Fourth Semester					
Theory					
Sl. No.	**Category**	**Subject Code**	**Subject Name**	**L-T-P**	**Credit**
1	HSMC	22CM4HS01T/ 22CM4HS02T	**Humanities-I** Organizational Behavior/ **Management-I** Engineering Economics and Costing	3-0-0	3
2	BSC	22CM4BS01T	Discrete Structure	3-1-0	4
3	PCC-3	22CT4PC01T	**PCC-3**: Computer Organization and Architecture	3-0-0	3
4	PCC-4	22CT4PC02T	**PCC-4**: Object-Oriented Programming using JAVA	3-0-0	3
5	PCC-5	22CT4PC03T	**PCC-5:** Design and Analysis of Algorithms	3-0-0	3
6	PEC-1	Professional Elective Course-1: 22CT4PE01T/ 22CT4PE02T/ 22CT4PE03T/ 22CT4PE04T/ 22CT4PE05T/ 22CT4PE07T/ 22CT4PE08T/	Computer Graphics/ Data Science for Engineers / Cryptography and Network Security/ Digital Signal Processing Mathematics for Data Science / Mobile Computing Advanced DBMS	3-0-0	3
7	**HSMC**	**22CM4HS03T**	**Universal Human Values-II**	**3-0-0**	**3**
Total Credit (Theory)					**22**
Practical					
2	PCC-3	22CT4PC01L	**PCC Lab-3**: Computer Organization and Architecture Laboratory	0-0-2	1
3	PCC-4	22CT4PC02L	**PCC Lab-4**: Object-Oriented Programming using JAVA Laboratory	0-0-2	1
4	PCC-5	22CT4PC03L	**PCC Lab-5:** Design and Analysis of Algorithms Laboratory	0-0-2	1
Total Credit (Practical)					**3**
Total Semester Credit					**25**

FOURTH SEMESTER DETAILED SYLLABUS

22CT4PC01T	Computer Organization and Architecture (3-0-0)	3 Credits

Course Objectives:

1. Able to understand the basic organizational structure of computer system along with the operational concepts, the concepts of ALU, CU and Memory design, the concept of cache memory, virtual memory and principle of pipelining.
2. Able to solve the problems related to cache memory and performance, page replacement algorithms, memory construction, arithmetic operations, and pipelining.
3. Able to analyze the performance differences of computing evolution on basic operation like addition, multiplication and division, page replacement algorithms and cache memory mappings.

Module-I: **[8 Hrs]**

Functional blocks of a computer: CPU, memory, input-output subsystems, Von-Neumanvs Harvard Architecture, Instruction set architecture of a CPU–registers, instruction execution cycle, Basic Operational Concepts, addressing modes, instruction set. Case study – instruction sets of some common CPUs.

Module-II: **[10 Hrs]**

Computer arithmetic – integer addition and subtraction, ripple carry adder, carry look-ahead adder, etc. multiplication – shift-and add, Booth multiplier, carry save multiplier, etc. Division restoring and non-restoring techniques, signed number representation, fixed and floating point representations, floating point arithmetic.
CPU control unit design: hardwired and micro-programmed design approaches, Case study – design of a simple hypothetical CPU.

Module-III: **[10 Hrs]**

Memory system design: semiconductor memory technologies, memory organization.
Memory organization: Memory interleaving, concept of hierarchical memory organization, cache memory, cache size vs. block size, mapping functions, replacement algorithms, write policies.

Module-IV: **[6 Hrs]**

Peripheral devices and their characteristics: Input-output subsystems, I/O device interface, I/O transfers–program controlled, interrupt driven and DMA, privileged and non-privileged instructions, software interrupts and exceptions. Programs and processes–role of interrupts in process state transitions, I/O device interfaces – SCII, USB

Module-V: **[6 Hrs]**

Pipelining: Basic concepts of pipelining, throughput, speedup and efficiency, pipeline hazards: Structural hazards, data hazards, control hazards.

Course Outcome:

1. Understand the theory and architecture of central processing unit.
2. Analyze some of the design issues in terms of speed, technology, cost, performance.
3. Design a simple CPU with applying the theory concepts.
4. Understand the architecture and functionality of central processing unit.
5. Exemplify in a better way the I/O and memory organization.
6. Define different number systems, binary addition and subtraction, 2's complement representation and operations with this representation.

Text Books:

1. “Computer Organization” 5th edition Carl Hamacher, Zvonkovranesic, SafwatZaky, McGraw Hill.
2. “Computer Organization and Design: The Hardware/Software Interface”, 5th Edition by David A. Patterson and John L. Hennessy, Elsevier.

22CT4PC02T	OBJECT-ORIENTED PROGRAMMING USING JAVA(3-0-0)	Credit: 3

Course Objective:

1. Learn the syntax, semantics and idioms of the Java programming language.
2. Gain confidence in object oriented programming principles through lots of practical exercises that provide useful exposure to the core Java class libraries.

Module- I : [8 Hrs]

Introduction to Java and Java programming Environment. Object Oriented Programming Concepts: Encapsulation, Abstraction ,Inheritance ,Polymorphism.

Fundamental Programming Structure: Data Types, variable, keywords, typecasting, Arrays, Operators and their precedence.

Control Flow: Java's Control Statements (if, switch, iteration, statement, while, do-while, for, Nested loop). Concept of Objects and Classes, Using Existing Classes building your own classes, constructor overloading, static , final, this keyword.

Module - II : [8 Hrs]

Inheritance: Introduction, types of inheritance. Use of super keyword. Method overriding, Dynamic method Dispatch, Using Abstract Classes, Using final with inheritance. The Object Class.

Packages & Interfaces: Packages, Access Protection, Importing package, Interface, Implementing Interfaces, variables in Interfaces, Interfaces can be extended.

Exception Handling: Fundamentals, Types Checked , Unchecked exceptions, Using try & catch, Multiple catch, throw , throws, finally, Java's Built in exceptions, user defined exception.

Module –III : [8 Hrs]

Multi Threading: Java Thread Life Cycle, Thread Priorities, Synchronization, Creating a thread, Runnable interface, Creating Multiple threads, Using isAlive () and join (), wait () &notiCS().

String Handling: String constructors, String length, Character Extraction, String Comparison, Modifying a string.

Java I/O: Classes & Interfaces, Stream classes, Byte streams, Character streams, Serialization.

Module IV : [6 Hrs]

Wrapper Classes : Wrapper classes and its methods.
Collection Framework: Introduction, interfaces, List, Set, Map etc, List interfaces and its classes. **Introduction to Database**: Introduction to DataBase. Driver Types, Registering Driver, Creating Connection, Executing SQL query using Statement, PreparedStatment.ResultSet methods.

Module-V: [6 Hrs]

Event Handing: Event Delegation Model, Event Classes, Event Listener Interfaces, Adapter classes.

AWT: AWT Classes window fundamentals, component, container, panel, Window, Frame, working with Graphics , Control Fundamentals , Layout managers, Handling Events by Extending AWT components.
Swing: Icons & Labels, Text fields, Buttons, Combo boxes, Tabbed panes, Scroll panes, Trees, Tables.

Course Outcome:

1. Implement and apply various Object Oriented programming concepts.
2. Applying Collection Classes and Files, Multiple Threads & handle Exceptions in developing a java applications.
3. Developing a Java standalone application having front end design and back end.

Text Books:

1. Java: One Step Ahead by Anita Seth (Author), B.L. Juneja (Author) Oxford University Press.
2. Head First Java 2nd edition Kathy Sierra & Bert Bates

Reference Books:

3. JAVA Complete Reference (9th Edition) Herbert Schildt.
4. https://www.udemy.com/java-the-complete-java-developer-course/
   Java Programming Masterclass for Software Developers Created by Tim Buchalka, Tim Buchalka's Learn Programming Academy, GoranLochert

22CT4PC03T	**Design and Analysis of Algorithm(3-1-0)**	**4 Credit**

Course Objective:

1. Translating a plain text problems to convert into an algorithm
2. Calculate best case, worst case time complexity and space complexities of different algorithm and choosing the best solution from the available options
3. Applying different design paradigm to solve different problems and comparing their best case, worst case scenarios.
4. Designing and applying different data structures over different algorithms for solving different problems.
5. Understand different P-class, NP class problems.

Module-I: **[12 Hrs]**

Introduction to problems and algorithms , Mathematics for algorithm analysis , Insertion sort Analysing algorithms, Designing of algorithms, Asymptotic notation Standard notations and common functions, Recurrence relations, The substitution method, The recursion-tree method, The master method, Divide and conquer: Min-Max Heap, Priority queue,Heapsort , Quicksort, Merge Sort, Sorting in Linear Time: Lower bounds for sorting: Counting sort, Radix sort, Bucket sort, Fast Fourier transform , Finding the convex hull : Graham Scan, Finding the closest pair of points

Module-II: **[8 Hrs]**

Greedy method: Elements of the greedy strategy, Huffman codes, task-scheduling problem, Fractional Knapsack problem, Coin change problem, Dynamic programming: Assembly-line Scheduling, Matrix-Chain Multiplication, Longest Common Subsequence(LCS), 0/1 Knapsack problem, Rod Cutting problem

Module-III: **[6 Hrs]**

Graph algorithms: Basic Definitions and Application, Representations of graphs, Breadth-first search and Depth-first search, Data Structures for Disjoint Sets, Strongly connected components, Minimum Spanning Trees: The algorithms of Kruskal and Prim

Module-IV: **[6 Hrs]**

Single-Source Shortest Paths: The Bellman-Ford algorithm, Dijkstra's algorithm, All-Pairs Shortest Paths- Shortest paths and matrix multiplication, The Floyd-Warshall algorithm

String Matching: The naive string-matching algorithm, The Rabin-Karp algorithm, The Knuth- Morris-Pratt algorithm.

Module-V: **[8 Hrs]**

Network Flow: Flow networks, The Ford-Fulkerson method, Maximum bipartite matching. Backtracking – n-Queen problem – Hamiltonian Circuit Problem – Subset Sum Problem.

Branch and Bound – LIFO Search and FIFO search – Assignment problem – Knapsack Problem, NP-Completeness: Classes P and NP, NP-complete problems.: Reduction of 3SAT to Subset Sum, Approximation Algorithm for TSP

Course Outcome:

1. Given a English language problem description define the problem precisely with input/output requirements, examine its inherent complexity and develop a generic or set of initial solutions and justify their correctness.
2. Given an algorithm descriptions, analyse the time and space complexity of the algorithm in the worst case, average case, and amortized scenario as needed in terms of asymptotic order of complexity.
3. Given a problem definition explore different alternative algorithmic solutions, compare them with respect to time and space complexity and choose the design scheme and /or design parameter and data structure appropriately to obtain the best possible choice(s) that can be converted to an executable programs.
4. Examine and prove whether a problem is of polynomial complexity, hard(np complete) or otherwise and develop optimal and approximate algorithm for them as applicable.

Text Books:

1. Thomas H Cormen, Charles E Lieserson, Ronald L Rivest and Clifford Stein, Introduction to Algorithms, Third Edition, MIT Press/McGraw-Hill, 2009.
2. Ellis Horowitz, SartajSahni and SanguthevarRajasekaran, Computer Algorithms/ C++, Second Edition, Universities Press, 2007.

Reference Books:

3. SanjoyDasgupta, Christos H. Papadimitriou and Umesh V. Vazirani, Algorithms, McGraw- Hill, 2008.
4. Jon Kleinberg and ÉvaTardos, Algorithm Design, Addison-Wesley/PEARSON EDUCATION- 2006.
5. S. Sridhar, ―Design and Analysis of Algorithms‖, Oxford university press, First Edition, 2015.

MOOC:

1. Prof.Abhiram G Ranade, Prof.Ajit A Diwan, Prof.SundarViswanathan,IIT Bombay, https://nptel.ac.in/courses/106101060/
2. Prof.MadhavanMukund,Chennai Mathematical Institute, https://nptel.ac.in/ courses/106106131/
3. Reyna Hulett, CS161, Stanford School of Engineering, https://online.stanford.edu/courses/cs161-design-and-analysis-algorithms

22CT4PE01T	**Computer Graphics (3-0-0)**	**3 Credit**

Course Objective:

1. To explain basic principles for representation of the geometric objects in the 2D and 3D coordinates. Utilize the computer system and methods
2. To demonstrate the implementation of the algorithms and techniques necessary to produce geometric objects in 2D and 3D space illustrations.
3. To elaborate the clipping and projection technique and curve tracing methods.
4. To elaborate the geometric optics necessary to determine how light bounces off surfaces. Shading algorithms to determine how a surface should be shaded to produce realistic illustrations. Curves and surfaces methods for rendering and shading curved objects.

Module-I: **[8 Hrs]**

Overview of Graphics System: Video Display Units, Raster-Scan and Random Scan Systems, Graphics Input and Output Devices. Output Primitives: Line drawing Algorithms: DDA and Bresenham's Line Algorithm, Circle drawing Algorithms: Midpoint Circle Algorithm and Bresenham's Circle drawing Algorithm.

Module-II: **[6 Hrs]**

Two Dimensional Geometric Transformation: Basic Transformation (Translation, Rotation, Scaling) Matrix Representation, Composite transformations, Reflection, Shear, Transformation between coordinate systems. Two Dimensional Viewing: Window-to- View Port Coordinate Transformation.

Module-III: **[10 Hrs]**

Clipping: Line Clipping (Cohen-Sutherland Algorithm) and Polygon Clipping (Sutherland- Hodgeman Algorithm), Aliasing and Antialiasing, Half Toning, Thresholding, Dithering. Polygon Filling: Seed Fill Algorithm, Scan line Algorithm. Two Dimensional Object Representations: Spline Representation, Bezier Curves, B-Spline Curves. Fractal Geometry: Fractal Classification and Fractal Dimension.

Module-IV: **[8 Hrs]**

3D Geometric and Modelling Transformations: Translation, Rotation, Scaling, Reflections, shear, Composite Transformation. Projections: Parallel Projection, Perspective Projection.

Visible Surface Detection Methods: Back-Face Detection, Depth Buffer, A- Buffer, Scan- Line Algorithm, Painters Algorithm.

Module-V: **[8 Hrs]**

Illumination Models: Basic Models, Displaying Light Intensities. Color models: properties of light, XYZ, RGB, YIQ and CMY color models, Surface Rendering Methods: Polygon Rendering Methods: Gouraud Shading, Phong Shading.

Computer Animation: Types of Animation, Key frame Vs. Procedural Animation, Methods of Controlling Animation, Morphing. Introduction to Virtual Reality and Augmented Reality.

Course Outcome:

1. Student will illustrate the basic principles for representation of the geometric objects in the 2D and 3D coordinates with clipping, projection and shading.
2. Student will implement the algorithms for producing geometric objects in 2D and 3D space using C language.

3. Develop a standalone graphics project using visual animation and rendering

Text Books:

1. Computer Graphics, C version; D. Hearn and M. P. Baker; Pearson Education, 2^{nd} Edition, 2002
2. Computer Graphics Principle and Practice, J.D. Foley, A. Dam, S.K. Feiner, Addison Wesley, 4^{th} Edition, 2014.

Reference Books:

3. Procedural Elements of Computer Graphics, David Rogers, TMH. 1998
4. https://www.coursera.org/learn/interactive-computer-graphics by Takeo Igarashi (Professor) Department of Computer Science, Graduate School of Information Science and Technology, University of Tokyo.

22CT4PE02T	**Data Science for Engineers (3-0-0)**	**3 Credit**

Course Objectives :

1. Introduce R as a programming language
2. Introduce the mathematical foundations required for datascience
3. Introduce the first level data science algorithms
4. Introduce a data analytics problem solving framework
5. Introduce a practical capstone case study

MODULE -I **[8 Hrs]**

Introduction: Introduction to Data Science, Data Science Venn Diagram, Relation to data mining, machine learning, big data and statistics, Business Intelligence (BI) vs. Data Science. Types of Data: Structured v/s unstructured data, Examples of data pre-processing, Quantitative vs qualitative data, Four levels of data. Stages of a data science project: Defining the goal, Data collection and management, Explore the data, Modeling, Model evaluation and critique, Presentation and documentation.

MODULE -II **[8 Hrs]**

Introduction to Linear algebra for data science: Vectors and matrices.
Introduction to Probability: Bayesian versus Frequentist, Frequentist approach, The law of large numbers, Compound events, Conditional probability, Bayesian ideas revisited, Bayes theorem , More applications of Bayes theorem, Random variables, Discrete random variables.
Basic Statistics: Obtaining data (Observational, Experimental), Sampling data, Probability sampling, Random sampling, Unequal probability sampling, measurement of statistics , Measures of center (Mean, Median, Mode, Skewness, Quantile, Percentile), Measures of variation, Measures of relative standing, Correlations in data, The Empirical rule.

MODULE -III **[12 Hrs]**

Data Visualization: Basic principles, ideas and tools for data visualization, Identify effective and ineffective visualization (Scatter plots, Line graphs, Bar charts, Histograms, Box plots), Correlation versus causation, Simpson's paradox, Verbal communication.
Machine Learning Essentials: Machine learning, Working principles, Types of machine learning (Supervised learning, Unsupervised learning, Reinforcement learning), How does statistical modeling fit. Some Basic Algorithms like Linear Regression, k-Nearest Neighbors (k-NN), k-Means, Decision Tree. Feature Extraction, Eigen vectors and Eigen values, Principal Component Analysis (PCA).

MODULE -IV **[6 Hrs]**

Beyond the Essentials: The bias variance tradeoff (Error due to bias, Error due to variance, Two extreme cases of bias/variance tradeoff, How bias/variance play into error functions), K folds cross-validation, Grid searching (Visualizing training error versus cross-validation error), Ensembling techniques (Random forests, Comparing Random forests with decision trees), Introduction to structure of Neural networks.

MODULE -V **[6 Hrs]**

Hands on laboratory using R Language for example like Data Visualization (Scatter plots, Line graphs, Bar charts, Histograms, Box plots), Some Basic Algorithms like Linear Regression, k- Nearest Neighbors (k-NN), k-Means, Decision Tree. Principal Component Analysis (PCA), Random Forests, Neural Networks.

Course Outcomes:

1. Describe a flow process for data science problems (Remembering)
2. Classify data science problems into standard typology (Comprehension)
3. Develop R codes for data science solutions (Application)
4. Correlate results to the solution approach followed (Analysis)
5. Assess the solution approach (Evaluation)
6. Construct use cases to validate approach and identify modifications required

Text Books:

1. Principles of Data Science, SinanOzdemir, Packt Publishing Ltd 2016.
2. Doing Data Science, Straight Talk From The Frontline, Cathy O'Neil and Rachel Schutt., O'Reilly. 2014.
3. An Introduction to Statistical Learning with Applications in R. James G, Witten D, Hastie Tibshirani R, Springer, 2013.
4. Hands-On Data Science with R: Techniques to perform data manipulation and ...,Vitor Bianchi Lanzetta, NatarajDasgupta, Ricardo AnjoletoFarias, Packt publishing ltd, 2018.
5. Data Science for Engineers :https://swayam.gov.in/nd1_noc19_cs60/preview (Prof. RaghunathanRengasamy& Prof. Shankar Narasimhan, IIT Madras).
6. https://www.udemy.com/course/data-science-and-machine-learning-bootcamp-with-r/ (Created by Jose Portilla)
7. https://www.udemy.com/machinelearning/ Machine Learning A-Z™: Hands-On Python & R In Data Science By: Kirill Eremenko, Hadelin de Ponteves

22CT4PE03T	**Cryptography and Network Security(3-0-0)**	**3 Credit**

Course Objective:

1. Understand OSI security architecture and classical encryption techniques.
2. Acquire fundamental knowledge on the concepts of finite fields and number theory.
3. Understand various block cipher and stream cipher models.
4. Describe the principles of public key cryptosystems, hash functions and digital signature.

Module I: **[8 Hrs]**

Introduction & Number Theory: Services, Mechanisms and attacks, the OSI security architecture, Network security model

Classical Encryption techniques: Symmetric cipher model, substitution techniques, transposition techniques, steganography.

FINITE FIELDS AND NUMBER THEORY: Groups, Rings, Fields, Modular arithmetic, Euclidean's algorithm Finite fields, Polynomial Arithmetic, Prime numbers: Fermat's and Euler's theorem, testing for primality, The Chinese remainder theorem, discrete logarithms.

Module II: **[8 Hrs]**

Block Ciphers and Public Key Cryptography: Block cipher principles, Data Encryption Standard, strength of DES, Block cipher design principles, block cipher modes of operation, Advanced Encryption Standard (AES), Triple DES

Principles of public key cryptosystems-The RSA algorithm-Key management – Diffie Hellman Key exchange-Elliptic curve arithmetic-Elliptic curve cryptography.

Module III: **[8 Hrs]**

Hash Functions and Digital Signatures: Authentication requirement, Authentication function, MAC, Hash function, Security of hash function and MAC, MD5, SHA, HMAC , CMAC, Digital signature and authentication protocols

Module IV: **[8 Hrs]**

System Security: Authentication applications, Kerberos, X.509 Authentication services, Intruders, Intrusion detection, password management, viruses and related threats, Firewall, Types of Firewalls, Internet Firewalls for Trusted system,

Module V: **[8 Hrs]**

Network Security:

E-mail Security: Security Services for E-mail-attacks possible through E-mail: Pretty Good Privacy, S/MIME. IPSecurity: Overview of IPSec, IP and IPv6-Authentication Header-Encapsulation, Web Security: SSL/TLS Basic Protocol, computing the keys, client authentication, Encoding-Secure Electronic Transaction (SET).

Course Outcome:

1. Explain how different Hash Functions and Digital Signatures algorithms and Public Key Cryptography techniques applied for developing cryptographic and digital signature systems.
2. Designing and solving block cyphering and public key cryptography algorithms
3. Explain and compare different Network security system and system security.

Text Books:

1. William Stallings, Cryptography and Network Security, 6th Edition, Pearson Education,2013.
2. Charlie Kaufman, Radia Perlman and Mike Speciner, “Network Security: Private Communication in a Public World”, 2nd Edition, Prentice-Hall, 2002.

Reference Books:

22CT4PE04T	**Digital Signal Processing (3-0-0)**	**3 Credit**

Course Objective:

1. Represent signals mathematically in discrete-time.
2. Analyze discrete-time systems using z-transform.
3. Understand the Discrete-Fourier Transform (DFT) and the FFT algorithms.
4. Design digital filters for various applications.

Module I: **[8 Hrs]**

Discrete-time signals and systems: Discrete-time signals, Discrete-time systems, LTI systems, Discrete-time systems described by difference equations;Implementation of Discrete-time systems,Correlation of Discrete-time signals.

Module II: **[8 Hrs]**

z-transform: z-transform, Region of Convergence, Analysis of Linear Time Invariant systems using z-transform,Properties of z-transform for causal signals, Interpretation of stability in z-domain, Inverse z-transforms, the one-sided z-transform.

Module III: **[8 Hrs]**

Discrete Fourier Transform : Frequency Domain Analysis, Discrete Fourier Transform (DFT), Properties of DFT, Convolutions ofsignals, Fast Fourier Transform Algorithm, Linear filtering methods based on the DFT, Implementation of Discrete Time Systems.

Module IV: **[8 Hrs]**

Designof Digital filters: Designof FIR Digital filters: Windowmethod,Park-McClellan's method. Effect of finite register length in FIR filter design.

Design of IIR Digital Filters:Butterworth, Chebyshev and Elliptic Approximations,Low-pass, Band- pass, Band-stop and High-passfilters.

Module V: **[8 Hrs]**

Designof Adaptive Digital filters : Adaptive Direct-form FIR filter, LMS and RLS algorithms, Adaptive lattice-ladder filter, Kalman Filter

Course Outcome:

1. Represent signals and systems mathematically in discrete-time
2. Analyzeand apply discrete-time systems using different transform functions.
3. Design digital filters for various applications.

Text Books:

1. J. G. Proakis and D.G. Manolakis, “Digital Signal Processing: Principles, Algorithms and Applications”,4th Edition, Pearson, 2007.
2. S. K. Mitra, “Digital Signal Processing: A computer based approach”, 4th Edition, McGraw Hill, 2013.

Reference Books:

3. A.V. Oppenheim and R. W. Schafer, “Discrete Time Signal Processing”, 3rd Edition, Pearson/Prentice-Hall, 2010.
4. L. R. Rabiner and B. Gold, “Theory and Application of Digital Signal Processing”, Prentice Hall, 1992.
5. NPTEL Course by Prof. S.C. Dutta Roy, IIT Delhi https://nptel.ac.in/courses/117102060/

22CT4PC01L	Computer Organization and Architecture Laboratory **(0-0-2)**	**1 Credits**

Course Objective:

1. Understanding the behavior of Logic Gates, Adders, Decoders, Multiplexers and Flip-Flops.
2. Understanding the behavior of ALU, RAM, STACK and PROCESSOR from working modules and the modules designed by the student as part of the experiment

Laboratory Experiments

1. Study of Computer Components

(a) Identification of different components of a PC.

(b) Assembling & disassembling of a PC.

2. Study of different troubleshooting of a dot matrix printer using LX 1050+ Printer Trainer Module.

3. Study of the functions of SMPS using SMPS Trainer Kit.

(a) Study of SMPS with Single Output under Line Regulation.

(b) Study of SMPS with Multi Output under Line Regulation.

(c) Study of SMPS with Single Output under Load Regulation.

4. Study of different troubleshooting of CPU using CPU Trainer Module.

5. Familiarization of different types of byte addressing instruction using 8085 simulator.

6. Study of assembly Language program in PC using 8086 architecture.

7. Design of digital circuits (H/A, F/A, Decoder & Encoder) in VHDL using Active VHDL.

8. Design of digital circuits (MUX, DEMUX & ALU) in VHDL using Active VHDL.

9. Write a C/C++ program to perform signed bit multiplication using Booth's algorithm.

10. Write a C/C++ program for IEEE-754 floating point representation and perform

Addition/Subtraction.

Course Outcome:

1. Analyze the behavior of logic gates
2. Design combinational circuits for basic components of computer system and applications.
3. Analyze the operational behavior and applications of various flip-flop
4. Design Arithmetic logic units and different types of memory blocks.

22CT4PC02L	Object-Oriented Programming using Java Laboratory (0-0-2)	1 Credit

Course Objective:

1. Learn and implement Programs with the syntax, semantics and idioms of the Java programming language.
2. Implement practical exercises
3. Develop a standalone application.

List of Experiments:

1. Data types& variables, decision control structures: if, nested if etc Loop control structures: do, while, for etc.
2. Classes and objects.
3. Data Abstraction & Data hiding, Inheritance.
4. Interfaces and inner classes, wrapper classes.
5. Exception handlings
6. Threads
7. IO Files
8. Collections
9. Database Connectivity.
10. Applets AWT and Swing.

Course Outcome:

1. Understand and implement various Object Oriented Concepts like inheritance, abstraction and polymorphism.
2. Work with Collection Classes and Files, Multiple Threads, & handle Exceptions.
3. Develop applications to interact with a Database.
4. Design and implement Graphical User Interface(GUI) Applications in Java using AWT and Swing.

Text Books and Reading Materials :

1. Java: One Step Ahead by Anita Seth (Author), B.L. Juneja (Author) Oxford University Press.
2. Head First Java 2nd edition Kathy Sierra & Bert Bates
3. JAVA Complete Reference (9th Edition) Herbert Schildt.
4. https://www.udemy.com/java-the-complete-java-developer-course/
5. Java Programming Masterclass for Software Developers Created by Tim Buchalka, Tim Buchalka's Learn Programming Academy, GoranLochert

22CT4PC03L	Design and Analysis of Algorithms Laboratory **(0-0-2)**	**1 Credit**

List of Experiments:

1. Insertion Sort/ Selection Sort
2. Divide and Conquer: Fibonacci search/Binary search
3. Divide and Conquer: Merge Sort/Quicksort/Heap Sort
4. Divide and Conquer: Convex hull/Finding closet pair
5. Dynamic Programming: MCM/LCS
6. Dynamic Programming: Rod Cutting problem /Assembly line Scheduling
7. Greedy method: Activity Selection/Huffman Coding
8. Graph Search: BFS/DFS
9. Graph Greedy MST: Kruskal/Prim's
10. Graph Greedy Shortest Path: Bellman ford/Dijkstra
11. Rabin Karp string matching algorithm/Subset Sum problem using Branch and Bound

Prerequisite: Each student should have a good knowledge on basic data structures like Stack, Queue, List, Heap, Matrix.