UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

Lecture-wise Plan

Subject Name: Organizational Behavior **Subject Code-HU801**
Year: 4th Year **Semester: Eighth**

Module Number	Topics	Number of Lectures
1	**Introduction:**	**23L**
	1. Organizational Behaviour: Definition, Importance, Historical Background, Fundamental Concepts of OB, Challenges and Opportunities for OB.	2
	2. Personality and Attitudes: Meaning of personality, Personality Determinants and Traits, Development of Personality, Types of Attitudes, Job Satisfaction.	3
	3. Perception: Definition, Nature and Importance, Factors influencing Perception, Perceptual Selectivity, Link between Perception and Decision Making.	3
	4. Motivation: Definition, Theories of Motivation - Maslow's Hierarchy of Needs Theory, McGregor's Theory X & Y, Herzberg's Motivation-Hygiene Theory, Alderfer's ERG Theory, McClelland's Theory of Needs, Vroom's Expectancy Theory.	5
2	5. Group Behaviour: Characteristics of Group, Types of Groups, Stages of Group Development, Group Decision Making.	3
	6. Communication: Communication Process, Direction of Communication, Barriers to Effective Communication.	3
	7. Leadership: Definition, Importance, Theories of Leadership Styles.	4
	Organizational Politics	
3.		**10L**
	8. Organizational Politics: Definition, Factors contributing to Political Behaviour.	2
	9. Conflict Management: Traditional vis-a-vis Modern View of Conflict, Functional and Dysfunctional Conflict, Conflict Process, Negotiation – Bargaining Strategies, Negotiation Process.	4
4	10. Organizational Design: Various Organizational Structures and their Effects on Human Behaviour, Concepts of Organizational Climate and Organizational Culture.	4
	TOTAL	**33L**

Faculty In-Charge HOD, Humamities Dept.

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

Lecture-wise Plan

Subject Name: Advance Computer Architecture **Subject Code: CS801A**
Year: 4th Year **Semester: Eighth**

Module Number	Topics	Number of Lectures
1	1. Computer Architecture and Organization-review, Fundamentals of Computer Design, Technology Trends Cost Performance Analysis 2. Parallel Processing Architectures-Taxonomy-SISD, MISD, SIMD, MIMD,PRAM models 3. Data and Resource, Dependencies, Program Partitioning and Scheduling, Control Flow vs. Data Flow	3 3 3
2	1. Network topologies - Static, Dynamic, Types of Networks 2. RISC vs. CISC, Memory Hierarchy, Virtual Memory 3. Concepts of Pipelining, Instruction Pipelining, dynamic pipelining, arithmetic pipelines	3 4 4
3	1. Multiprocessors- Multi stage Networks, Cache Coherence, Synchronization, Message – passing 2. Vector Processing Principles – Instruction types, Compound, Vector Loops, Chaining 3. Array Processors- Structure, Algorithms	4 4 3
4	1. Data Flow Architecture -Graphs. Petri Nets, Static and Dynamic DFA,VLSI Computations 2. Parallel Programming Models, Languages, Compilers	4 4
Total Lecture Hours – 39 l.h.		

Faculty In-Charge HOD, CSE Dept.

Assignments: -

Unit 1:-

1. A digital computer has a common bus system for 16 registers of 32 bits each. The bus is constructed with multiplexers.
   (i) How many selection inputs are there in each multiplier?
   (ii) What sizes of multiplexers are needed?
   (iii) How many multiplexers are there in the bus?

2. Explain the importance of different addressing modes in computer architecture with suitable example.
3. What is an instruction format? Explain different types of instruction formats in detail.
4. Explain Flynn's classification of computers.
5. Explain in detail RISC pipeline. Why is the cache miss penalty, greater in deeply pipelined processor?
6. Why should the sign of the remainder after a division be the same as the sign of the dividend?
7. What is the difference between a direct and indirect address instruction? How many references to memory are needed for each type of instruction to bring an operand into a processor register?

Unit 2: -

1. What do you mean by instruction cycle and interrupt cycle? Draw the flowchart for instruction Cycle.
2. A no pipeline system takes 50 ns to process a task. The same task can be processed in 6 segment pipeline with a clock cycle of 10 ns. Determine the speedup ratio of pipeline for 100 tasks. What is maximum speedup ratio?
3. Explain hazards to the instruction pipeline with their solution.
4. Draw a diagram showing how the instruction fetch and execution units of a superscalar processor are connected. Show the widths of the data path (in words - not bits; your diagram should be relevant to a 32-bit or 64-bit processor). Which factor primarily determines performance: the instruction issue width (number of instructions issued per cycle) or the number of functional units?
5. Discuss the dynamic pipeline with proper example.

Unit 3:-

1. How many instructions can a 4-way superscalar complete in one cycle?
2. Distinguish between a 'Uniform Memory Access' system and a 'Non-Uniform Memory Access' system.

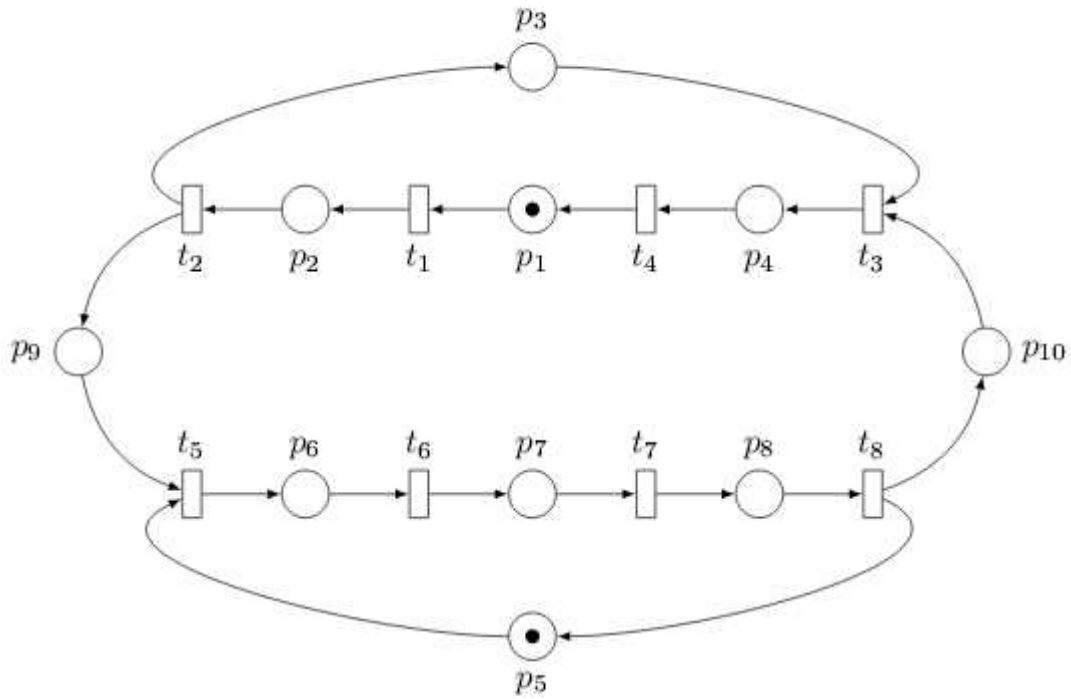
3. When using a shared memory, cache coherence transactions are expensive and could potentially clog up a bus so much that the bandwidth available to useful transactions becomes very low. A hacker (who's spent his time reading the manual for his processor on the net instead of going to SE363 lectures) discovers that there's an instruction that will disable the cache and decides that this will solve the problem. Why is this likely to be a bad idea?
4. Where can you find a small dataflow machine in every high performance processor?
5. What if vector data is not stored in a strided fashion in memory? (irregular memory access to a vector)
6. Explain briefly (what it is, what it does, and what it is used for?):
   a. Network topology
   b. Routing

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

Lecture-wise Plan

Subject Name: Advance Computer Architecture **Subject Code: CS801A**
Year: 4th Year **Semester: Eighth**
Unit 4:-

1. Given is the following petri net N_1:



   1. What are the pre and post sets of transitions t_5 and t_8 and of place p_3?
   2. Which transitions are enabled after t_1 and t_2 fired?

2. Why is it not possible with a reachability algorithm to determine in general, whether a given state in a petri net is reachable or not?
3. Differentiate between static and dynamic DFA.
4. Why should the algorithm be distributed?

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

Lecture-wise Plan

Subject Name: Parallel Computing **Subject Code-CS801B**

Year: 4th Year **Semester: Eighth**

Module Number	Topics	Number of Lectures
1	**Introduction:**	**7L**
	Parallel Processing Environment Overview	1
	Pipelining and Data Parallelism	1
	Scalability, Flynn's Taxonomy	1
	Parallel Processing Organization- Mesh	1
	Hyper-Tree	1
	Pyramid and Butterfly	1
	Hypercube network	1
2	**Parallel Algorithm:**	**16L**
	Parallel Algorithms –Structure, cost, Analysis ;	2
	Elementary Algorithms: Broadcast, Prefix sums, All sums	2
	Algorithms on Selection problem,	2
	Merging-Odd-even merging network	2
	CREW Merging, N-ary searching	2
	Matrix Transposition	1
	Matrix Multiplications- 2D Mesh SIMD ,Hypercube SIMD, Shuffle-Exchange SIMD models.	3
	Discrete Fourier Transform, Fast Fourier Transform	2
3.	**Linear system of equations:**	**6L**
	Linear system of equations- Gaussian Elimination,	2
	Gauss-Seidel algorithm, Jacobi algorithm	1
	Sorting – Enumeration sort, Odd-even transposition sort,	2
	Bitonic merge Ellis's Algorithm	1
4	**Graph Algorithms:**	**8L**
	Graph Algorithms	2
	Spanning Tree Algorithms	2
	Parallel Programming Languages –FORTRAN 90,	2
	OCCAM	2
Total Number Of Hours = 37		

Faculty In-Charge HOD, CSE Dept.

Assignment:

Module-1:

1. What are the levels of parallel processing?
2. What is pipelining? Discuss in detail principles of designing pipeline processor.

Module-2:

1. Explain SIMD interconnection networks?
2. What are the types of SIMD array processors?

Module-3:

1. What is parallel algorithm? Explain the design process of Parallel Algorithms.
2. Discuss different attributes of parallel algorithms?

Module-4:

1. Explain row wise 1-D & 2-D partitioning parallel algorithm for Matrix-Vector Multiplication.
2. What is Cannon's algorithm for Matrix multiplication? Discuss performance analysis of cannon's algorithm.

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

Lecture-wise Plan

Subject Name: Natural Language Processing **Subject Code-CS801C**
Year: 4th Year **Semester: Eighth**

Module Number	Topics	Number of Lectures
1	**Regular Expressions and Automata Recap**	**2L**
	Introduction to NLP, Regular Expression, Finite State Automata	2
2	**Tokenization**	**5L**
	Word Tokenization, Normalization, Sentence Segmentation, Named Entity Recognition, Multi Word Extraction, Spell Checking – Bayesian Approach, Minimum Edit Distance	5
3.	**Morphology:**	**4L**
	Morphology – Inflectional and Derivational Morphology, Finite State Morphological Parsing, The Lexicon and Morphotactics, Morphological Parsing with Finite State Transducers, Orthographic Rules and Finite State Transducers, Porter Stemmer.	4
4	**Language Modeling:**	**4L**
	Introduction to N-grams, Chain Rule, Smoothing – Add-One Smoothing, Witten-Bell Discounting; Back off, Deleted Interpolation, N-grams for Spelling and Word Prediction, Evaluation of language models. Hidden Markov Models and POS Tagging [4L] Markov Chain, Hidden Markov Models, Forward Algorithm, Viterbi Algorithm, Part of Speech Tagging – Rule based and Machine Learning based approaches, Evaluation	2
5	**Text Classification:**	**4L**
	Text Classification, Naïve Bayes' Text Classification, Evaluation, Sentiment Analysis – Opinion Mining and Emotion Analysis, Resources and Techniques	**4**
6	**Context Free Grammar:**	**5L**
	Context Free Grammar and Constituency, Some common CFG phenomena for English, Top-Down and Bottom-up parsing, Probabilistic Context Free Grammar, Dependency Parsing	**5**
7	**Computational Lexical Semantics:**	**4L**
	Introduction to Lexical Semantics – Homonymy, Polysemy, Synonymy, Thesaurus – WordNet, Computational Lexical Semantics – Thesaurus based and Distributional Word Similarity	**4**

8	**Information Retrieval:**	**5L**
	Boolean Retrieval, Term-document incidence, The Inverted Index, Query Optimization, Phrase Queries, Ranked Retrieval – Term Frequency – Inverse Document Frequency based ranking, Zone Indexing, Query term proximity, Cosine ranking, Combining different features for ranking, Search Engine Evaluation, Relevance Feedback	**5**
Total Number Of Hours = 33		

Assignment:

Module-1:

1. Write an algorithm for parsing a finite-state transducer using the pseudo- code. Explain the algorithm with an example. Also give the merits and demerits of this algorithm.
2. How the natural language processing systems are evaluated? Explain. Differentiate between natural language processing and natural language understanding.

Module-2:

1. Explain Bayesian Approach in spell checking.
2. What do you mean by Word Tokenization?

Module-3:

1. Write the difference between Inflectional and Derivational Morphology
2. What are the Orthographic Rules?

Module-4:

1. Discuss the following topics:
   A. Language as a rule-based system.
   B. Stochastic Part-of-Speech tagging.
2. Between the words eat and find which would you expect to be more effective in selection restriction-based sense disambiguation? Why?

Module-5:

1. Discuss Naïve Bayes' Text Classification with example.
2. Write down the techniques of sentiment analysis

Module-6:

1. Write an algorithm for converting an arbitrary context- free grammar into Chomsky normal form. Explain it with a suitable example.
2. Define Probabilistic Context Free Grammar.

Module-7:

1. Discuss Thesaurus based and Distributional Word Similarity.
2. Explain Lexical Semantics.

Module-8:

1. Write Short notes on the following topics:
   A. Frequency based ranking,
   B. Cosine ranking

UNIVERSITY OF ENGINEERING & MANAGEMENT,JAIPUR

Lecture-wise Plan

Subject Name: Cryptography & Network Security **Subject Code-CS801D**
Year: 4th Year **Semester: Eighth**

Module Number	Topics	Number of Lectures
1.	**Introduction:**	**4L**
	1. Attacks on Computers & Computer Security	2
	2. Need for Security, Security approaches,	1
	3. Principles of Security, Types of attack	1
2.	**Cryptography: Concepts & Techniques**	**6L**
	1. Introduction, Plaintext & Cipher text, Substitution Techniques	2
	2. Transposition Techniques, Encryption & Decryption,	2
	3. Symmetric & Asymmetric key Cryptography, Key Range & Key Size	2
3.	**Symmetric Key Algorithm**	**6L**
	1. Introduction, Algorithm types & Modes, Overview of Symmetric Key Cryptography	2
	2. DES(Data Encryption Standard) algorithm,	2
	3. IDEA(International Data Encryption Algorithm) algorithm, RC5(Rivest Cipher 5) algorithm.	2
4.	**Asymmetric Key Algorithm, Digital Signature and RSA**	**6L**
	1. Introduction, Overview of Asymmetric key Cryptography	2
	2. RSA algorithm, Symmetric & Asymmetric key Cryptography together,	2
	3. Digital Signature, Basic concepts of Message Digest and Hash Function	2
5.	**Internet Security Protocols, User Authentication**	**6L**
	1. Internet Security Protocols, User Authentication	2
	2. Authentication Basics, Password, Authentication Token	2
	3. Certificate based Authentication, Biometric Authentication	2
6.	**Electronic Mail Security**	**6L**
	1. Basics of mail security	3
	2. Pretty Good Privacy, S/MIME	3
7.	**Firewall**	**6L**
	1. Introduction, Types of firewall	3
	2. Firewall Configurations, DMZ Network	3
Total Number Of Hours = 40		

Faculty In-Charge HOD, CSE Dept.

Assignment :

Module -1 (Introduction)

Module -2 (Cryptography: Concepts & Techniques)

Module -3 (Symmetric Key Algorithm)

Module -4 (Asymmetric Key Algorithm, Digital Signature and RSA)

Module -5 (Internet Security Protocols, User Authentication)

Module -6 (Electronic Mail Security)

Module -7 (Firewall)

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

Lecture-wise Plan

Subject Name: Cyber law & Security Policy **Subject Code-CS802B**

Year: 4th Year Semester: Eighth

Module Number	Topics	Number of Lectures
1	**Introduction:**	**4L**
	1. What is cybercrime? Forgery, Hacking, Software Piracy, Computer Network intrusion.	4L
2	**Category of Cybercrime**	**4L**
	1. How criminals plan attacks, passive attack, Active attacks, cyberstalking.	4L
3	**Cybercrime Mobile & Wireless devices**	**8L**
	1. Security challenges posted by mobile devices	2L
	2. Cryptographic security for mobile devices	2L
	3. Attacks on mobile/cell phones, Theft, Virus, Hacking. Bluetooth	2L
	4. Different viruses on laptop	2L
4	**Tools and Methods used in Cyber crime**	**8L**
	1. Proxy servers, password checking, Random checking	3L
	2. Trojan Horses and Backdoors; DOS & DDOS attacks; SQL injection: buffer over flow	5L
5	**Phishing & Identity Theft:**	**4L**
	1. Phising methods, ID Theft, Online identity method.	4L
6	**Cybercrime & Cybersecurity:**	**4L**
	Legal aspects, Indian laws, IT act, Public key certificate	4L

Faculty In-Charge HOD, CSE Dept.

Assignment:

Module-1 (Introduction):

1. Hacking
2. Software Piracy
3. Computer Network intrusion.

Module-2 (Category of Cybercrime):

1. Passive attack
2. Active attacks

Module-3 (Cybercrime Mobile & Wireless devices):

1. Security challenges posted by mobile devices
2. Cryptographic security for mobile devices
3. Attacks on mobile/cell phones

Module-4 (Tools and Methods used in Cyber crime):

1. Trojan Horses and Backdoors
2. DOS & DDOS attacks
3. SQL injection: buffer over flow

Module-5 (Phishing & Identity Theft):

1. ID Theft

Module-6 (Cybercrime & Cyber security):

1. Indian laws
2. Public key certificate

UNIVERSITY OF ENGINEERING & MANAGEMENT,JAIPUR

Lecture-wise Plan

Subject Name: Optical Networking **Subject Code-CS802C**
Year: 4th Year **Semester: Eighth**

Module Number	Topics	Number of Lectures
1.	**Unit 1**	**10L**
	Basics of Optical communications-: Sources. Transmitters. Modulators. Optical fiber. Photodetectors, and Receivers.	2
	Switching in networks. Circuit switched. Packet switched. Cell switched. Virtual circuit switched. Burst switched (fast circuit switched).	2
	Transmission, Asynchronous, Synchronous.	1
	Layering in packet switched networks, Motivation. Commonly used abstraction, Physical layer. Datalink layer. Network layer. Transport layer. Application layer.	2
	Layering in circuit switched networks., Physical layer. Multiplexing standards. Signalling- CAS, CCS. SS7concept.	3
2.	**Unit 2**	**8L**
	Data plane, management plane, control plane-concept.	1
	First generation networks, SDH/SONET. Computer interconnections- ESCON, Fiber Channel, HIPPI. FDDI. ATM. DQDB	2
	Components– description., Mode locked laser (for ps pulses). Tunable filters. Multiplexers. Demultiplexers. Tunable wavelength convertors. Optical amplifiers. a. Fiber- EDFA. b. SOA, Tunable transmitters. Tunable receivers. Dispersion compensating fibers.	3
	Multiplexing techniques, SDM, TDMA, WDMA(OFDMA), DWDM, SCM, CDMA.	2
3.	**Unit 3**	**9L**
	Protocols for single channel broadcast networks. (recapitulation), ALOHA, CSMA/CD, Problems with CSMA/CD, Definition of high speed network.	1
	Classification of multiple access methods.(recapitulation), Random access, Reserved access, Scheduled access.	1
	Multi-channel multiple access protocols, Desirable character sticks of protocol, Scalability, Fairness, TTTR, TTFR, FTTR, FTFR, Problem of wavelength stability.	2
	Multi hop WDM network, Shuffle net, MSN.	2
	Wavelength routed networks, Mesh, Ring-Traffic grooming problem.	3
4.	**Unit 4**	**4L**
	IPover Optical framework, ASON, MPëS, Burst switched network (buffer less networks), All-optical circuit switches, All-optical packet switches. Broadcast and select. Wavelength routed. Spaces witch based. Discussion on various switch architectures. Packet buffering techniques. Travelling type. Recirculating type.	2

	Protection and restoration., Restoration mechanism. Restoration timing issues. Path protection. Span protection. P-cycles	2
Total Number Of Hours = 31		

Faculty In-Charge HOD, CSE Dept.

Assignment:

Assignment 1:

1. A silica optical fiber with a core diameter large enough to be considered by ray theory analysis has a core refractive index of 1.50 and a cladding refractive index of 1.47. Determine: (a) the critical angle at the core–cladding interface; (b) the NA for the fiber; (c) the acceptance angle in air for the fiber.
2. A typical relative refractive index difference for an optical fiber designed for longdistance transmission is 1%. Estimate the NA and the solid acceptance angle in air for the fiber when the core index is 1.46. Further, calculate the critical angle at the core–cladding interface within the fiber. It may be assumed that the concepts of geometric optics hold for the fiber.
3. An optical fiber in air has an NA of 0.4. Compare the acceptance angle for meridional rays with that for skew rays which change direction by 100° at each reflection.
4. Discuss about different mode of operation.
5. A graded index fiber with a parabolic refractive index profile core has a refractive index at the core axis of 1.5 and a relative index difference of 1%. Estimate the maximum possible core diameter which allows single-mode operation at a wavelength of 1.3 μm.
6. Determine the cutoff wavelength for a step index fiber to exhibit single-mode operation when the core refractive index and radius are 1.46 and 4.5μm, respectively, with the relative index difference being 0.25%.

Assignment 2:

1. What are direct and indirect bandbackmaterial?Which one is used for LED and why?
2. Explain the structure,material,characteristics and modulation of light emitting diode used as optical sources.
3. What are the desirable properties of an optical source for optical communication application.Define emission response time and quantum efficiency of an optical source.
4. Define and differentiate external and internal quantum efficiency of an LED.
5. What is meant by optical confinement of light in LASER.How it is oriented,explain.
6. Compare LED and LASER sources used in optical fiber communication.
7. What is meant by stimulated emission of light in LASER.Describe with neat sketches the working of Ditributed Feedback LASER structures and its characteristics.
8. Explain the various types of LASER with its different configuration diagram and modals.
9. Explain the construction and working of a heterojunction of LED.

UNIVERSITY OF ENGINEERING & MANAGEMENT,JAIPUR

Lecture-wise Plan

Assignment 3:

1. Explain the detection process in the p–n photodiode. Compare this device with the p–i–n photodiode.
2. A p–i–n photodiode on average generates one electron–hole pair per three incident photons at a wavelength of 0.8 μm. Assuming all the electrons are collected calculate:
   (a) the quantum efficiency of the device;
   (b) its maximum possible bandgap energy;
   (c) the mean output photocurrent when the received optical power is 10^{-7}W.
3. Define the quantum efficiency and the responsivity of a photodetector.
4. Derive an expression for the responsivity of an intrinsic photodetector in terms of the quantum efficiency of the device and the wavelength of the incident radiation.
5. Determine the wavelength at which the quantum efficiency and the responsivity are equal.

Assignment 4:

1. Give the major reasons which have led to the development of optical amplifiers, outlining the attributes and application areas for these devices.
2. Describe the two main SOA types and indicate their distinguishing features.
3. Explain the gain process in a Raman fiber amplifier and comment upon the flexibility associated with the pumping process in this fiber amplifier type.

UNIVERSITY OF ENGINEERING & MANAGEMENT,JAIPUR

Lecture-wise Plan

Subject Name: Low Power Circuits & Systems **Subject Code-CS802D**
Year: 4th Year **Semester: Eighth**

Module Number	Topics	Number of Lectures
1	**Basics of MOS circuits:**	**6L**
	1. MOS Transistor structure and device modelling	4L
	2. MOS Inverters	1L
	3. MOS Combinational Circuits–Different Logic Families.	3L
2	**Sources of Power dissipation:**	**8L**
	1. Dynamic Power Dissipation	2L
	2. Short Circuit Power	2L
	3. Switching Power	2L
	4. Gliching Power	1L
	5. Static Power Dissipation	1L
3	**Supply Voltage Scaling Approaches**	**8L**
	1. Device feature size scaling	1L
	2. Multi-Vdd Circuits	2L
	3. Architectural level approaches: Parallelism, Pipelining	2L
	4. Voltage scaling using high-level transformations: Dynamic voltage scaling	2L
	5. Power Management.	1L
4	**Switched Capacitance Minimization Approaches**	**14L**
	1. Hardware Software Trade-off	3L
	2. Bus Encoding	2L
	3. Two's complement Vs Sign Magnitude	2L
	4. Architectural optimization	2L
	5. Clock Gating; Logic styles	2L
Total Number Of Hours = 36		

Faculty In-Charge HOD, ECE Dept.

Assignment 1:

1. Why low power has become an important issue inthe present day VLSI circuit realization?
2. How reliability of a VLSI circuit is related to itspower dissipation?
3. How environment is affected by the powerdissipation of VLSI circuits?
4. Why leakage power dissipation has become animportant issue in deep submicron technology?
5. Distinguish between energy and power dissipationof VLSI circuits. Which one is more important forportable systems?
6. Explain the three modes of operation of a MOStransistor.
7. What is the threshold voltage of a MOS transistor? How itvaries with the body bias?
8. The following parameters are given for an nMOS process: t_{ox} 500Å, $N_A = 1x1016cm^{-3}$, $N_D = 1x1020cm^{-3}$, $N_{OX} = 2x1010cm^{-1}$. (i) Calculate V_t for an unimplanted transistor, (ii) what type and what concentration must be implanted to achieve V_t = +1.5V and V_t = - 2.0V?
9. What is channel length modulation effect? How the voltagecurrent characteristics are affected because of this effect?
10. What is body effect? How does it influences the thresholdvoltage of a MOS transistor?
11. What is transconductance of a MOS transistor? Explain its role in the operation of the transistor.

Assignment 2:

1. Explain the behaviour of anMOS transistor as a switch.
2. Explain the behaviour of a pMOS transistor as a switch.
3. How one nMOS and one pMOS transistor are combined tobehave like an ideal switch.
4. The input of a lightly loaded transmission gate is slowlychanges from HIGH level to LOW level. How the currents throughthe two transistors vary?
5. How its ON-resistance of a transmission gate changes as theinput varies from 0 V to Vdd, when the output has a light capacitiveload.
6. Draw the ideal characteristics of a CMOS inverterand compare it with the actual characteristics.
7. What is noise margin? Find out the noise marginfrom the actual characteristics of the inverter.
8. Compare the characteristics of the different typesof MOS inverters in terms of noise margin and powerdissipation.
9. What is the inversion voltage of an inverter? Findout the inversion voltage of a CMOS inverter.
10. How the inversion voltage is affected by the relativesizes of the sizes of the nMOS and pMOS transistors of the CMOS transistors of the CMOSinverter?
11. Find out the noise margin of a CMOS inverter.
12. How the noise margin is affected by voltagescaling?
13. What is the lower limit of supply voltage of a CMOSinverter?
14. What happens if the supply voltage is inverter?
15. What happens if the supply voltage is furtherreduced?
16. What is sheet resistance? Find out the expressionof the resistance of rectangular sheet in terms of sheetresistance.
17. Find out the capacitance of a MOS capacitor.
18. Find out the expression of delay time of a CMOSinverter.

19. What are the various ways to reduce the delay timeof a CMOS inverter?
20. Explain the commonly used technique to estimatethe delay time of a CMOS inverter.

Assignment 3:

1. How the transfer characteristic of a CMOS NAND gate is affectedwith increase in fan with increase in fan-in?
2. How the transfer characteristic of a CMOS NOR gate is affectedwith increase in fan with increase in fan-in?
3. How switching characteristic of a CMOS NAND gate is affectedwith increase in fan with increase in fan-in?
4. How switching characteristic of a CMOS NOR gate is affectedwith increase in fan with increase in fan-in?
5. How noise margin of a CMOS NAND/NOR gate is affected withincrease in fan increase in fan-in?
6. For a complex/compound CMOS logic gate, how do you realize the pull the pull-up and the pull up and the pull-down networks? down networks?
7. Give the two possible topologies AND. Give the two possible topologies AND-OR-INVERT AND-OR- ERT ANDINVERT (AOI) and OR INVERT (AOI) and OR-AND-INVERT (OAI) to realize CMOS logic INVERT (OAI) to realize CMOS logicgate. Explain with an example.
8. Give the AOI and OAI realizations for the sum and carryfunctions of a full adder.
9. How do you realize pseudo nMOS logic circuits. Compare its logic circuits. Compare itsadvantage and disadvantages with respect to standard static CMOS circuits.
10. What is charge leakage problem of dynamic CMOS circuits?
11. How is it overcome?
12. What is charge sharing problem? How can it be overcome?
13. Explain the clock skew problem of dynamic CMOS circuits?
14. How clock skew problem is overcome in in domino CMOS circuits?
15. How clock skew problem is overcome in in NORA CMOS circuits?

Assignment 4:

1. Explain the basic concepts of supply voltage scaling.
2. As you move to a new process technology with a scaling factor S = 1.4, how the drain current, power density, delay and energy requirementchanges for the constant field scaling?
3. Distinguish between constant field and constant voltage feature size scaling? Compare their advantages and disadvantages.
4. Compare the constant field and constant voltage scaling approaches in
5. Compare the constant field and constant voltage scaling approaches in terms of area, delay, energy and power density parameters.
6. Explain how parallelism can be used to achieve low power instead of high performance in realizing digital circuits.
7. Explain how multicore architecture provides low power compared to the single core architecture of the same performance.
8. Explain the basic concept of multi-level voltage scaling.
9. What is the impact of multiple supply voltages on thedistribution of path delays of a circuit with respect to that for singlesupply voltage?
10. List and explain three important issues in the context ofmultiple supply voltage scaling?

11. What problem arises when a signal passes from low voltagedomain to high voltage domain? How this problem is overcome?
12. Explain the design decision for the placement of converters inthe voltage scaling interfaces.

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

Lecture-wise Plan

Subject Name: E-Commerce **Subject Code-CS802E**
Year: 4th Year **Semester: Eighth**

Module Number	Topics	Number of Lectures
1	**Introduction:**	**6L**
	1. Definition, Scope of E-Commerce, Hardware requirements.	2
	2. E-Commerce and Trade Cycle	1
	3. Electronic Markets	1
	4. Electronic Data Interchange and Internet Commerce	2
2	**Business to Business E-Commerce :**	**7L**
	1. Electronic Markets, Electronic Data Interchange (EDI): Technology, Standards (UN/EDIFACT),Communications, Implementations, Agreements, Security,	3
	2. 2. EDI and Business	2
	3. Inter-Organizational E-commerce.	2
3.	**Legal issues :**	**5L**
	1. Risks: Paper Document vs. Electronic document, Authentication of Electronic document, Laws.	2
	2. Legal issues for Internet Commerce : Trade marks and Domain names, Copyright, Jurisdiction issues, Service provider liability, Enforceable online contract.	3
4	**Security Issues :**	**6L**
	1. Security Solutions : Symmetric and Asymmetric Cryptosystems, RSA,DES ,and Digital Signature.	3
	2. Protocols for secure messaging	1
	3. Secure Electronic Transaction(SET) Protocol	1
	4. Electronic cash over internet, Internet Security.	1
5	**Business to Consumer E-Commerce :**	**8L**
	1. Consumer trade transaction.	2
	2. Internet	1
	3. Page on the Web.	1
	4. Elements of E-Commerce with VB,ASP	2
	5. SQL	2
6	**E-business :**	**7L**
	1. Internet book shops, Software supplies and support, Electronic Newspapers, Internet Banking.	2

	2. Virtual Auctions, Online Share Dealing, Gambling on the net	2
	3. E-Diversity	2
	4. Case studies through internet.	1
Total Number Of Hours = 39		

Faculty In-Charge HOD, CSE Dept.

Assignment:

1. What is E-Commerce?
2. Short Notes: EDI, RSA,DES and Digital Signature.
3. Definition of B2B, B2C, C2B, C2C.

UNIVERSITY OF ENGINEERING & MANAGEMENT,JAIPUR

Lecture-wise Plan

Subject Name: Robotics **Subject Code-CS802F**

Year: 4th Year **Semester: Eighth**

Module Number	Topics	Number of Lectures
1	**Unit 1: Introduction**	**1L**
	Brief history, types, classification and usage, Science and Technology of robots, Some useful websites, textbooks and research journals.	1
2	**Unit 2: Elements of robots–links, joints, actuators, and sensors**	**5L**
	Position and orientation of a rigid body, Homogeneous transformations, Representation of joints, link representation using D-H parameters, Examples of D-H parameters and link transforms, different kinds of actuators–stepper, DC servo and brushless motors, model of a DC servo motor, Types of transmissions, Purpose of sensors, internal and external sensors, common sensors–encoders, tachometers, strain gauge based force-torque sensors, proximity and distance measuring sensors, and vision.	5
3	**Unit 3: Kinematics of serial robots**	**4L**
	Introduction, Direct and inverse kinematics problems, Examples of kinematics of common Serial manipulators, workspace of a serial robot, Inverse kinematics of constrained and redundant robots, Tractrix based approach for fixed and free robots and multi-body systems, simulations and experiments, Solution procedures using theory of elimination, Inverse kinematics solution for the general 6R serial manipulator.	4
4	**Unit 4: Kinematics of parallel robots**	**5L**
	Degrees-of- freedom of parallel mechanisms and manipulators, Active and passive joints, Constraint and loop- closure equations, Direct kinematics problem, Mobility of parallel manipulators, Closed- from and numerical solution, Inverse kinematics of parallel manipulators and mechanisms, Direct kinematics of Gough-Stewart platform.	5
5	**Unit 5: Velocity and static analysis of robot manipulators**	**5L**
	Linear and angular velocity of links, Velocity propagation, Manipulator Jacobians for serial and parallel manipulators, Velocity ellipse and ellipsoids, Singularity analysis for serial and parallel manipulators, Loss and gain of degree of freedom, Statics of serial and parallel manipulators, Statics and force transformation matrix of a Gough- Stewart platform, Singularity analysis and statics.	5

6	**Unit 6: Dynamics of serial and parallel manipulators**	**4L**
	Mass and inertia of links, Lagrangian formulation for equations of motion for serial and parallel manipulators, Generation of symbolic equations of motion using a computer, Simulation (direct and inverse) of dynamic equations of motion, Examples of a planar 2R and four-bar mechanism, Recursive dynamics, Commercially available multi-body simulation software (ADAMS) and Computer algebra software Maple.	4
7	**Unit 7: Motion planning and control**	**6L**
	Joint and Cartesian space trajectory planning and generation, Classical control concepts using the example of control of a single link, Independent joint PID control, Control of a multi-link manipulator, Non-linear model based control schemes, Simulation and experimental case studies on serial and parallel manipulators, Control of constrained manipulators, Cartesian control, Force control and hybrid position/ force control, Advanced topics in non-linear control of manipulators.	6
8	**Unit 8: Modelling and control of flexible robots**	**4L**
	Models of flexible links and joints, Kinematic modelling of multi-link flexible robots, Dynamics and control of flexible link manipulators, Numerical simulations results, Experiments with a planar two-link flexible manipulator.	4
9	**Unit 9: Modelling and analysis of wheeled mobile robots**	**3L**
	Introduction and some well-known wheeled mobile robots(WMR), two and three-wheeled WM Ron flat surfaces, Slip and its modelling, WMRon uneven terrain, Design of slip-free motion on uneven terrain, Kinematics, dynamics and static stability of a three-wheeled WMR'son uneven terrain, Simulations using Matlab and ADAMS.	3
10	**Unit 10: Selected advanced topics in robotics**	**3L**
	Introduction to chaos, Non-linear dynamics and chaos in robot equations, Simulations of Planar 2 DOF manipulators, Analytical criterion for unforced motion. Gough-Stewart platform and its singularities, use of near singularity for fine motion for sensing, design of Gough-Stewart platform based sensors. Over-constrained mechanisms and deployable structures, Algorithm to obtain redundant links and joints, Kinematics and statics of deployable structures with pantographs or scissor-like elements(SLE's).	3
Total Number Of Hours = 40		

UNIVERSITY OF ENGINEERING & MANAGEMENT,JAIPUR

Lecture-wise Plan

Faculty In-Charge HOD, CSE Dept.

UNIVERSITY OF ENGINEERING AND MANAGEMENT, JAIPUR
Lab Manual

Title of Course: Parallel Computing Lab
Course Code: CS891B
L-T-P scheme: 0-0-3 **Course Credit: 2**

Objectives:
To learn and understand the concepts of parallel processing, designing the parallel algorithm, interfacing and communication between multiple processors, analyzing the complexity of parallel algorithm.

Learning Outcomes: The students will have a detailed knowledge of the concepts of parallel processing, designing the parallel algorithm, interfacing and communication between multiple processors, analyzing the complexity of parallel algorithm. Upon the completion of Parallel Computing practical course, the student will be able to:

- **Understand** and implement basic services and functionalities of parallel processing.
- **To understand** the interfacing between multiple computers.
- **Understand** the benefits of thread over process and implement synchronized programs using multithreading concepts.
- **Analyze** the complexity of parallel algorithm
- **Design** and analysis of parallel algorithm
- **Simulate** sub programs in parallel through multiple computers.

.

Course Contents:
Exercises that must be done in this course are listed below:
Exercise No.1: Basic arithmetic operations in parallel
Exercise No. 2: Quick Sort through parallel processing
Exercise No. 3: Generation of Fibonacci series, finding prime numbers in an interval
Exercise No. 4: Matrix Multiplication by parallel computers
Exercise No. 5: Parallel Tree Traversals
Exercise No. 6: Enumeration sort
Exercise No. 7: Odd-Even Transposition sort

Text Book:
1. T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, “Introduction to Algorithms”
2. E.Horowitz and Shani “Fundamentals of Computer Algorithms”

Recommended Systems/Software Requirements:
1. Intel based desktop PC with minimum of 166 MHZ or faster processor with at least 64 MB RAM and 100 MB free disk space.
2. Turbo C or TC3 complier in Windows XP or Linux Operating System.

Experiment No:1 BASIC ARITHMETIC OPERATION IN PARALLEL

Aim: To perform a set of arithmetic operations in parallel using a cluster of computers

Description: In computers, parallel processing is the processing of program instructions by dividing them among multiple processors with the objective of running a program in less time. In the earliest computers, only one program ran at a time. A computation-intensive program that took one hour to run and a tape copying program that took one hour to run would take a total of two hours to run. An early form of parallel processing allowed the interleaved execution of both programs together. The computer would start an I/O operation, and while it was waiting for the operation to complete, it would execute the processor-intensive program. The total execution time for the two jobs would be a little over one hour.

Procedure:

Step 1: Identify a set of numbers over which arithmetic operations are to be done

Step 2: Identify the rank in the cluster

Step 3: Split the set of numbers into domains and assign each domain to their corresponding processor

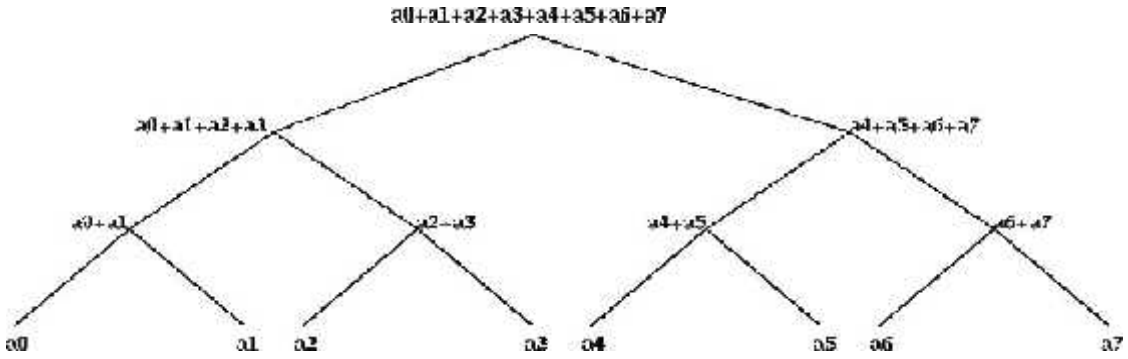
Step 4: Get the result from each computer in the cluster and assimilate them at the master

Step 5: Verify the time taken for the completion of each task.

Pseudo Code:-

```
function sum(+,identity,a) = if #a
== 1 then [identity] else
  let e = even_elts(a); o =
      odd_elts(a);
      s = scan_op(op,identity,{op(e,o): e in e; o in o}) in
  interleave(s,{op(s,e): s in s; e in e});
```

Input:- An array of integers



Output: - The sum

3,2,7,6	0,5,4,8	2,0,1,5	2,3,8,6
P1	P2	P3	P4
18	17	8	19
	62		

Experiment No-2 QUICK SORT THROUGH PARALLEL PROCESSING

Aim: To sort the list of numbers using Quick sort by parallel computers.

Description: Quicksort is generally recognized as the fastest sorting algorithm based on comparison of keys, in the average case. Quicksort has some natural concurrency. The low list and high list can sort themselves concurrently. We consider the case of distributed memory.Each process holds a segment of the unsorted list.The unsorted list is evenly distributed among the processes. Desired result of a parallel quicksort algorithm:The list segment stored on each process is sorted. The last element on process-I's list is smaller than the first element on process I+1's list.

Example:

Given a list of numbers, we want to sort the numbers in an increasing order. The same as finding a suitable permutation. Sequential quicksort algorithm: a recursive procedure. Select one of the numbers as pivot. Divide the list into two sublists: a "low list" containing numbers smaller than the pivot, and a "high list" containing numbers larger than the pivot. The low list and high list recursively repeat the procedure to sort themselves. The final sorted result is the concatenation of the sorted low list, the pivot, and the sorted high list.

Given a list of numbers:{79,17,14,65,89,4,95,22,63,11}

The first number, 79, is chosen as pivot Low list contains

{17, 14, 65, 4, 22, 63, 11}

High list contains {89, 95}

For sublist {17, 14, 65, 4, 22, 63, 11}, choose 17 as pivot

Low list contains {14, 4, 11 }

High list contains {64,22,63}. . .{4,11,14,17,22,63,65} is the sorted result of sublist

{17, 14, 65, 4, 22, 63, 11}

For sublist {89, 95}

Choose 89 as pivot

Low list is empty (no need for further recursions)

High list contains {95}

(no need for further recursions)

{89, 95} is the sorted result of sublist {89, 95}

Final sorted result:

{4, 11, 14, 17, 22, 63, 65, 79, 89, 95}

Procedure:

Step 1: Accept the list of numbers
Step 2: Split the set of numbers into domains and assign each domain to their corresponding processor.
Step 3: sort the numbers using Quick sort.
Step 4: Verify the time taken for the completion of each task

Pseudo code:-

```
function quicksort(a) = if (#a <2)
then a
else
  let pivot  = a[#a/2];
      lesser = {e in a| e < pivot};
      equal  = {e in a| e == pivot};
      greater = {e in a| e > pivot};
      result = {quicksort(v): v in [lesser,greater]};
  in result[0] ++ equal ++ result[1];
```

Input:-

P1	P2	P3	P4
11,50,53,44	35,16,1,23	97,48,96,8	58,76,54,21

Here P1, P2, P3 and P4 be the processors

Output:-

P1	P2	P3	P4
1,8,11,16	21,23,35,44	48,50,53,54	58,76,,96,97

Experiment-3 GENERATION OF FIBONACCI SERIES AND FINDING PRIME NUMBERS IN AN INTERVAL.

Aim: To generate the Fibonacci series of numbers and finding a set prime number in a given interval

Description: The Fibonacci sequence is a set of numbers that starts with a one or a zero, followed by a one, and proceeds based on the rule that each number (called a Fibonacci number) is equal to the sum of the preceding two numbers. If the Fibonacci sequence is denoted F (n), where n is the first term in the sequence, the following equation obtains for n = 0, where the first two terms are defined as 0 and 1 by convention.

F (0) = 0, 1, 1, 2, 3, 5, 8, 13, 21, 34 ...

In some texts, it is customary to use n = 1. In that case, the first two terms are defined as 1 and 1 by default, and therefore:

F (1) = 1, 1, 2, 3, 5, 8, 13, 21, 34 …..

A prime number is a whole number greater than 1, whose only two whole-number factors are 1 and itself. The first few prime numbers are 2, 3, 5, 7, 11, 13, 17, 19, 23, and 29.

As we proceed in the set of natural numbers **N** = {1, 2, 3, ...}, the primes become less and less frequent in general. However, there is no largest prime number. For every prime number *p*, there exists a prime number *p*' such that *p*' is greater than *p*. This was demonstrated in ancient times by the Greek mathematician Euclid.

Procedure:

Fibonacci

Step 1 : Identify a maximum limit up to which Fibonacci series has to be generated.
Step 2: Identify the rank
Step 3: if rank==1 then apply the formula , fib1[i]=fib1[i-1]+fib1[i-2]
Step 4: if rank==2 then apply the formula,
fib2[0] = (pow((1+sqrt(5))/2,9) – pow((1-sqrt(5))/2,9))/sqrt(5) fib2[1] =
(pow((1+sqrt(5))/2,10) – pow((1-sqrt(5))/2,10))/sqrt(5)
Step 4: Get the result from each computer in the cluster and assimilate them at the server.
Step 5: Verify the time taken for the completion of each task.

Prime

```
Procedure PRIME(n);
Let A be an array of length n
Set all but the first element of A to TRUE
For I from 2 to sqrt(n)
Begin:
      If A[i] is TRUE
      Then set all multiples of I up to n to FALSE
End.
```

Experiment-4 MATRIX MULTIPLICATION

Aim: To perform matrix multiplication using a cluster of computers

Description: The product C of two matrices A and B is defined as

$C_{ik} = A_{ij} B_{jk}$

where j is summed over for all possible values of I and k and the notation above uses the Einstein summation convention. The implied summation over repeated indices without the presence of an explicit sum sign is called Einstein summation, and is commonly used in both matrix and tensor analysis. Therefore, in order for matrix multiplication to be defined, the dimensions of the matrices must satisfy

(n X m) (m X p)= (n X p) where (a X b) denotes a matrix with a rows and b columns. Writing out the product explicitly,

$$\begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1p} \\ c_{21} & c_{22} & \cdots & c_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{np} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mp} \end{bmatrix},$$

where

$$\begin{aligned}
c_{11} &= a_{11} b_{11} + a_{12} b_{21} + \ldots + a_{1m} b_{m1} \\
c_{12} &= a_{11} b_{12} + a_{12} b_{22} + \ldots + a_{1m} b_{m2} \\
c_{1p} &= a_{11} b_{1p} + a_{12} b_{2p} + \ldots + a_{1m} b_{mp} \\
c_{21} &= a_{21} b_{11} + a_{22} b_{21} + \ldots + a_{2m} b_{m1} \\
c_{22} &= a_{21} b_{12} + a_{22} b_{22} + \ldots + a_{2m} b_{m2} \\
c_{2p} &= a_{21} b_{1p} + a_{22} b_{2p} + \ldots + a_{2m} b_{mp} \\
c_{n1} &= a_{n1} b_{11} + a_{n2} b_{21} + \ldots + a_{nm} b_{m1} \\
c_{n2} &= a_{n1} b_{12} + a_{n2} b_{22} + \ldots + a_{nm} b_{m2} \\
c_{np} &= a_{n1} b_{1p} + a_{n2} b_{2p} + \ldots + a_{nm} b_{mp}.
\end{aligned}$$

Procedure:

Step 1: Accept two Matrices
Step 2: Decide on the number of processors in the cluster.
Step 3: Send the matrices and the range of values to be processed by each processor
Step 4: Form the resultant matrix at the master end
Step 5: Verify the time taken for the completion of each task

Pseudo Code:

```
Begin
For i <- 0 to l-1 do
For j <- 0 to n-1 do
T <- 0
For k <-0 to m-1 do
T <- T + a[i][k] X b[k][j]
```

```
End for
        C[i][j] <- T
        Endfor
        Endfor
End.
```

Experiment No-5 PARALLEL TREE TRAVERSAL

Aim: To perform tree traversal using a cluster of computers

Description: Three commonly used traversal methods for binary trees (forsets) are pre-order, in-order and post-order. It is well known that sequential algorithms for these traversals takes order O(N) time where N is the total number of nodes. This experiment establishes a one-to-one correspondence between the set of nodes that possess right sibling and the set of leaf nodes for any forest. For the case of pre-order traversal, this result is shown to provide an alternate characterization that leads to a simple and elegant parallel algorithm of time complexity O(log N) with or without read-conflicts on an N processor SIMD shared memory model, where N is the total number of nodes in a forest.

Procedure:

Step 1: Create a binary tree for a given set of data
Step 2: Split the tree into left sub tree and right sub tree
Step 3: Assign them to one processor each
Step 4: Use any tree traversal method in processors
Step 4: Get the ordered data at the master end from each processor
Step 5: Verify the time taken for the completion of each task

Pseudo Code:-

PRE-ORDER:

```
preorder(T, v)
   visit node v if
T.hasLeft(v)
   preorder(T, T.left(v)) // recursively traverse left sub tree if T.hasRight(v)
   preorder(T, T.right(v)) // recursively traverse right sub tree
```

POST-ORDER

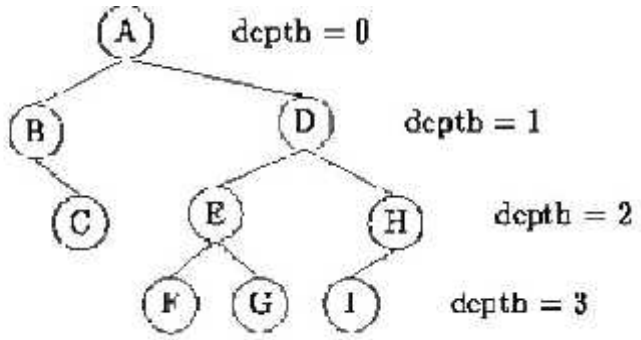
```
 Algorithm postorder(T, v)
   postorder(T, T.left(v))                  // recursively traverse left sub tree
 if T.hasRight(v)
   postorder(T, T.right(v))  // recursively traverse right sub tree
    visit node v
```

IN-ORDER

```
Algorithm inorder(T, v) if
 T.hasLeft(v)
   inorder(T, T.left(v)) // recursively traverse left subtree visit node v
if T.hasRight(v)
   inorder(T, T.right(v)) // recursively traverse right sub tree
```

INPUT:
A binary tree T and a node v of T.

$$T = \{A, \{B, \emptyset, \{C, \emptyset, \emptyset\}\}, \{D, \{E, \{F, \emptyset, \emptyset\}, \{G, \emptyset, \emptyset\}\}, \{H, \{I, \emptyset, \emptyset\}, \emptyset\}\}\}$$



OUTPUT
A pre order traversal of the tree shown in Figure visits the nodes in the following order:

$$A, B, C, D, E, F, G, H, I.$$

A post order traversal of the tree shown in Figure visits the nodes in the following order:

$$C, B, F, G, E, I, H, D, A.$$

An in order traversal of the tree shown in Figure visits the nodes in the following order:

$$B, C, A, F, E, G, D, I, H.$$

Experiment No-6 ENUMERATION SORT

Aim: To sort the list of numbers using enumeration sort.

Description: Enumeration sort is a method of arranging all the elements in a list by finding the final position of each element in a sorted list. It is done by comparing each element with all other elements and finding the number of elements having smaller value. Therefore, for any two elements, a_i and a_j any one of the following cases must be true −

- $a_i < a_j$
- $a_i > a_j$
- $a_i = a_j$

Example:

Algorithm:

```
procedure ENUM_SORTING (n)
begin
  for each process P1,j do
    C[j] := 0;
 for each process Pi, j do
 if (A[i] < A[j]) or A[i] = A[j] and i < j) then
       C[j] := 1;
    else
       C[j] := 0;
for each process P1, j do
    A[C[j]] := A[j];
end ENUM_SORTING
```

Procedure:

Step 1: Accept the list of numbers
Step 2: Split the set of numbers into domains and assign each domain to their corresponding processor.
Step 3: sort the numbers using enumeration sort.
Step 4: Verify the time taken for the completion of each task

Pseudo Code - Enumeration Sort

```
Step 1: For i ← 1 to n do
                        do Step3 (create thread)
Step 2: Wait for the completion of all the processes
Step 3: At Processor i

   e ← x[i]
   k ← 1
     for j ← 1 to n do
           if e >= x[j] and (e != x[j] or i>j) then
                k ← k+1
                R[k] ← e
```

Experiment No-7 ODD-EVEN TRANSPOSITION SORT

Aim: To sort the list of numbers using Odd-Even transposition sort.

Description: Odd-Even Transposition Sort is based on the Bubble Sort technique. It compares two adjacent numbers and switches them, if the first number is greater than the second number to get an ascending order list. The opposite case applies for a descending order series. Odd-Even transposition sort operates in two phases − odd phase and even phase. In both the phases, processes exchange numbers with their adjacent number in the right.

Procedure:

Step 1: Accept the list of numbers

Step 2: Split the set of numbers into domains and assign each domain to their corresponding processor.

Step 3: sort the numbers using Odd-Even transposition sort.

Step 4: Verify the time taken for the completion of each task

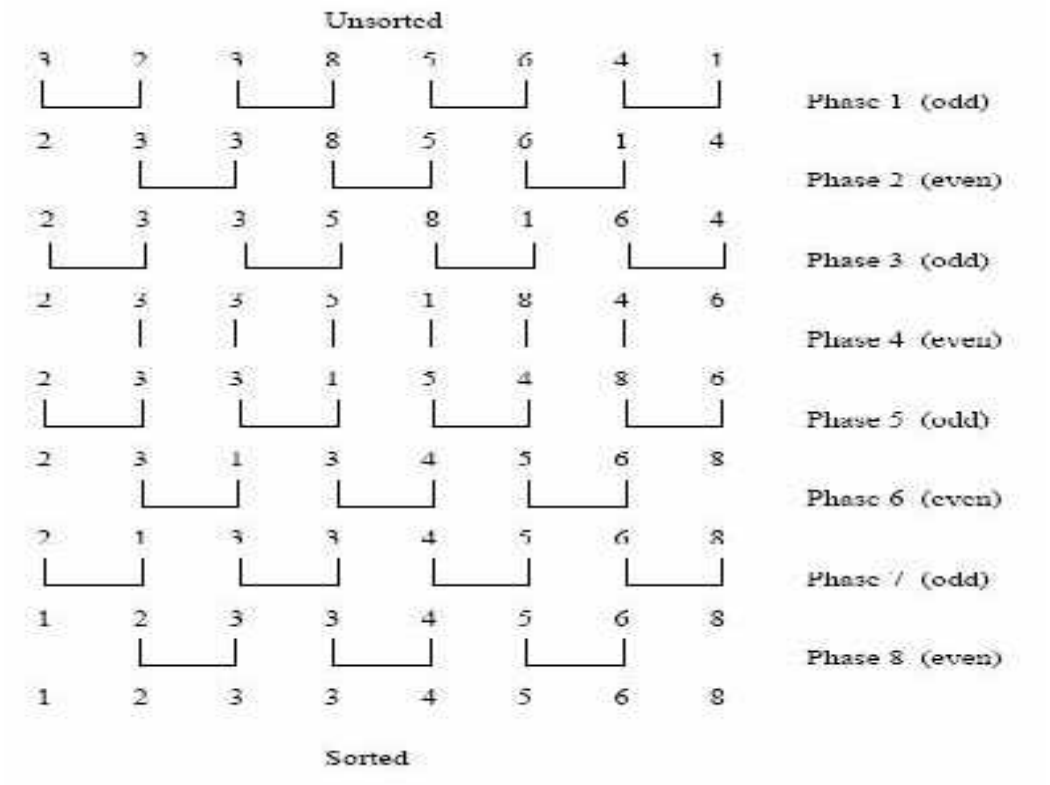
Sequential odd-even transposition sorts Pseudo Code:

```
Procedure ODD-EVEN (n)
Begin
    For i:=1 to n do
      Begin
        If i is odd then
            For j:= 0 to n/2-1 do
                Compare-exchange (a2j+1, a2j+2)

        If  i is even then

            For  j:=1 to n/2-1 do
              Compare-exchange (a2j, a2j+1);
        End for
End ODD-EVEN
```

Example:



UNIVERSITY OF ENGINEERING AND MANAGEMENT, JAIPUR

Lab Manual

UNIVERSITY OF ENGINEERING AND MANAGEMENT, JAIPUR

Lab Manual

UNIVERSITY OF ENGINEERING AND MANAGEMENT, JAIPUR

Lab Manual

UNIVERSITY OF ENGINEERING AND MANAGEMENT, JAIPUR

Lab Manual

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

Lab Manual

Title of Course: Natural Language Processing Lab
Course Code: CS891C
L-T-P scheme: 0-0-3 **Course Credit: 3**

Objectives:
The objective of Natural Language Processing lab is to introduce the students with the basics of NLP which will empower them for developing advanced NLP tools and solving practical problems in the field.

Learning Outcomes: The experiments in this lab are arranged in a logical sequence to inculcate a new concept at every step, starting from very basic ones to advanced ones.
The students by taking up projects on these domains will learn then once of these domains which will definitely help them better in landing up in such companies or continue their Masters/Research in the similar or related domains.

Course Contents:
Exercises that must be done in this course are listed below:

Exercise No.1: Word Analysis
Exercise No. 2: Word Generation
Exercise No. 3: Morphology
Exercise No. 4: N-Grams
Exercise No. 5: N-Grams Smoothing
Exercise No. 6: POS Tagging: Hidden Markov Model
Exercise No. 7: POS Tagging: Viterbi Decoding
Exercise No. 8: Building POS Tagger
Exercise No. 9: Chunking
Exercise No. 10: Building Chunker

Text Book:
1. Akshar Bharati, Rajeev Sangal and Vineet Chaitanya: "Natural Language Processing: A Paninian Perspective", Prentice-Hall of India , New Delhi, 1995.

Recommended Systems/Software Requirements:

1. Six PCs with the following specification
2. Intel Core i3 (4th Gen) Processor
3. 19.5 Inches Screen
4. Touch screen Support
5. 1 TB Hard Drive
6. 4 GB DDR3 RAM
7. Windows 8.1
8. Two LAN connections for the Pcs.
9. Wireless Internet access.

Experiment No: 1. Word Analysis

Objective:
The objective of the experiment is to learn about morphological features of a word by analysing it.

A word can be simple or complex. For example, the word 'cat' is simple because one cannot further decompose the word into smaller part. On the other hand, the word 'cats' is complex, because the word is made up of two parts: root 'cat' and plural suffix '-s'

- **Further Readings**

1. **Speech and Language Processing** - ***An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition*** BY: Daniel Jurafsky and James H. Martin - *Chapter 3*

2. **Natural Language Processing** - ***A Paninian Perspective*** BY: Akshar Bharti, Vineet Chaitanya and Rajeev Sangal - *Chapter 3*

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR
Lab Manual

Title of Course: Cryptography & Network Security Lab
Course Code: CS891D
L-T-P Scheme: 0-0-3 **Course Credits: 3**

Introduction:

With the growth of the Internet, the need for secured data transmission increased manifold. In fact it became a pre-condition to the usage of the Internet for business transaction. Therefore security is a major concern in the Internet World.

Objectives:

We will study security from multiple perspectives. We will consider a variety of security policies, authentication before access, integrity of information, and confidentiality of information. The course will focus on the models, the tools, and the techniques for enforcement of security policies, with some emphasis on the use of cryptography. And because today's implementation approaches are typically flawed, we will also address the penetration and disruption of information systems in the context of operating systems and networks.

Learning Outcomes:
Knowledge:

1. You will understand the basic security services e.g.bAuthentication, Access Control, Confidentiality, Integrity, and Non repudiation)
2. You will understand the concepts of risk, threats, vulnerabilities and attack.
3. You will know the important ethical and legal issues to consider in computer security.
4. You will know the goals of end-to-end data security.
5. You will understand the role of random numbers and prime numbers in security.
You will learn standard symmetric encryption algorithms
6. You will learn the architecture for public and private key cryptography and how public key infrastructure (PKI) supports network security.
7. You will learn the methods of digital signature and encryption.
8. You will learn key management and how key exchange protocols work.
9. You will learn security protocols at different layers of Network layer heirachy.
10. You will learn futuristic cryptographic techniques like Eliptic Curve and quantum cryptography.
11. You will learn the concept of trusted computing.
12. You will learn the Web security Protocol.

Application :

1. Apply appropriate known cryptographic techniques for a given scenario.
2. You will be able to analyze the tradeoffs of balancing key security properties.
3 You will be able to design a security solution and do the cryptanalysis. .

Course Contents:

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR
Lab Manual

1. Encryption & Decryption using Cipher Algorithms

AIM:

Write a Java program to perform encryption and decryption using the following algorithms:

a) Ceaser Cipher

b) Substitution Cipher

c) Hill Cipher

PROGRAM:

a) Ceaser Cipher

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.Scanner;
public class CeaserCipher {

static Scanner sc=new Scanner(System.in);
static BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
public static void main(String[] args) throws IOException {
// TODO code application logic here

System.out.print("Enter any String: ");
String str = br.readLine();
System.out.print("\nEnter the Key: ");
int key = sc.nextInt();
String encrypted = encrypt(str, key);
System.out.println("\nEncrypted String is: " +encrypted);
String decrypted = decrypt(encrypted, key);
System.out.println("\nDecrypted String is: " +decrypted);
System.out.println("\n");
}

public static String encrypt(String str, int key) {
String encrypted = "";
for(int i = 0; i < str.length(); i++) {
int c = str.charAt(i);

if (Character.isUpperCase(c)) {
c = c + (key % 26);
if (c > 'Z')
c = c - 26;
}
else if (Character.isLowerCase(c)) {
c = c + (key % 26);
if (c > 'z')
c = c - 26;
```

```
}

encrypted += (char) c;
}
return encrypted;
}

public static String decrypt(String str, int key) {
String decrypted = "";
for(int i = 0; i < str.length(); i++) {
int c = str.charAt(i);
if (Character.isUpperCase(c)) {
c = c - (key % 26);
if (c < 'A')
c = c + 26;
}

else if (Character.isLowerCase(c)) {
c = c - (key % 26);
if (c < 'a')
c = c + 26;
}
decrypted += (char) c;
}
return decrypted;
}
}
```

Output:
Enter any String: Hello World
Enter the Key: 5
Encrypted String is: MjqqtBtwqi
Decrypted String is: Hello World

b) Substitution Cipher
PROGRAM:

```
import java.io.*;
import java.util.*;
public class SubstitutionCipher {

static Scanner sc = new Scanner(System.in);
static BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
public static void main(String[] args) throws IOException {

// TODO code application logic here
String a = "abcdefghijklmnopqrstuvwxyz";
String b = "zyxwvutsrqponmlkjihgfedcba";

System.out.print("Enter any string: ");
String str = br.readLine();
String decrypt = "";
char c;
for(int i=0;i<str.length();i++)
```

```
{
c = str.charAt(i);
int j = a.indexOf(c);
decrypt = decrypt+b.charAt(j);
}

System.out.println("The encrypted data is: " +decrypt);
}
}
```

Output:

```
Enter any string: aceho
The encrypted data is: zxvsl
```

c) Hill Cipher

PROGRAM:

```
import java.io.*;
import java.util.*;
import java.io.*;
public class HillCipher {
static float[][] decrypt = new float[3][1];
static float[][] a = new float[3][3];
static float[][] b = new float[3][3];
static float[][] mes = new float[3][1];
static float[][] res = new float[3][1];

static BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
static Scanner sc = new Scanner(System.in);
public static void main(String[] args) throws IOException {
// TODO code application logic here
getkeymes();

for(int i=0;i<3;i++)
for(int j=0;j<1;j++)
for(int k=0;k<3;k++) {
res[i][j]=res[i][j]+a[i][k]*mes[k][j]; }
System.out.print("\nEncrypted string is : ");
for(int i=0;i<3;i++) {

System.out.print((char)(res[i][0]%26+97));
res[i][0]=res[i][0];
}
inverse();
for(int i=0;i<3;i++)
for(int j=0;j<1;j++)
for(int k=0;k<3;k++) {

decrypt[i][j] = decrypt[i][j]+b[i][k]*res[k][j]; }
System.out.print("\nDecrypted string is : ");
for(int i=0;i<3;i++){
System.out.print((char)(decrypt[i][0]%26+97));
}

System.out.print("\n");
```

```
}
public static void getkeymes() throws IOException {
System.out.println("Enter 3x3 matrix for key (It should be inversible): ");
for(int i=0;i<3;i++)
for(int j=0;j<3;j++)
a[i][j] = sc.nextFloat();

System.out.print("\nEnter a 3 letter string: ");
String msg = br.readLine();
for(int i=0;i<3;i++)
mes[i][0] = msg.charAt(i)-97;
}

public static void inverse() {
floatp,q;
float[][] c = a;
for(int i=0;i<3;i++)
for(int j=0;j<3;j++) {
//a[i][j]=sc.nextFloat();
if(i==j)
b[i][j]=1;
else b[i][j]=0;
}

for(int k=0;k<3;k++) {
for(int i=0;i<3;i++) {
p = c[i][k];
q = c[k][k];
for(int j=0;j<3;j++) {
if(i!=k) {
c[i][j] = c[i][j]*q-p*c[k][j];
b[i][j] = b[i][j]*q-p*b[k][j];
} } } }

for(int i=0;i<3;i++)
for(int j=0;j<3;j++) {
b[i][j] = b[i][j]/c[i][i]; }

System.out.println("");
System.out.println("\nInverse Matrix is : ");
for(int i=0;i<3;i++) {
for(int j=0;j<3;j++)
System.out.print(b[i][j] + " ");
System.out.print("\n"); }
} }
```

Output:

```
Enter a 3 letter string: hai
Encrypted string is :fdx
Inverse Matrix is :
0.083333336 0.41666666 -0.33333334
-0.41666666 -0.083333336 0.6666667
0.5833333 -0.083333336 -0.33333334
```

Decrypted string is :hai

2.Java program for DES algorithm logic

AIM: Write a Java program to implement the DES algorithm logic.

PROGRAM:

```
import java.util.*;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.security.spec.KeySpec;
import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;

import javax.crypto.spec.DESedeKeySpec;
import sun.misc.BASE64Decoder;
import sun.misc.BASE64Encoder;
public class DES {

private static final String UNICODE_FORMAT = "UTF8";
public static final String DESEDE_ENCRYPTION_SCHEME = "DESede";
privateKeySpecmyKeySpec;
privateSecretKeyFactorymySecretKeyFactory;
private Cipher cipher;
byte[] keyAsBytes;
private String myEncryptionKey;
private String myEncryptionScheme;
SecretKey key;

static BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
public DES() throws Exception {
// TODO code application logic here
myEncryptionKey = "ThisIsSecretEncryptionKey";
myEncryptionScheme = DESEDE_ENCRYPTION_SCHEME;
keyAsBytes = myEncryptionKey.getBytes(UNICODE_FORMAT);
myKeySpec = new DESedeKeySpec(keyAsBytes);
mySecretKeyFactory = SecretKeyFactory.getInstance(myEncryptionScheme);
cipher = Cipher.getInstance(myEncryptionScheme);
key = mySecretKeyFactory.generateSecret(myKeySpec);
}
public String encrypt(String unencryptedString) {
String encryptedString = null;
try {
cipher.init(Cipher.ENCRYPT_MODE, key);
byte[] plainText = unencryptedString.getBytes(UNICODE_FORMAT);
byte[] encryptedText = cipher.doFinal(plainText);
BASE64Encoder base64encoder = new BASE64Encoder();

encryptedString = base64encoder.encode(encryptedText); }
catch (Exception e) {
```

```
e.printStackTrace(); }
returnencryptedString; }
public String decrypt(String encryptedString) {
String decryptedText=null;
try {

cipher.init(Cipher.DECRYPT_MODE, key);
BASE64Decoder base64decoder = new BASE64Decoder();
byte[] encryptedText = base64decoder.decodeBuffer(encryptedString);
byte[] plainText = cipher.doFinal(encryptedText);
decryptedText= bytes2String(plainText); }
catch (Exception e) {
e.printStackTrace(); }
returndecryptedText; }
private static String bytes2String(byte[] bytes) {
StringBufferstringBuffer = new StringBuffer();
for (int i = 0; i <bytes.length; i++) {

stringBuffer.append((char) bytes[i]); }
returnstringBuffer.toString(); }
public static void main(String args []) throws Exception {
System.out.print("Enter the string: ");
DES myEncryptor= new DES();
String stringToEncrypt = br.readLine();
String encrypted = myEncryptor.encrypt(stringToEncrypt);


String decrypted = myEncryptor.decrypt(encrypted);
System.out.println("\nString To Encrypt: " +stringToEncrypt);
System.out.println("\nEncrypted Value : " +encrypted);
System.out.println("\nDecrypted Value : " +decrypted);
System.out.println("");
}
}
```

OUTPUT:

```
Enter the string: Welcome
String To Encrypt: Welcome
Encrypted Value : BPQMwc0wKvg=
Decrypted Value : Welcome
```

3. Program to implement BlowFish algorithm logic

AIM: Write a C/JAVA program to implement the BlowFish algorithm logic.

PROGRAM:

```
import java.io.*;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.security.Key;
import javax.crypto.Cipher;
import javax.crypto.CipherOutputStream;
import javax.crypto.KeyGenerator;
```

```
import sun.misc.BASE64Encoder;
public class BlowFish {

public static void main(String[] args) throws Exception {
// TODO code application logic here
KeyGeneratorkeyGenerator = KeyGenerator.getInstance("Blowfish");
keyGenerator.init(128);
Key secretKey = keyGenerator.generateKey();
Cipher cipherOut = Cipher.getInstance("Blowfish/CFB/NoPadding");
cipherOut.init(Cipher.ENCRYPT_MODE, secretKey);
BASE64Encoder encoder = new BASE64Encoder();
byte iv[] = cipherOut.getIV();
if (iv != null) {

System.out.println("Initialization Vector of the Cipher: " + encoder.encode(iv)); }
FileInputStream fin = new FileInputStream("inputFile.txt");
FileOutputStreamfout = new FileOutputStream("outputFile.txt");
CipherOutputStreamcout = new CipherOutputStream(fout, cipherOut);
int input = 0;
while ((input = fin.read()) != -1) {
cout.write(input); }
fin.close(); cout.close(); } }
```

OUTPUT:

```
Initialization Vector of the Cipher: dI1MXzW97oQ=
Contents of inputFile.txt: Hello World
Contents of outputFile.txt: ùJÖ˜ NåI“
```

4. Program to implement Rijndael algorithm logic

AIM: Write a C/JAVA program to implement the Rijndael algorithm logic.

PROGRAM:

```
import java.security.*;
import javax.crypto.*;
import javax.crypto.spec.*;
import java.io.*;
public class AES {
public static String asHex (byte buf[]) {
StringBuffer strbuf = new StringBuffer(buf.length * 2);
int i;
for (i = 0; i < buf.length; i++) {

if (((int) buf[i] & 0xff) < 0x10)
strbuf.append("0");
strbuf.append(Long.toString((int) buf[i] & 0xff, 16)); }
return strbuf.toString(); }
public static void main(String[] args) throws Exception {
String message="AES still rocks!!";
// Get the KeyGenerator
KeyGenerator kgen = KeyGenerator.getInstance("AES");
```

```
kgen.init(128); // 192 and 256 bits may not be available
// Generate the secret key specs.
SecretKey skey = kgen.generateKey();
byte[] raw = skey.getEncoded();
SecretKeySpec skeySpec = new SecretKeySpec(raw, "AES");
// Instantiate the cipher
Cipher cipher = Cipher.getInstance("AES");
cipher.init(Cipher.ENCRYPT_MODE, skeySpec);
byte[] encrypted = cipher.doFinal((args.length == 0 ? message : args[0]).getBytes());
System.out.println("encrypted string: " + asHex(encrypted));
cipher.init(Cipher.DECRYPT_MODE, skeySpec);
byte[] original = cipher.doFinal(encrypted);
String originalString = new String(original);
System.out.println("Original string: " + originalString + " " + asHex(original)); } }
```

5. Encrypt a string using BlowFish algorithm

AIM: Using Java Cryptography, encrypt the text “Hello world” using BlowFish. Create your own key using Java keytool.

PROGRAM:

```
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.swing.JOptionPane;
public class BlowFishCipher {

public static void main(String[] args) throws Exception {
// create a key generator based upon the Blowfish cipher
KeyGeneratorkeygenerator = KeyGenerator.getInstance("Blowfish");
// create a key
SecretKeysecretkey = keygenerator.generateKey();
// create a cipher based upon Blowfish
Cipher cipher = Cipher.getInstance("Blowfish");
// initialise cipher to with secret key
cipher.init(Cipher.ENCRYPT_MODE, secretkey);

// get the text to encrypt
String inputText = JOptionPane.showInputDialog("Input your message: ");
// encrypt message
byte[] encrypted = cipher.doFinal(inputText.getBytes());
// re-initialise the cipher to be in decrypt mode
cipher.init(Cipher.DECRYPT_MODE, secretkey);
// decrypt message
byte[] decrypted = cipher.doFinal(encrypted);
// and display the results

JOptionPane.showMessageDialog(JOptionPane.getRootFrame(),
"\nEncrypted text: " + new String(encrypted) + "\n" +
"\nDecrypted text: " + new String(decrypted));
```

```
System.exit(0);
} }
```

OUTPUT:

```
Input your message: Hello world
Encrypted text: 3ooo&&(*&*4r4
Decrypted text: Hello world
```

6. RSA Algorithm

AIM: Write a Java program to implement RSA Algoithm.

PROGRAM:

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.math.*;
import java.util.Random;
import java.util.Scanner;
public class RSA {

static Scanner sc = new Scanner(System.in);
public static void main(String[] args) {
// TODO code application logic here
System.out.print("Enter a Prime number: ");
BigInteger p = sc.nextBigInteger(); // Here's one prime number..
System.out.print("Enter another prime number: ");
BigInteger q = sc.nextBigInteger(); // ..and another.
BigInteger n = p.multiply(q);
BigInteger n2 = p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));
BigInteger e = generateE(n2);

BigInteger d = e.modInverse(n2); // Here's the multiplicative inverse
System.out.println("Encryption keys are: " + e + ", " + n);
System.out.println("Decryption keys are: " + d + ", " + n);
}
public static BigIntegergenerateE(BigIntegerfiofn) {
int y, intGCD;
BigInteger e;
BigInteger gcd;
Random x = new Random();
do {
y = x.nextInt(fiofn.intValue()-1);
String z = Integer.toString(y);
e = new BigInteger(z);
gcd = fiofn.gcd(e);
intGCD = gcd.intValue();
}
while(y <= 2 || intGCD != 1);
return e;
}
}
```

OUTPUT:

```
Enter a Prime number: 5
```

```
Enter another prime number: 11
Encryption keys are: 33, 55
Decryption keys are: 17, 55
```

7. Diffie-Hellman

AIM: Implement the Diffie-Hellman Key Exchange mechanism using HTML and JavaScript. Consider the end user as one of the parties (Alice) and the JavaScript application as other party (bob).

PROGRAM:

```
import java.math.BigInteger;
import java.security.KeyFactory;
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.SecureRandom;
import javax.crypto.spec.DHParameterSpec;
import javax.crypto.spec.DHPublicKeySpec;
public class DiffeHellman {
public final static int pValue = 47;
public final static int gValue = 71;
public final static int XaValue = 9;
public final static int XbValue = 14;
public static void main(String[] args) throws Exception {
// TODO code application logic here
BigInteger p = new BigInteger(Integer.toString(pValue));
BigInteger g = new BigInteger(Integer.toString(gValue));
BigIntegerXa = new BigInteger(Integer.toString(XaValue));
BigIntegerXb = new BigInteger(Integer.toString(XbValue));
createKey();
intbitLength = 512; // 512 bits
SecureRandomrnd = new SecureRandom();
p = BigInteger.probablePrime(bitLength, rnd);
g = BigInteger.probablePrime(bitLength, rnd);
createSpecificKey(p, g);
}
public static void createKey() throws Exception {
KeyPairGeneratorkpg = KeyPairGenerator.getInstance("DiffieHellman");

kpg.initialize(512);
KeyPairkp = kpg.generateKeyPair();
KeyFactorykfactory = KeyFactory.getInstance("DiffieHellman");
DHPublicKeySpeckspec = (DHPublicKeySpec) kfactory.getKeySpec(kp.getPublic(),
DHPublicKeySpec.class);
System.out.println("Public key is: " +kspec);
}
public static void createSpecificKey(BigInteger p, BigInteger g) throws Exception {
KeyPairGeneratorkpg = KeyPairGenerator.getInstance("DiffieHellman");
```

```
DHParameterSpecparam = new DHParameterSpec(p, g);
kpg.initialize(param);
KeyPairkp = kpg.generateKeyPair();
KeyFactorykfactory = KeyFactory.getInstance("DiffieHellman");
DHPublicKeySpeckspec = (DHPublicKeySpec) kfactory.getKeySpec(kp.getPublic(),
DHPublicKeySpec.class);
System.out.println("\nPublic key is : " +kspec);
}
}
```

OUTPUT:

Public key is: javax.crypto.spec.DHPublicKeySpec@5afd29
Public key is: javax.crypto.spec.DHPublicKeySpec@9971ad

8. SHA-1

AIM: Calculate the message digest of a text using the SHA-1 algorithm in JAVA.

PROGRAM:

```
import java.security.*;
public class SHA1 {
public static void main(String[] a) {
try {
MessageDigest md = MessageDigest.getInstance("SHA1");
System.out.println("Message digest object info: ");
System.out.println(" Algorithm = " +md.getAlgorithm());
System.out.println(" Provider = " +md.getProvider());
System.out.println(" ToString = " +md.toString());

String input = "";
md.update(input.getBytes());
byte[] output = md.digest();
System.out.println();
System.out.println("SHA1(\""+input+"\") = " +bytesToHex(output));
input = "abc";
md.update(input.getBytes());
output = md.digest();
System.out.println();
System.out.println("SHA1(\""+input+"\") = " +bytesToHex(output));
input = "abcdefghijklmnopqrstuvwxyz";
md.update(input.getBytes());
output = md.digest();
System.out.println();
System.out.println("SHA1(\"" +input+"\") = " +bytesToHex(output));
System.out.println(""); }
catch (Exception e) {
System.out.println("Exception: " +e);
}
}

public static String bytesToHex(byte[] b) {
```

```
char hexDigit[] = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F'};
StringBufferbuf = new StringBuffer();
for (int j=0; j<b.length; j++) {
buf.append(hexDigit[(b[j] >> 4) & 0x0f]);
buf.append(hexDigit[b[j] & 0x0f]); }
returnbuf.toString(); }
}
```

OUTPUT:

```
Message digest object info:
Algorithm = SHA1
Provider = SUN version 1.6
ToString = SHA1 Message Digest from SUN, <initialized>
SHA1("") = DA39A3EE5E6B4B0D3255BFEF95601890AFD80709
SHA1("abc") = A9993E364706816ABA3E25717850C26C9CD0D89D
SHA1("abcdefghijklmnopqrstuvwxyz")=32D10C7B8CF96570CA04CE37F2A19D8424
0D3A89
```

9. MD5

AIM: Calculate the message digest of a text using the SHA-1 algorithm in JAVA.

PROGRAM:

```
import java.security.*;
public class MD5 {
public static void main(String[] a) {
// TODO code application logic here
try {
MessageDigest md = MessageDigest.getInstance("MD5");
System.out.println("Message digest object info: ");
System.out.println(" Algorithm = " +md.getAlgorithm());
System.out.println(" Provider = " +md.getProvider());
System.out.println(" ToString = " +md.toString());
String input = "";
md.update(input.getBytes());
byte[] output = md.digest();

System.out.println();
System.out.println("MD5(\""+input+"\") = " +bytesToHex(output));
input = "abc";
md.update(input.getBytes());
output = md.digest();
System.out.println();
System.out.println("MD5(\""+input+"\") = " +bytesToHex(output));
input = "abcdefghijklmnopqrstuvwxyz";
md.update(input.getBytes());
output = md.digest();
System.out.println();
System.out.println("MD5(\"" +input+"\") = " +bytesToHex(output));

System.out.println("");
```

```
}
catch (Exception e) {
System.out.println("Exception: " +e); }
}
public static String bytesToHex(byte[] b) {
char hexDigit[] = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F'};
StringBufferbuf = new StringBuffer();
for (int j=0; j<b.length; j++) {
buf.append(hexDigit[(b[j] >> 4) & 0x0f]);
buf.append(hexDigit[b[j] & 0x0f]); }
return buf.toString(); } }
```

OUTPUT:

```
Message digest object info:
Algorithm = MD5
Provider = SUN version 1.6
ToString = MD5 Message Digest from SUN, <initialized>
MD5("") = D41D8CD98F00B204E9800998ECF8427E
MD5("abc") = 900150983CD24FB0D6963F7D28E17F72
MD5("abcdefghijklmnopqrstuvwxyz") = C3FCD3D76192E4007DFB496CCA67E13B
```

Text Books

1. “Cryptography and Network Security”, William Stallings, 2nd Edition, Pearson Education Asia
2. “Network Security private communication in a public world”, C. Kaufman, R. Perlman and M. Speciner, Pearson
3. Cryptography & Network Security: Atul Kahate, TMH.

Reference :

1. “Network Security Essentials: Applications and Standards” by William Stallings, Pearson
2. “Designing Network Security”, Merike Kaeo, 2nd Edition, Pearson Books
3. “Building Internet Firewalls”, Elizabeth D. Zwicky, Simon Cooper, D. Brent Chapman, 2nd Edition, Oreilly
4. “Practical Unix & Internet Security”, Simson Garfinkel, Gene Spafford, Alan Schwartz, 3rd Edition, Oreilly

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

Course Description

Title of Course: Grand Viva
Course Code: CS881
L-T –P Scheme: 0-0-0 **Course Credits: 4**

Aims and Objectives

1. To compare the traditional viva examination (TVE) with OSVE (Objective Structured Viva Examination).

2. To obtain the students' opinion regarding OSVE as an assessment tool.

3. A suggestion to include OSVE as a part of university examination.

Materials and Methods

The study was carried out in November 2012, at K.J. Somaiya Medical College, in the department of Anatomy. 50 students were exposed to different stations of viva as well as OSVE. A comparison was made of the student's performance and a feedback was taken from the students regarding the same.

As the OSVE was being conducted for the first time, the students were notified in advance regarding the plan for conducting the part ending practical assessment – by both the TVE and OSVE. The OSVE was planned for 20 marks, viva voce of 20 marks.

Purpose and Format of the Viva Voce Examination

Literally, "viva voce" means by or with the living voice - i.e., by word of mouth as opposed to writing. So the viva examination is where you will give a verbal defence of your thesis.

Put simply, you should think of it as a verbal counterpart to your written thesis. Your thesis demonstrates your skill at presenting your research in writing. In the viva examination, you will demonstrate your ability to participate in academic discussion with research colleagues.

Purpose of the Exam

The purpose of the viva examination is to:

- demonstrate that the thesis is your own work
- confirm that you understand what you have written and can defend it verbally
- investigate your awareness of where your original work sits in relation to the wider research field

- establish whether the thesis is of sufficiently high standard to merit the award of the degree for which it is submitted
- allow you to clarify and develop the written thesis in response to the examiners' questions

The Examiners and Exam Chair

You will normally have two examiners:

- an internal examiner who will be a member of academic staff of the University, usually from your School/Department but not one of your supervisors
- an external examiner who will normally be a member of academic staff of another institution or occasionally a professional in another field with expertise in your area of research (candidates who are also members of University staff will normally have two external examiners in place of an internal and an external examiner)

Your supervisor should let you know who your examiners will be as it is important that you ensure you are familiar with their work and any particular approach that they may take when examining your thesis.

In some cases there may also be a Chair person for the examination. A Chair is appointed if the Graduate Dean or either of the examiners feels this is appropriate, for example where the examining team has relatively little experience of examining UK research degrees. The Chair is there to ensure the examination is conducted in line with University regulations and is not there to examine your thesis. If there is a Chair person, it will usually be a senior member of the academic staff of your School/Department.

Normally no one else is present in the exam.

Exam Venue and Arrangements

Your internal examiner is responsible for arranging your viva exam and they will contact you with the relevant details - date, time, venue, etc.

Usually the viva exam will take place in your School/Department, though occasionally another University location may be used. If you are unsure where you need to go, make sure you check this before the day of your exam.

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

Course Description

If you returned your Notice of Intention to Submit Your Thesis three months before your submission date, your viva exam should normally take place quite soon after submission. Almost all viva exams take place within three months of thesis submission and in many cases it is within one month.

Format of the Exam

All viva examinations are different, so it is not possible to describe exactly what will happen - but there are general points which can be made which may be helpful, and you should have the opportunity before your examination to discuss what will happen with your supervisor or to attend the University's pre-viva examination workshop.

The purpose of the viva is to establish that your work is of a sufficiently high standard to merit the award of the degree for which it is submitted. In order to be awarded a research degree, the thesis should demonstrate an original contribution to knowledge and contain work which is deemed worthy of publication.

In order to do this, examiners may:

- ask you to justify your arguments
- ask you to justify not only things which you have included in your thesis but also things which you may have left out
- ask you questions about the wider research context in which the work has been undertaken
- argue certain points with you
- expect you to discuss any developments which may flow from your work in the future

Inevitably, your thesis will have strengths and weaknesses and the examiners will want to discuss these. It is considered a positive thing, indeed an essential thing, that you can discuss both the strengths and the weaknesses. You can think of the weaknesses as an opportunity to demonstrate your skill at critical appraisal.

Remember that examiners seek to find and discuss weaknesses in all theses - you should not interpret criticism as an indication that the examination will not end successfully.

UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

Course Description

Title of Course: Project Part-II
Course Code: CS882
L-T –P Scheme: 0-0-12 **Course Credits: 6**

Project: an activity where the participants have some degree of *choice* in the outcome. The result is complete and functional, that is, it has a beginning, middle and end. Usually, it spans multiple lab periods and requires work outside scheduled lab periods. Since there are choices in implementation, *design* is inherently a component of a project. A project is inherently different from an *analysis* or *exercise*, in which the solution has a predictable form. Projects span a wide variety of possibilities: design and build, identify a system, do a forensic analysis, evaluate a product or assess some environmental situation.

Program Objective 1
Graduates shall make their way to the society with proper scientific and technical knowledge in mechanical engineering.

Program Objective 2
Graduates shall work in design and analysis of mechanical systems with strong fundamentals and methods of synthesis.

Program Objective 3
Graduates shall adapt to the rapidly changing environment in the areas of mechanical engineering and scale new heights in their profession through lifelong learning.

Program Objective 4
Graduates shall excel in career by their ability to work and communicate effectively as a team member and/or leader to complete the task with minimal resources, meeting deadlines.

Program Outcomes:

1. Ability to apply knowledge of mathematics, science and mechanical engineering fundamentals for solving problems.
2. Ability to Identify, formulate and analyze mechanical engineering problems arriving at meaningful conclusions involving mathematical inferences.
3. Ability to design and develop mechanical components and processes to meet desired needs considering public health, safety, cultural, social, and environmental aspects.
4. Ability to understand and investigate complex mechanical engineering problems experimentally.
5. Ability to apply modern engineering tools, techniques and resources to solve complex mechanical engineering activities with an understanding of the limitations.
6. Ability to understand the effect of mechanical engineering solutions on legal, cultural, social, public health and safety aspects./li>

Course Description

7. Ability to develop sustainable solutions and understand their impact on society and environment.
8. Ability to apply ethical principles to engineering practices and professional responsibilities.
9. Ability to function effectively as an individual and as a member or leader in diverse teams and in multidisciplinary settings.
10. Ability to comprehend, design documentation, write effective reports, make effective presentations to the engineering community and society at large.
11. Ability to apply knowledge of engineering and management principles to lead teams and manage projects in multidisciplinary environments.
12. Ability to engage in independent and life-long learning in the broad context of technological changes and advancements.