ltd.
3. Atul Kahate: Object Oriented Analysis & Design, The McGraw-Hill
   Companies.
4. Mark Priestley: Practical Object-Oriented Design with UML,TATA
   McGrawHill
5. Appling UML and Patterns: An introduction to Object – Oriented Analysis
   and Design and Unified Process, Craig Larman, Pearson Education.

III Year B.Tech. CSE -II Sem

T	P	C
0	3	2

COMPUTER NETRWORKS AND CASE TOOLS LAB

Objective:
- To Understand the functionalities of various layers of OSI model
- To inculcate object oriented software design

System/ Software Requirement
- Intel based desktop PCs LAN CONNECTED with minimum of 166 MHZ or faster processor with atleast 64 MB RAM and 100 MB free disk space
- Tools Such as Rational Rose

Part - A

1. Implement the data link layer framing methods such as character, character stuffing and bit stuffing.
2. Implement on a data set of characters the three CRC polynomials – CRC 12, CRC 16 and CRC CCIP
.
3. Implement Dijkstra 's algorithm to compute the Shortest path thru a graph.
4. Take an example subnet graph with weights indicating delay between nodes. Now obtain Routing table art each node using distance vector routing algorithm
5. Take an example subnet of hosts . Obtain broadcast tree for it.
6. Take a 64 bit playing text and encrypt the same using DES algorithm .
7. Write a program to break the above DES coding
8. Using RSA algorithm Encrypt a text data and Decrypt the same .

Part - B

1. The student should take up the case study of Unified Library application which is mentioned in the theory, and Model it in different views i.e Use case view, logical view, component view, Deployment view, Database design, forward and Reverse Engineering, and Generation of documentation of
the project.
2. Student has to take up another case study of his/her own interest and do the same what ever mentioned in first problem. Some of the ideas regarding case studies are given in reference books which were mentioned in theory syllabus can be referred for some idea.
Note : The analysis, design, coding, documentation, database design of mini project which will be carried out in 4th year should be done in object-oriented approach using UML and by using appropriate software which supports UML, otherwise the mini project will not be evaluated.

III Year B.Tech. CSE -II Sem

T	P	C
0	3	2

OPERATING SYSTEMS AND COMPILER DESIGN LAB

Objective :
- To provide an understanding of the language translation peculiarities by designing a complete translator for a mini language.
- To provide an understanding of the design aspects of operating system

Recommended Systems/Software Requirements:
- Intel based desktop PC with minimum of 166 MHZ or faster processor with atleast 64 MB RAM and 100 MB free disk space
- C++ complier and JDK kit

Part - A
1. Simulate the following CPU scheduling algorithms
a) Round Robin b) SJF c) FCFS d) Priority
2. Simulate all file allocation strategies
a) Sequentialb) Indexed c) Linked
3. Simulate MVT and MFT
4. Simulate all File Organization Techniques
a) Single level directory b) Two level c) Hierarchical d) DAG
5. Simulate Bankers Algorithm for Dead Lock Avoidance
6. Simulate Bankers Algorithm for Dead Lock Prevention
7. Simulate all page replacement algorithms
a) FIFO b) LRU c) LFU Etc. …
8. Simulate Paging Technique of memory management.

PART - B
Consider the following mini Language, a simple procedural high-level language, only operating on integer
data, with a syntax looking vaguely like a simple C crossed with Pascal. The syntax of the language is
defined by the following BNF grammar:
<program> ::= <block>
<block> ::= { <variabledefinition> <slist> }
| { <slist> }
<variabledefinition> ::= int <vardeflist> ;
<vardeflist> ::= <vardec> | <vardec> , <vardeflist>
<vardec> ::= <identifier> | <identifier> [<constant>]
<slist> ::= <statement> | <statement> ; <slist>
<statement> ::= <assignment> | <ifstatement> | <whilestatement>
| <block> | <printstatement> | <empty>
<assignment> ::= <identifier> = <expression>
| <identifier> [<expression>] = <expression>
<ifstatement> ::= if <bexpression> then <slist> else <slist> endif
| if <bexpression> then <slist> endif
<whilestatement> ::= while <bexpression> do <slist> enddo
<printstatement> ::= print (<expression>)
<expression> ::= <expression> <addingop> <term> | <term> | <addingop> <term>
<bexpression> ::= <expression> <relop> <expression>
<relop> ::= < | <= | == | >= | > | !=
<addingop> ::= + | -
<term> ::= <term> <multop> <factor> | <factor>
<multop> ::= * | /
<factor> ::= <constant> | <identifier> | <identifier> [<expression>]
| (<expression>)
<constant> ::= <digit> | <digit> <constant>
<identifier> ::= <identifier> <letterordigit> | <letter>
<letterordigit> ::= <letter> | <digit>
<letter> ::= a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z
<digit> ::= 0|1|2|3|4|5|6|7|8|9
<empty> has the obvious meaning
Comments (zero or more characters enclosed between the standard C/Java-style comment brackets /
.../) can be inserted. The language has rudimentary support for 1-dimensional arrays. The declaration
int a[3] declares an array of three elements, referenced as a[0], a[1] and a[2]. Note also that you should

worry about the scoping of names.
A simple program written in this language is:
```
{ int a[3],t1,t2;
t1=2;
a[0]=1; a[1]=2; a[t1]=3;
t2=-(a[2]+t1*6)/(a[2]-t1);

if t2>5 then
print(t2);
else {
int t3;
t3=99;
t2=-25;
print(-t1+t2*t3); /* this is a comment
on 2 lines */
} endif }
```

1. Design a Lexical analyzer for the above language. The lexical analyzer should ignore redundant spaces, tabs and newlines. It should also ignore comments. Although the syntax specification states that identifiers can be arbitrarily long, you may restrict the length to some reasonable value.
2. Implement the lexical analyzer using JLex, flex or lex or other lexical analyzer generating tools.
3. Design Predictive parser for the given language
4. Design LALR bottom up parser for the above language.
5. Convert the BNF rules into Yacc form and write code to generate abstract syntax tree.
6. Write program to generate machine code from the abstract syntax tree generated by the parser. The following instruction set may be considered as target code.
The following is a simple register-based machine, supporting a total of 17 instructions. It has three distinct internal storage areas. The first is the set of 8 registers, used by the individual instructions as detailed below, the second is an area used for the storage of variables and the third is an area used for the storage of program. The instructions can be preceded by a label. This consists of an integer in the range 1 to 9999 and the label is followed by a colon to separate it from the rest of the instruction. The numerical label can be used as the argument to a jump instruction, as detailed below.
In the description of the individual instructions below, instruction argument types are specified as follows :

R

specifies a register in the form R0, R1, R2, R3, R4, R5, R6 or R7 (or r0, r1, etc.).

L

specifies a numerical label (in the range 1 to 9999).

V

specifies a "variable location" (a variable number, or a variable location pointed to by a register - see below).

A

specifies a constant value, a variable location, a register or a variable location pointed to by a register (an indirect address). Constant values are specified as an integer value, optionally preceded by a minus sign, preceded by a # symbol. An indirect address is specified by an @ followed by a register.

So, for example, an A-type argument could have the form 4 (variable number 4), #4 (the constant value 4), r4 (register 4) or @r4 (the contents of register 4 identifies the variable location to be accessed).

The instruction set is defined as follows:

LOAD A,R

loads the integer value specified by A into register R.

STORE R,V

stores the value in register R to variable V.

OUT R

outputs the value in register R.

NEG R

negates the value in register R.

ADD A,R

adds the value specified by A to register R, leaving the result in register R.

SUB A,R

subtracts the value specified by A from register R, leaving the result in register R.

MUL A,R

multiplies the value specified by A by register R, leaving the result in register R.

DIV A,R

divides register R by the value specified by A, leaving the result in register R.

JMP L

causes an unconditional jump to the instruction with the label L.

JEQ R,L

jumps to the instruction with the label L if the value in register R is zero.

JNE R,L

jumps to the instruction with the label L if the value in register R is not zero.

JGE R,L

jumps to the instruction with the label L if the value in register R is greater than or equal to zero.

JGT R,L

jumps to the instruction with the label L if the value in register R is greater than zero.

JLE R,L

jumps to the instruction with the label L if the value in register R is less than or equal to zero.

JLT R,L

jumps to the instruction with the label L if the value in register R is less than zero.

NOP

is an instruction with no effect. It can be tagged by a label.

STOP

stops execution of the machine. All programs should terminate by executing a STOP instruction.