FACULTY OF ENGINEERING

Scheme of Instruction & Examination

(AICTE Model Curriculum)

and

Syllabi

B.E. V and VI Semesters

of

Four Year Degree Program

in

B.E (COMPUTER SCIENCE and ENGINEERING)

(w.e.f: 2022-23)

Issued by

Dean, Faculty of Engineering

Osmania University, Hyderabad – 500 007

2022

Chairperson, BoS

Dean, FoE OU

SCHEME OF INSTRUCTION & EXAMINATION
B.E (Computer Science and Engineering)
SEMESTER-V

S. No.	Course Code	Course Title	Scheme of Instruction				Scheme of Examination			Credits
			L	T	D/P	Contact Hrs/Wk	CIE	SEE	Duration in Hrs	
Theory Courses										
1	PC 501 CS	Software Engineering	3	1	-	4	30	70	3	3
2	PC 502 CS	principles of Programming Languages	3	1	-	4	30	70	3	3
3	PC 503 CS	Automata Languages & Computation	3	1	-	4	30	70	3	3
4	PC 504 CS	Artificial intelligence	3	-	-	3	30	70	3	3
5	PC 505 CS	Computer Networks	3	-	-	3	30	70	3	3
6	PE-51X CS	Professional Elective-I	3	-	-	3	30	70	3	3
Practical/Laboratory Courses										
7	PC 551 CS	Software Engineering Lab	-	-	2	2	25	50	3	1
8	PC 552 CS	Artificial intelligence lab	-	-	2	2	25	50	3	1
9	PC 553 CS	Computer Networks LAB	-	-	2	2	25	50	3	1
Total			**18**	**03**	**06**	**27**	**255**	**570**		**21**

Profession Elective – I	
Course Code	**Course Title**
PE 511 CS	Parallel and Distributed Algorithms
PE 512 CS	Embedded Systems
PE 513 CS	Computer Graphics
PE 514 CS	Object oriented Analysis and Design
PE 515 CS	Data Science
PE 516 CS	Blockchain Technology

SOFTWARE ENGINEERING

PC 501 CS

Instruction: 3+1 periodsperweek Duration of SEE: 3 hours

CIE: 30marks SEE: 70marks

Credits: 3

Objectives:

1. To introduce the basic concepts of software development processes from defining a product to shipping and maintaining.
2. To impart knowledge on various phases, methodologies, and practices of software development.
3. To understand the importance of testing in software development, study various testing strategies along with its relationship with software quality and metrics.

Outcomes:

Student will be able to

1. Acquired working knowledge of alternative approaches and techniques for each phase of software development.
2. Decompose the given project in various phases of a lifecycle.
3. Judge an appropriate process model(s) assessing software project attributes and analyze necessary requirements for project development eventually composing SRS
4. Acquire skills necessary as an independent or as part of a team for architecting a complete software project by identifying solutions for recurring problems exerting knowledge on patterns
5. Concede product quality through testing techniques employing appropriate metrics by understanding the practical challenges associated with the development of a significant software system

UNIT – I

Introduction to Software Engineering: A generic view of Process: Software Engineering, Process Framework, CMM Process Patterns, ProcessAssessment.

Process Models: Prescriptive Models, Waterfall Model, Incremental Process Models, Evolutionary Process Models, Specialized Process Models, The Unified Models, Personal and Team Process Models, Process Technology, Product andProcess.

An Agile view of Process: Introduction to Agility and Agile Process, Agile Process Models.

UNIT – II

Software Engineering Principles: SE Principles, Communication Principles, Planning Principles, Modeling Principles, Construction Principles, Deployment.

System Engineering: Computer-based Systems, The System Engineering Hierarchy, Business Process Engineering, Product Engineering, System Modeling.

Requirements Engineering: A Bridge to Design and Construction, RequirementsEngineering Tasks, Initiating Requirements Engineering Process, Eliciting Requirements, Developing Use- Cases, Building the Analysis Model, Negotiating Requirements, ValidatingRequirements.

UNIT – III

Building the Analysis Model: Requirements Analysis Modeling Approaches, Data Modeling Concepts, Object-Oriented Analysis, Scenario-based Modeling, Flow-oriented Modeling, Class-based Modeling, Creating a Behavioral Model.

Design Engineering: Design within the context of SE, Design Process and Design Quality, Design Concepts, The Design Model, Pattern-based Software Design.

UNIT – IV

Creating an Architectural Design: Software Architecture, Data Design, Architectural Styles and Patterns, Architectural Design.

Modeling Component-Level Design: Definition of Component, Designing Class-based Components, Conducting Component-level Design, Object Constraint Language, Designing Conventional Components.

Performing User Interface Design: The Golden Rules, User Interface Analysis and Design, Interface Analysis, Interface Design Steps, Design Evaluation.

UNIT – V

Testing: Strategies: A Strategic Approach to Conventional Software Testing, Test Strategies for O-OSoftware.**Tactics:**SoftwareTestingFundamentals,Black-boxandWhite-boxTesting, Basis Path Testing, Control Structure Testing, O-O TestingMethods.

Debugging: Debugging Techniques, The Art of Debugging.

Product Metrics: A Framework for Product Metrics, Metrics for each phase of software development.

Software Quality: Definition, **Quality Assurance:** Basic Elements, Formal Approaches, Statistical Software Quality Assurance, Software Reliability, ISO9000 Quality Standards,SQA Plan.

Suggested Readings:

1. Roger S. Pressman, *Software Engineering: A Practitioner's Approach*, 7th Edition, McGraw Hill, 2009
2. Ali Behforooz and Frederick J. Hudson, *Software Engineering Fundamentals*, Oxford University Press, 1996
3. PankajJalote, *An Integrated Approach to Software Engineering*, 3rd Edition, Narosa Publishing House, 2008

PRINCIPLES OF PROGRAMMING LANGUAGES

PC 502 CS

Instruction: 3+1 periods per week — Duration of SEE: 3 hours

CIE: 30marks — SEE: 70marks

Credits: 3

Objectives:

1. To briefly describe various programming paradigms.
2. To provide conceptual understanding of High level language design and implementation.
3. To introduce the power of scripting languages.
4. To provide an introduction to formalisms for specifying syntax and semantics of programming languages.
5. To provide an exposure to core concepts and principles in contemporary programming languages.
6. To analyze and optimize the complexity of the programming languages.

Outcomes:

Students will be able to :

1. Ability to express syntax and semantics in formal notation.
2. Ability to apply suitable programming paradigm for the application
3. Gain Knowledge and comparison of the features programming languages.
4. Identify and describe semantic issues associated with variable binding, scoping rules, parameter passing, and exception handling.
5. Understand the design issues of object-oriented and functional languages.

UNIT- I

Preliminary Concepts: Reasons for studying, concepts of programming languages, Programming domains, Language Evaluation Criteria, influences on Language design, Language categories, Programming Paradigms – Imperative, Object Oriented, functional Programming, Logic Programming. Programming Language Implementation – Compilation and Virtual Machines, programming environments. Syntax and Semantics: general Problem of describing Syntax and Semantics, formal methods of describing syntax - BNF, EBNF for common programming languages features, parse trees, ambiguous grammars, attribute grammars, denotation semantics and axiomatic semantics for common programming language features.

UNIT- II

Data types: Introduction, primitive, character, user defined, array, associative, record, union, pointer and reference types, design and implementation uses related to these types. Names, Variable, concept of binding, type checking, strong typing, type compatibility, named constants, variable initialization. Expressions and Statements: Arithmetic relational and Boolean expressions, Short circuit evaluation mixed mode assignment, Assignment Statements, Control Structures – Statement Level, Compound Statements, Selection, Iteration, Unconditional Statements, guardedcommands.

UNIT-III

Subprograms Block and Fundamentals of sub-programs: Scope and lifetime of variable, static and dynamic scope, Design issues of subprogram and operations, local referencing environments, parameter passing methods, overloaded sub-programs, generic sub-programs, parameters that are subprogram names, design issues for functions user defined overloaded operators, co-routines.

UNIT- IV

Abstract types: Data Abstractions and encapsulation, introductions to data abstraction, design issues, language examples, C++ parameterized ADT, object oriented programming in small talk, C++, Java, C#, Ada 95 Concurrency: Subprogram level concurrency, semaphores, monitors, massage passing, Java threads, C# threads. Exception handling: Exceptions, exception Propagation, Exception handler in Ada, C++ and Java. Logic Programming Language: Introduction and overview of logic programming, basic elements of prolog, application of logic programming.

UNIT- V

Functional Programming Languages: Introduction, fundamentals of FPL, LISP, ML, Haskell, application of Functional Programming Languages and comparison of functional and imperative Languages. Scripting Language: Pragmatics, Key Concepts, Case Study: Python – Values and Types, Variables, Storage and Control, Bindings and Scope, Procedural Abstraction, Data Abstraction, Separate Compilation, Module Library.

Suggested Readings:

1. Concepts of Programming Languages Robert W. Sebesta 8/e, PearsonEducation,2008.
2. Programming Language Design Concepts, D. A. Watt, Wiley dreamtech,rp-2007
3. Programming Languages, 2nd Edition, A.B. Tucker, R.E. Noonan,TMH.
4. Programming Languages, K. C. Louden, 2nd Edition, Thomson,2003.
5. LISP, Patric Henry Winston and Paul Horn, PearsonEducation.
6. Programming in Prolog, W.F. Clocksin, & C.S. Mellish, 5th Edition,Springer.
7. Programming Python, M. Lutz, 3rd Edition, O'reilly, SPD,rp-2007.
8. Core Python Programming, Chun, II Edition, Pearson Education,2007.
9. Guide to Programming with Python, Michael Dawson, Thomson,2008.

AUTOMATA LANGUAGES & COMPUTATION

PC 503 CS

Instruction: 3+1 periods per week Duration of SEE: 3 hours

CIE: 30marks SEE: 70marks

Credits: 3

Objectives:

1. Develop a formal notation for strings, languages, andmachines.
2. Design finite automata to accept a set of strings of alanguage.
3. Prove that a given language is regular and apply the closure properties of languages.Design context free grammars to generate strings from a context free language and Convert them into normalforms.
4. Prove equivalence of languages accepted by Push down Automata and languages generated by context freegrammars.
5. Identify the hierarchy of formal languages, grammars, and machines.
6. Distinguish between computability and non-computability and Decidability and undecidability

Outcomes:

Students will be able to:

1. Design finite automata to accept a set of strings of a language.
2. For a given language determine whether the given language is regular or not.
3. Design context free grammars to generate strings of context free languages.
4. Determine equivalence of languages accepted by Pushdown Automata and languages generated by context free grammars.
5. Distinguish between computability and non-computability and Decidability and undecidability.

UNIT – I

Introduction: Finite state automata, Non-deterministic finite state automata, FA with €- transitions, Regular expressions, Applications of FA, Properties of regular sets, Pumping Lemma, Closure properties, Myhill-Nerode Theorem, Minimization of FA.

UNIT – II

Context Free Grammars and Languages: Derivations, Parse-trees, Ambiguity in Grammars and Languages. Pushdown Automata–Definitions, The languages of PDA, Equivalence of PDAs and CFGs, Deterministic Pushdown Automata.

UNIT – III

Properties of CFLs: Normal forms for CFGs, Pumping Lemma, Closure properties, Deterministic Context Free Languages,Decision properties.

UNIT – IV
Turing Machines: Introduction, Computational Languages and Functions, Techniques for construction of Turing machines. Modifications of TM, TM as enumerator, Restricted TM.

UNIT – V
Undecidability: Recursive and Recursively enumerable languages, UTM and undecidable problem, Rice Theorem, Post's correspondence problem. Chomsky's Hierarchy–Regular grammars, Unrestricted grammar, CSL, Relationship between classes of languages.

Suggested Readings:

1. John E. Hopcroft, Rajeev Motwani and Jeffrey D. Ullman, *Introduction to Automata Theory, Languages, and Computation,* 3rd Edition, Pearson EducationAsia,2007.

2. JohnMartin,*IntroductiontoLanguagesandTheTheoryofComputation*,3rd Edition,Tata McGrawHill,2013.

ARTIFICIAL INTELLIGENCE

PC 504 CS

Instruction: 3 periods per week Duration of SEE: 3 hours

CIE: 30marks SEE: 70marks

Credits: 3

Objectives:

1. Understand the importance of the field of AI by discussing its history and various Applications
2. Learn about one of the basic applications of A.I, search state formulations.
3. Learn methods of expressing knowledge by a machine with appropriate reasoning and different mathematics involved behind it.
4. Learn how to reason when an agent has only uncertain information about its task.
5. Know various supervised and unsupervised learning algorithms.

Outcomes:

Students will be able to:

1. Formalize a problem in the language/framework of different AI methods.
2. Illustrate basic principles of AI in solutions that require problem solving, search, Inference.
3. Represent natural language/English using Predicate Logic to build knowledge through various representation mechanisms.
4. Demonstrate understanding of steps involved in building of intelligent agents, expert systems, Bayesian networks.
5. Differentiate between learning paradigms to be applied for an application.

UNIT – I

Problem Solving & Search: Introduction- introduction to intelligenceFoundations of artificial intelligence (AI). History of AI, Structure of Agents.

Problem Solving - Formulating problems, problem types, states and operators, statespace. **Search Strategies. -** Informed Search Strategies- Best first search, A* algorithm, heuristic functions, Iterative deepening A*.

Adversarial Search/ Game playing - Perfect decision game, imperfect decision game, evaluation function, alpha-beta pruning.

UNIT – II

Knowledge, Reasoning &Planning:Reasoning - Knowledge based agent, Propositional Logic, Inference, Predicate logic (first order logic),Resolution

Structured Knowledge Representation – Frames, Semantic Nets

Planning - A Simple Planning Agent, Form Problem Solving to Planning, Basic representation of plans, partial order planning, hierarchical planning.

UNIT – III

Expert Systems, Reasoning with Uncertainty: Expert System and Applications: Introduction, Phases in Building Expert Systems, Expert System Architecture, Applications. **Uncertainty -** Basic probability, Bayes rule, Belief networks, Inference in Bayesian Networks**,** Fuzzy sets, and fuzzy logic: Fuzzy logic system architecture, membership function.

Decision Making- Utility theory, utility functions.

UNIT – IV

Learning: Machine-Learning Paradigms: Introduction, Machine Learning Systems, Supervised and Unsupervised Learning, Inductive Learning, Learning Decision Trees

Artificial Neural Networks: Introduction, Artificial Neural Networks, Single-Layer Feed- Forward Networks, Multi-Layer Feed-Forward Networks

Reinforcement learning: Learning from rewards, Passive and Active reinforcement learning, Applications.

UNIT – V

Communicating & Perceiving: Introduction to NLP- Progress & applications of NLP, Components of NLP, Grammars, Parsing.

Automatic Speech Recognition (ASR) – Speech Processing, Ex: DRAGON, HARPY,

Machine Vision – Applications, Basic Principles of Vision, Machine vision techniques: Low, Middle and High-level vision.

AI Today & Tomorrow - Achievements, ubiquitous AI.

Suggested Readings:

1. Stuart Russell and Peter Norvig. *Artificial Intelligence – A Modern Approach*, 3rd Edition, Pearson Education Press, 2009.
2. Kevin Knight, Elaine Rich, B. Nair, *Artificial Intelligence*, 3rd Edition, McGraw Hill, 2008.
3. Nils J. Nilsson, *The Quest for Artificial Intelligence*, Cambridge University Press, 2009.

COMPUTER NETWORKS

PC 505 CS

Instruction: 3L periods per week *Duration of SEE: 3 hours*
CIE:30marks *SEE: 70marks*
Credits: 3

Objectives:

1. To develop an understanding of communication in modern network architectures from a design and performance perspective.
2. To understand Data Transmission standards and MAC protocols.
3. To introduce the protocols functionalities in Network Layer and Transport Layer.
4. To understand DNS and supportive application protocols.
5. To provide basic concepts of Cryptography.

Outcomes:

Student will be able to:

1. Explain the functions of the different layer of the OSI and TCP/IP Protocol.
2. Understand wide-area networks (WANs), local area networks (LANs) andWireless LANs (WLANs) describe the function of each block.
3. Illustrate network layer and transport layer protocols. For a given problem related TCP/IP protocol developed the network programming.
4. Configure DNS , EMAIL, SNMP, Bluetooth, Firewalls using open source available software and tools.
5. Identify the types of encryption techniques.

UNIT – I

Data communication Components: Representation of data communication, flow of Networks, Layered architecture, OSI and TCP/IP model, Transmission Media. (William stalling).
Techniques for Bandwidth utilization: Line configuration, Multiplexing - Frequency division, Time division and Wave division, Asynchronous and Synchronous transmission , XDSL , Introduction to Wired and Wireless LAN

UNIT – II

Data Link Layer and Medium Access Sub Layer: Error Detection and Error Correction - Fundamentals, Block coding, Hamming Distance, CRC.
Flow Control and Error control protocols: Stop and Wait, Go back – N ARQ, Selective Repeat ARQ, Sliding Window, Piggybacking.
Multiple access protocols: Pure ALOHA, Slotted ALOHA, CSMA/CD,CDMA/CA

UNIT – III

Network Layer: Switching techniques (Circuit and Packet) concept.
Logical addressing: IPV4(Header), IPV6(Header), NAT , Sub-Netting concepts. **Inter-Networking:**Tunnelling , Fragmentation , congestion control (Leaky Bucket and Token Bucket algorithm), Internet control protocols: ARP, RARP, BOOTP and DHCP.**Network Routing Algorithms:** Delivery, Forwarding and Unicast Routing protocol, Gateway protocols.

UNIT – IV

Transport Layer: Process to Process Communication, Elements of transport protocol,
Internet Transport Protocols: UDP, TCP.**Congestion and Quality of Service**, QoS improving techniques.

UNIT – V

Application Layer: Domain Name Space (DNS), EMAIL, SNMP, Bluetooth.**Basic concepts of Cryptography:** Network Security Attacks, firewalls, symmetric encryption, Data encryptionStandards,publickeyEncryption(RSA),Hashfunction,Messageauthentication, Digital Signature.

Suggested Readings:

1. Data Communication and Networking, 4th Edition, Behrouz A.Forouzan,McGrawHill.

2. Data and Computer Communication, 8th Edition, William Stallings, Pearson PrenticeHall India.

3. W. Richard Stevens, Unix Network Programming, Prentice Hall / Pearson Education,2009

PROFESSIONAL ELECTIVE-I

PARALLEL AND DISTRIBUTED ALGORITHMS

PE511CS

Instruction: 3 periods per week — Duration of SEE: 3 hours

CIE: 30marks — SEE: 70marks

Credits: 3

Objectives:

1. To learn parallel and distributed algorithms development techniques for shared memory and message passing models.
2. To study the main classes of parallel algorithms.
3. To study the complexity and correctness models for parallel algorithms.

Outcomes:

Students must able to:

1. Analyze Parallel Algorithms.
2. Implement PRAM and Basic Algorithms.
3. Instrument with Shared Memory Algorithms.
4. Design Sorting and Selection Networks.
5. Understand Distributed Algorithms

UNIT-I

Introduction to Parallelism: Parallel Processing Ups and Downs, Types of Parallelism: A Taxonomy

Parallel Algorithms: Simple Computations and Architectures, Algorithms for a Linear Array, Algorithms for a Binary Tree, Algorithms for a 2D Mesh, Algorithms withShared Variables.

UNIT-II

PRAM and Basic Algorithms:PRAM Sub-models and Assumptions , Data Broadcasting, Semi-group or Fan, In Computation , Parallel Prefix Computation , Ranking the Elementsof a Linked List , Matrix Multiplication.

UNIT-III

Shared Memory Algorithms: Sequential Rank Based Selection, A Parallel Selection Algorithm, A Selection Based Sorting Algorithm, Alternative Sorting Algorithms, Convex Hull of a 2D Point Set, Bitonic sort algorithm.

UNIT-IV

Mesh Base Architectures: Recursive Sorting Algorithms, Greedy Routing Algorithms, GraphAlgorithms, Image Processing Algorithms.

UNIT-V

Distributed Algorithms: Models and complexity measures, Safety, liveness, termination, logical time and event ordering, Global state and snapshot algorithms, Mutual exclusion and Clock Synchronization, Distributed Graph algorithms.

Suggested Readings

1. Ananth Grama, Anshul Gupta, George Karypis and Vipin Kumar, Introduction to Parallel Computing, second edition, Addison- Wesley/Pearson, 1994/2003.

2. Introduction to Parallel Processing Algorithms and Architectures, Behrooz Parhami, Platinum series of computer science.

3. Vijay K. Garg, "Elements of Distributed computing", Wiley Joseph F Jájá, An Introduction to Parallel Algorithms, Addison Wesley, 1992.

4. Michael J Quinn, Parallel Programming in C with MPI and OpenMP, first edition, McGraw Hill, 2004/2003.

5. Michael J Quinn, Parallel Computing: Theory and Practice, second edition, McGraw Hill, 1994/2002.

6. Nancy Lynch, Distributed Algorithms, Morgan Kaufmann.

7. Andrew S. Tanenbaum, Distributed Operating Systems, ACM Press.

EMBEDDED SYSTEMS

PE512CS

Instruction: 3 periodsperweek
Duration of SEE: 3 hours
CIE: 30marks
SEE: 70marks
Credits: 3

Objectives:

1. Know Embedded system compared to General Purpose System.
2. Learn the typical core of Embedded process Modelling, Bus Communication in Processor, Input/ Output interfacing
3. To introduce Process scheduling algorithms, Basic Real time operating system.

Outcomes:

Student will be able to

1. Understand embedded system, also recognize the classification of embedded systems.
2. Justify the importance of hardware software co-design and models involved.
3. Understand the ability how communication buses work, Interrupts and service mechanism.
4. Design real time embedded systems using the concepts of RTOS.
5. Analyze various examples of embedded systems.

UNIT – I

Introduction to embedded systems: Embedded systems, Processor embedded into a system, Embedded hardware units and device in a system, Embedded software in a system, Examples of embedded systems, Design process in embedded system, Formalization of system design, Design process and design examples, Classification of embedded systems, skills required for an embedded system designer.

UNIT – II

Devices and communication buses for devices network: IO types and application with Keyboards , Serial communication devices, Parallel device ports, Sophisticated interfacing features in device ports, Wireless devices, Timer and counting devices, Watchdog timer, Real time clock, Networked embedded systems, Serial bus communication protocols, Parallel bus device protocols-parallel communication internet using ISA, PCI, PCI-X and advanced buses, Internet enabled systemsnetwork protocols, Wireless and mobile system protocols.

UNIT – III

Device drivers and interrupts and service mechanism: Programming-I/O busy-wait approach without interrupt service mechanism, ISR concept, Interrupt sources, Interrupt servicing (Handling) Mechanism, Multiple interrupts, Context and the periods for context switching, interrupt latency and deadline, Classification of processors interrupt service mechanism from Context-saving angle, Direct memory access, Device driver programming.

UNIT – IV

Inter process communication and synchronization of processes, Threads and tasks: Multiple process in an application, Multiple threads in an application, Tasks, Task states, Task and Data, Clear-cut distinction between functions. ISRS and tasks by their characteristics, concept and semaphores, Shared data, Interprocess communication, Signal function, Semaphore functions, Message Queue functions, Mailbox functions, Pipe functions, Socket functions, RPC functions.

Unit –V

Real-time operating systems: OS Services, Process management, Timer functions, Event functions, Memory management, Device, file and IO subsystems management, Interrupt routines in RTOS environment and handling of interrupt source calls, Real-time operating systems, Basic design using an RTOS, RTOS task scheduling models, interrupt latency and response of the tasks as performance metrics, OS security issues. Introduction to embedded software development process and tools, Host and target machines, Linking and location software.

Suggested Readings:

1. Microcontroller and Embedded Systems Using Assembly and C (Second Edition),- Muhammed Ali Mazidi ,Janice Gillispie Mazidi, Rolin D. McKinlay ;2008;Pearson Publication ; ISBM : 978-81-317-1026-5 .

2. Raj Kamal, "Embedded Systems", 2nd edition, Tata McGraw Hill, 2009.

3. Peter Barry and Patric Crowley, Intel architecture for Embedded system.

4. Wayne Wolf, "Computers as Components-principles of Embedded Computer system Design", 1st edition, Elseveir, 2009.

5. Tammy Noergaard, "Embedded System Architecture, A comprehensive Guide for Engineers and Programmers", Elsevier, 2006.

COMPUTER GRAPHICS

PE513CS

Instruction: 3 periods per week
Duration of SEE: 3 hours
CIE: 30marks
SEE: 70marks
Credits: 3

Objectives:

1. To introduce the concept of synthetic camera model, programmable pipeline and OpenGL API.
2. To study different interaction modes and data structures that store 2-D and 3-D geometric objects.
3. To understand different transformations in 2-D and 3-D.
4. To study different rasterization and rendering algorithms.

Outcomes:

Student will be able to

1. Describe the steps in graphics programming pipeline.
2. Write interactive graphics applications using OpenGL geometric primitives.
3. Apply affine transformations for viewing and projections.
4. Use of geometric transformations on graphics objects and their application in composite form.
5. Create realistic images of 3-d objects that involve lighting shading aspects.

UNIT – I

Graphics Systems and Models: Graphics system, Images, Physical and Synthetic, Imaging system,Syntheticcameramodel,Programminginterface,Graphicsarchitectures,Programmable pipelines.

Graphics Programming: Programming two-dimensional applications, OpenGL API, Primitives and attributes, Color, Viewing and Control functions.

UNIT – II

Input and Interaction: Input devices, Display lists & modeling, Programming event-driven input, Picking, building interactive models, Animating interactive programs, Logic operations.

Geometric Objects: Three-dimensional primitives, Coordinate systems and frames, Frames in OpenGL, Modeling colored cube.

UNIT – III

Transformations: Affine transformations, Transformations in homogeneous coordinates, Concatenation of transformations, OpenGL transformation matrices.

Viewing: Classical and Computer views, Viewing with a computer, Positioning of camera, Simple projections, Projections in OpenGL, Hidden surface removal, Parallel-projection matrices, Perspective-projection matrices.

UNIT – IV

Lighting and Shading: Light sources, The Phong lighting model, Computational vectors, Polygonal shading, Light sources in OpenGL, Specification of matrices in OpenGL, Global illumination.

From Vertices to Frames: Basic implementation strategies, Line-segment clipping, Polygon clipping, Clipping in three dimensions, Rasterization, Anti-aliasing.

UNIT – V

Modeling & Hierarchy: Hierarchal models, Trees and traversal, Use of tree data structure, Animation, Graphical objects, Scene graphs, Simple scene graph API, Open Scene graph, Other tree structures.

Suggested Readings:

1. Edward Angel, Interactive Computer Graphics: A Top-Down Approach Using OpenGL, Pearson Education, 5th edition, 2009.
2. Francis S Hill Jr., Stephen M Kelley, Computer Graphics using OpenGL, Prentice-Hall Inc., 3rd Edition, 2007.
3. Jim X. Chen, Foundations of 3D Graphics Programming using JOGL and Java3D, Springer Verlag, 2006.
4. Hearn Donald, Pauline M Baker, Computer Graphics, 2nd edition, 1995.

OBJECT ORIENTED ANALYSIS AND DESIGN

PE514CS

Instruction: 3 periods per week — Duration of SEE: 3 hours

CIE: 30marks — SEE: 70marks

Credits: 3

Objectives

To understand the Object-based view of Systems
To develop robust object-based models for Systems
To inculcate necessary skills to handle complexity in software design

Learning Outcomes

1. Ability to analyze and model software specifications.
2. Ability to abstract object-based views for generic software systems.
3. Ability to deliver robust software components.
4. Design Class and Object Diagrams that represent Static Aspects of a Software System
5. Analyze Dynamic Aspects of a Software System using Use Case, Interaction and Activity Diagrams.

UNIT – I

Object-Oriented Analysis and Design: Introduction, UML
Iterative, Evolutionary, and Agile: Introduction, Unified Process, Agile Modeling, Agile Unified Process.Case Studies: Basics
Inception&Usecases: Introduction, Evolutionary Requirements, Use Cases, Usecase Diagrams, Activity Diagrams.

UNIT – II

Elaboration & Domain Models: Iteration 1Requirements and Emphasis: Core OOA/D Skills, Domain Models, Class Diagrams, System Sequence Diagrams, Requirements to Design,
Package Diagram and UML Tools: Logical Architecture, Software Architecture, Package Diagrams, On to Object Design, UML CASE Tools, UML Class Diagrams.

UNIT – III

UML Class Diagrams, UML Interaction Diagrams, UML Activity Diagram and Modelling, Mapping Design to Code, UML State Machine Diagram and Modelling, Test Driven Development and Agile Concepts, Documenting Architecture, Case Studies.

UNIT – IV

UML Deployment and Component Diagram, GoF Design Patterns and Iterative Planning, Introduction to GRASP – Methodological approach to OO Design, Architectural analysis and UML Package Design.

UNIT – V

Test Driven Development and Agile Concepts, Documenting Architecture, Case Studies.

Suggested Readings:

1. Craig Larman, "Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development", 3rd edition, Pearson 2008
2. Grady Booch, James Rumbaugh, Ivar Jacobson (2009), The Unified Modeling Language User guide, 2nd edition, Pearson Education, New Delhi, India.
3. Cay Horstmann), Object-Oriented Design and Patterns, Wiley India edition, New Delhi, India.
4. Meilir Page-Jones (2000), Fundamentals of Object Oriented Design in UML, Pearson Education and NewYork.
5. John W. Satzinger, Robert B Jackson, Stephen D Burd (2004), Object-Oriented Analysis and Design with theUnified Process, Cengage learning, India.

DATA SCIENCE

PE515CS

Instruction: 3 periods per week Duration of SEE: 3 hours

CIE: 30marks SEE: 70marks

Credits: 3

Objectives:

1. To understand the concepts of Data Science and their related mathematical concepts.
2. To learn various statistical concepts like linear and logistic regression and cluster analysis.
3. To learn basics of R Programming environment: R language, R-studio andR packages
4. To learn Predictive models and their real time applications.

Outcomes:

Students must be able to:

1. Use various data structures and packages in R for data visualization and summarization
2. Use linear, non-linear regression models, and classification techniques for data analysis
3. Use clustering methods including K-means.
4. Apply Basic Machine Learning Algorithms.
5. Utilize Matrix decomposition techniques to perform data analysis

UNIT – I

Data Science: Introduction to data science, Linear Algebra for data science, Linear equation,Distance, Eigen values, Eigenvectors, Dimensionality reduction.

UNIT II

Statistical Modeling, Random variables, sample statistics, Inference: Statistical Hypothesis Testing, Confidence Intervals, P hacking, Bayesian Inference.

UNIT III

Predictive Modeling: Linear Regression, Simple Linear Regression model building, Multiple Linear Regression, Logistic regression.

UNIT IV

Introduction to R Programming, getting started with R: Installation of R software and using the interface, Variables and data types, R Objects, Vectors and lists, Operations: Arithmetic, Logical and Matrix operations, Data frames, functions, Control structures, Debugging and Simulation in R.

UNIT V

Classification: performance measures, Logistic regression implementation in R, K-Nearest neighbors (KNN), K-Nearest neighbors implementation in R, Clustering: K-Means Algorithm, K-Means implementation in R.

Case Studies of Data Science Application Weather forecasting, Stock market prediction, Object recognition, Real Time Sentiment Analysis.

Suggested Readings:

1. Nina Zumel, Practical Data Science with R, ManningPublications,2014.
2. Peter Bruce and Andrew Bruce, Practical Statistics for Data Scientists,O'Reilly,2017.

BLOCK CHAIN TECHNOLOGY

PE516CS

Instruction: 3 periodsperweek
Duration of SEE: 3 hours
CIE: 30marks
SEE: 70marks
Credits: 3

Objectives:

1. Understand how block chain systems (mainly Bitcoin and Ethereum)work.
2. To securely interact withthem.
3. Design, build, and deploy smart contracts and distributed applications.
4. Integrate ideas from block chain technology into their ownprojects.

Outcomes:

Student will be able to

1. Explain design principles of Bitcoin andEthereum.
2. Explain the Simplified Payment Verification protocol.
3. List and describe differences between proof-of-work and proof-of-stakeconsensus.
4. Interact with a block chain system by sending and reading transactions.
5. Design, build, and deploy a distributed application.

UNIT – I

Basics: Distributed Database, Two General Problem, Byzantine General problem and Fault Tolerance, Hadoop Distributed File System, Distributed Hash Table, ASIC resistance, Turing Complete.

Cryptography: Hash function, Digital Signature - ECDSA, Memory Hard Algorithm, Zero Knowledge Proof.

UNIT – II

Blockchain: Introduction, Advantage over conventional distributed database, Block chain Network, Mining Mechanism, Distributed Consensus, Merkle Patricia Tree, Gas Limit, Transactions and Fee, Anonymity, Reward, Chain Policy, Life of Block chain application, Soft & Hard Fork, Private and Public block chain.

UNIT – III

Distributed Consensus: Nakamoto consensus, Proof of Work, Proof of Stake, Proof of Burn, Difficulty Level, Sybil Attack, Energy utilization and alternate.

UNIT – IV

Cryptocurrency: History, Distributed Ledger, Bitcoin protocols - Mining strategy and rewards, Ethereum -
Construction, DAO, Smart Contract, GHOST, Vulnerability, Attacks, Sidechain, Name coin

UNIT – V

Cryptocurrency Regulation: Stakeholders, Roots of Bit coin, Legal Aspects-Crypto currency Exchange, Black Market and Global Economy.

Applications: Internet of Things, Medical Record Management System, Domain Name Service, and future of Block chain.

Case study: Naive Blockchain construction, Memory Hard algorithm - Hashcash implementation, Direct Acyclic Graph, Play with Go-Ethereum,Smart Contract Construction, Toy application using Blockchain, Miningpuzzles

Suggested Readings:

1. Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller and Steven Goldfeder, Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction, Princeton University Press, 2016.
2. Satoshi Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System.
3.DR.GavinWood,"ETHEREUM:ASecureDecentralizedTransactionLedger,"Yellow paper.2014.
4. Nicola Atzei, Massimo Bartoletti, and TizianaCimoli, A survey of attacks on Ethereum smart contracts.

PRACTICAL/ LABORATORY COURSES

SOFTWARE ENGINEERING LAB

PC551CS

Instruction:2 periodsperweek Duration of SEE: 3 hours

CIE: 25marksSEE: 50marks

Credits: 1

Objectives:

1. To understand the software engineering methodologies for project development.
2. To gain knowledge about open-source tools for Computer Aided Software Engineering (CASE).
3. To develop test plans and test cases to perform various testing.

Outcomes:

Student will be able to

1. Analyze and design software requirements in an efficient manner.
2. Use open-source case tools to develop software.
3. Implement the design, debug and test the code.
4. Ability to generate a high-level design of the system from the software requirements.
5. Ability to translate end-user requirements into system and software requirements.

I. FORWARD ENGINEERING

Students have to form a team with a batch size of two or three and take up a **case study- based project** to analyze, plan, design UML models and create a prototypical model (identifying deliverables) by coding the developed designs and finally documenting considering any one example of the following domains: -

1. Academics (Course Registration System, Student marks analyzingsystem)
2. Health Care (Expert system to prescribe medicines for given symptoms,Remote Diagnostics, Patient/Hospital ManagementSystem)
3. Finance (Banking: ATM/NetBanking, UPI:PayTM/Phone Pay,Stocks:Zerodha)
4. E-Commerce (various online shopping portals likeFlipKart/Amazon/Myntra)
5. Logistics (Postal/Courier:IndiaPost/DTDC/UPS/FedEx,Freight:Maersk)
6. Hospitality (Tourism Management:Telangana Tourism/Incredible India, Event Management: MeraEvents/BookMyShow/Explara/EventBrite)
7. Social Networking (LinkedIn, FaceBook, Shaadi.com, BharatMatrimony,Tinder)
8. Customer Support (Banking Ombudsman,Indian Consumer ComplaintsForum)
9. Booking/Ticketing(Food:Zomato/Swiggy/BigBasket/Grofers/JioMart, Hotel:OYO/Trivago or Travel: {Cars:Uber/OLA/Zoom, Railways:IRCTC, Buses:OnlineTSRTC/RedBus/AbhiBus, Flights:MakeMyTrip/Goibibo,Ships:Lakport})

II. REVERSE ENGINEERING

Students have to refer any project repository: GitLab/GitHub, execute the code in order to observe its functionalities/features/requirements and by the help of any tool derive the designs from the code for understanding the relationships among various subsystems/classes/components and if the tool partially generates models then identify by associating elements to judge/mark the appropriate relationships.

III. TESTING

Prepare Test Plan and develop Test Case Hierarchy to monitor or uncover/report errors using manual/automated testing tools

Software Required:

1. StarUML/Umbrello, NetBeans/Eclipse IDE, XAMPP/MEAN stack, JUnit, JMeter, Selenium, Bugzilla

Artificial Intelligence Lab

PC 553 CS
Instruction: 2P periods per week *Duration of SEE: 3 hours CIE:25marks*
SEE: 50marks
Credits: 1

Course Objectives :

- To apply programming skills to formulate the solutions for computational problems.
- To study implementation first order predicate calculus using Prolog
- To familiarize with basic implementation of NLP with the help of Python libraries NLTK
- To understand python library scikit-learn for building machine learning models
- To enrich knowledge to select and apply relevant AI tools for the given problem

Course Outcomes

1. Design and develop solutions for informed and uninformed search problems in AI.
2. Demonstrate reasoning in first order logic using Prolog
3. Utilize advanced package like NLTK for implementing natural language processing.
4. Demonstrate and enrich knowledge to select and apply python libraries to synthesize information and develop supervised learning models
5. Develop a case study in multidisciplinary areas to demonstrate use of AI

Prerequisite: Basics of programming in Python

1. Write a program to implement Uninformed search techniques:
a. BFS
b. DFS

2. Write a program to implement Informed search techniques
a. Greedy Best first search
b. A* algorithm

3. Study of Prolog, its facts, and rules.
a. Write simple facts for the statements and querying it.
b. Write a program for Family-tree.

4. Write a program to train and validate the following classifiers for given data (scikit-learn):
a. Decision Tree
b. Multi-layer Feed Forward neural network

5. Text processing using NLTK
a. Remove stop words
b. Implement stemming
c. POS (Parts of Speech) tagging

In addition to the above programs, students should be encouraged to study implementations of one of the following
- Game bot (Tic Tac toe, 7 puzzle)
- Expert system (Simple Medical Diagnosis)
- Text classification
- Chat bot

COMPUTER NETWORKS LAB

PC 553 CS

Instruction: 2P periodsperweek *Duration of SEE: 3 hours*
CIE:25marks *SEE: 50marks*
Credits: 1

Prerequisite:

1. Data Communications

Objectives:

1. Learn to communicate between two desktop computers.
2. Learn to implement the different protocols
3. Be familiar with socket programming.
4. Be familiar with the various routing algorithms
5. Be familiar with simulation tools.
6. To use simulation tools to analyze the performance of various networkprotocols

Outcomes:

1. Implement various protocols using TCP and UDP.
2. Program using sockets.
3. Use simulation tools to analyze the performance of various network protocols.
4. Implement cryptographic algorithms.
5. Implement and Analyze various routing algorithms.

List of Experiments:

1. Running and using services/commands like tcpdump, netstat, ifconfig, nslookup, FTP, TELNET and traceroute. Capture ping and trace route PDUs using a network protocol analyzer and examine.
2. Configuration of router, switch . (using real devices or simulators)
3. Socket programming using UDP and TCP (e.g., simple DNS, data & time client/server, echo client/server, iterative & concurrent servers)
4. Network packet analysis using tools like Wireshark, tcpdump, etc.
5. Network simulation using tools like Cisco Packet Tracer, NetSim, OMNeT++, NS2, NS3, etc.
6. Study of Network simulator (NS) and Simulation of Congestion Control Algorithms using NS. Performance evaluation of Routing protocols using Simulationtools.
7. Programming using raw sockets
8. Programming using RPC

Note: The Instructor may add/delete/modify/tune experiments, wherever he/she feels in a justified manner.

Laboratory requirement for students:

Hardware:

1. Standalone desktops

Software:

1. C / C++ / Java / Python / Equivalent Compiler
2. Network simulator like NS2/NS3/OPNET/ CISCO Packet Tracer / Equivalent

SCHEME OF INSTRUCTION & EXAMINATION
B.E (Computer Science and Engineering)
SEMESTER-VI

S. No	Course Code	Course Title	Scheme of Instruction				SchemeofExamination			Credits
			L	T	D/P	Contact Hrs/Wk	CIE	SEE	Duration in Hrs/Wk	
Theory Courses										
1	PC 601 CS	Compiler Design	3	1	-	4	30	70	3	3
2	PC 602 CS	Design and Analysis of Algorithms	3	1	-	4	30	70	3	3
3	PC 603 CS	Machine learning	3	1	-	4	30	70	3	3
4	PC-604 CS	Cryptography and Network Security	3	-	-	3	30	70	3	3
5	PE-62X CS	Professional Elective –II	3	-	-	3	30	70	3	3
6	OE-I	Open Elective-I	3	-	-	3	30	70	3	3
Practical/Laboratory Courses										
7	PC 651 CS	Machine learning LAB	-	-	2	2	25	50	3	1
8	PC 652 CS	Design and Analysis of Algorithms Lab	-	-	2	2	25	50	3	1
9	PW 653 CS	Mini Project	-	-	4	4	25	50	3	2
10	SI 671 CS	Summer Internship*	-	-	-	-	-	-	-	-
Total			**18**	**3**	**8**	**29**	**255**	**570**		**22**

Profession Elective – II	
Course Code	**Course Title**
PE 621 CS	Quantum Computing
PE 622 CS	Advanced Computer Architecture
PE 623 CS	Image Processing
PE 624 CS	Software Testing
PE 625 CS	Data Mining
PE 626 CS	Mobile Computing

Open Elective – I	
Course Code	**Course Title**
OE601 CS	OOP using Java (Not for CSE Students)
OE602 CS	Data Structure and Algo. (Not for CSE Students)

COMPILER DESIGN

PC 601 CS

Instruction: 3L+1T periodsperweek *Duration of SEE: 3 hours*
CIE:30marks *SEE: 70marks*
Credits: 3

Objectives:

1. To understand and list the different stages in the process of compilation.
2. Identify different methods of lexical analysis
3. Design top-down and bottom-up parsers
4. Identify synthesized and inherited attributes
5. Develop syntax directed translation schemes
6. Develop algorithms to generate code for a target machine

Outcomes:

Upon completion of the course, the students will be able to:

1. For a given grammar specification, develop the lexical analyzer.
2. For a given parser specification, design top-down and bottom-up parsers.
3. Develop syntax directed translation schemes.
4. Develop algorithms to generate code for target machine.
5. Choose a data structures for symbol table organization and dynamic memory management.

UNIT-I

Introduction: The Structure of a Compiler, Phases of Compilation, The Translation Process, Major Data Structures in a Compiler, Bootstrapping and Porting.

Lexical Analysis (Scanner): The Role of the Lexical Analyzer, Input Buffering, Specification of Tokens, Recognition of Tokens, The Lexical Analyzer Generator Lex.

UNIT-II

Syntax Analysis (Parser): The Role of the Parser, Syntax Error Handling and Recovery, Top-Down Parsing, Bottom-Up Parsing, Simple LR Parsing, More Powerful LR Parsing, Using Ambiguous Grammars, Parser GeneratorYaac.

UNIT-III

Syntax-Directed Translation: Syntax-Directed Definitions, Evaluation Orders for SDD's Applications of Syntax-Directed Translation.

Symbol Table: Structure, Operations, Implementation and Management.

UNIT-IV

Intermediate Code Generation: Variants of Syntax Trees, Three-Address Code, Types and Declarations, Translation of Expressions, Type Checking, Control Flow, Backpatching, Switch-statements, Intermediate Code for Procedures.

Run-time environment: Storage Organization, Stack Allocation of Space, Accessto Nonlocal Data on the Stack, Parameter passing, Heap Management and Garbage Collection.

UNIT-V

Code Generation: Issues in the Design of a Code Generator, The Target Language, Addresses in the Target Code, Basic Blocks and Flow graphs, Optimization of Basic Blocks, Peephole Optimization, Register Allocation andAssignment.

Machine-Independent Optimizations: The Principal Sources of Optimizations, Introduction to Data-FlowAnalysis.

Suggested Readings:

1. Alfred V. Aho, Monica S. Lam, Ravi Sethi, & Jeffrey D. Ullman , *Compilers :Principles, Techniques and Tools*, 2nd Edition, Pearson Education, 2006.
2. Kenneth C. Louden, *Compiler Construction: Principles and Practice*, *Thomson Learning Inc., 1997.*
3. P.Trembley and P.S.Sorenson, ***The Theory and Practice of Compiler Writing***, TMH-1985.

Course Code	CourseTitle	Core/Elective
PC 602CS	**DESIGN AND ANALYSIS OF ALGORITHMS**	Core

Prerequisite	Contact hours per week L	T	D	P	CIE	SEE	Credits
-	3	1	-	-	**30**	**70**	**3**

CourseObjectives:

- Toreviewelementarydatastructures ,ordernotationandalgorithmanalysis
- Tolearnalgorithmdesignstrategiessuchas Divide-and-Conquer,greedymethod,dynamicprogramming,backtrackingandbranch&boundtechnique
- Tounderstandtheconceptsof NP-hardandNP-complete

CourseOutcomes:

Studentwillbeableto:

1. Design algorithms for various computing problems
2. Analyze the time and space complexity of algorithms
3. Critically analyze the different algorithm design techniques for a given problem.
4. Modify existing algorithms to improve efficiency.
5. Identify the complexity class of a given problem

UNIT–I

Introduction & Elementary Data Structures: Introduction, Fundamentals of algorithm(Line Count, Operation Count), Analysis of algorithms (Best, Average, Worst case), Asymptotic Notations(O,Ω,Θ) Recursive Algorithms, Analysis using Recurrence Relations, Master's Theorem.
Review of elementary datastructures – Graphs: BFS, DFS, Bi-Connected Components. Sets: representation, UNION, FIND operations.

UNIT–II

Divide-and-Conquer Method: The general method, Binary search, Finding maximum and minimum, Mergesort, Quicksort.

Brute Force: Knapsack,Traveling salesman problem, Convex-Hull

UNIT–III

Greedy Method: Knapsack problem, Minimum spanning trees, Single source shortest path, Job sequencing with deadlines, Optimal storage on tapes, Optimal merge pattern
Dynamic programming method: All pairs shortest paths,Optimal binary search tress, 0/1 Knapsack problem, Reliability design, Traveling salesman problem,

UNIT–IV

Backtracking*: N-queens problem, Graph coloring, Hamiltonian cycles*
Branch-and-bound: FIFO & LC branch and Bound methods, 0/1Knapsack problem,Travelingsalesperson

UNIT–V

NP-hard and NP-complete problems: Non Deterministic algorithms, The classes: P, NP, NP Complete, NP Hard, Satisfiability problem, Proofs for NP Complete Problems: Clique, Vertex Cover.
Parallel Algorithms: I*ntroduction, models for parallel computing.*

Suggested Reading:

1. HorowitzE, Sahni S,FundamentalsofComputer Algorithms,2nd Edition,UniversitiesPress,2007
2. Thomas H.Cormen, Charles E.Leiserson, Ronald L. Rivest and Clifford Stein, "Introduction to Algorithms", Third Edition, PHI Learning Private Limited, 2012
3. MichaelT.Goodrich,RobertoTamassia,AlgorithmDesign:Foundations,AnalysisandInternetExamples,JohnWiley&Sons,2002

MACHINE LEARNING

PC 603 CS

Instruction: 3L+1T periods per week *Duration of SEE: 3 hours*
CIE:30 marks *SEE: 70marks*
Credits: 3

Objectives:

1. To learn the concepts of machine learning and types of learning along with evaluation metrics.
2. To study various supervised learning algorithms.
3. To learn ensemble techniques and various unsupervised learning algorithms.
4. To explore Neural Networks and Deep learning basics.
5. To learn reinforcement learning and study applications of machine learning.

Outcomes:

Students must be able to:

1. Extract features that can be used for a particular machine learning approach in various applications.
2. Compare and contrast pros and cons of various machine learning techniques and to get an insight when to apply particular machine learning approach.
3. Understand different machine learning types along with algorithms.
4. Understand how to apply machine learning in various applications.
5. Apply ensemble techniques for improvement of classifiers.

UNIT-I

Introduction: Representation and Learning: Feature Vectors, Feature Spaces, Feature Extraction and Feature Selection, Learning Problem Formulation

Types of Machine Learning Algorithms: Parametric and Nonparametric Machine Learning Algorithms, Supervised, Unsupervised, Semi-Supervised and Reinforced Learning.

Preliminaries: Overfitting, Training, Testing, and Validation Sets, The Confusion Matrix, Accuracy Metrics: Evaluation Measures: SSE, RMSE, R2, confusion matrix, precision, recall, F-Score, Receiver Operator Characteristic (ROC) Curve. Unbalanced Datasets. some basic statistics: Averages, Variance and Covariance, The Gaussian, the bias-variance tradeoff.

UNIT-II

Supervised Algorithms

Regression: Linear Regression, Logistic Regression, Linear Discriminant Analysis.

Classification: Decision Tree, Naïve Bayes, K-Nearest Neighbors, Support Vector Machines, evaluation of classification: cross validation, hold out, .

UNIT-III

Ensemble Algorithms: Bagging, Random Forest, Boosting

Unsupervised Learning:

Cluster Analysis: Similarity Measures, , categories of clustering algorithms, k-means, Hierarchical, Expectation-Maximization Algorithm, Fuzzy c-means algorithm.

UNIT-IV

Neural Networks: Multilayer Perceptron, Back-propagation algorithm, Training strategies, Activation Functions, Gradient Descent For Machine Learning, Radial basis functions, Hopfield network, Recurrent Neural Networks.

Deep learning: Introduction to deep learning, Convolutional Neural Networks (CNN), CNN Architecture, pre-trained CNN (LeNet, AlexNet).

UNIT-V

Reinforcement Learning: overview, example: getting lost, State and Action Spaces, The Reward Function, Discounting, Action Selection, Policy, Markov decision processes, Q-learning, uses of Reinforcement learning

Applications of Machine Learning in various fields: Text classification, Image Classification, Speech Recognition.

Suggested Books(

1. Machine Learning: An Algorithmic Perspective, Stephen Marsland, Second Edition (Chapman & Hall/Crc Machine Learning & Pattern Recognition) (2014)
2. Machine Learning, Tom Mitchell, McGraw-Hill Science/Engineering/Math; (1997).
3. Deep Learning by Ian Goodfellow, Yoshua Bengio and Aaron Courville, MIT Press(2017)
4. Trevor Hastie, Robert Tibshirani, Jerome Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Second Edition, Springer Series in Statistics.(2009)
5. Pattern Recognition and Machine Learning, Christopher M. Bishop, Springer. (2006)
6. An Introduction to Pattern Recognition and Machine Learning, M Narasimha Murty,V Susheela Devi, IISc Press.
7. Uma N. Dulhare , Khaleel Ahmad , Khairol Amali Bin Ahmad , Machine Learning and Big Data: Concepts, Algorithms, Tools and Applications, Scrivener Publishing,Wiley,2020

CRYPTOGRAPHY AND NETWORK SECURITY

PC 604 CS

Instruction: 3L+1T periods per week *Duration of SEE: 3 hours*
CIE:30 marks *SEE: 70marks*
Credits: 3

Course Objectives:-

- Understand security concepts, Ethics in Network Security.
- Obtain knowledge on mechanisms to encounter threats
- Appreciate and apply relevant cryptographic techniques
- Apply authentication services and security mechanisms
- Comprehend computer network access control and ethics in network security.

Course Outcomes: At the end of the course the students will be able to -

- Develop familiarity with cryptography and security techniques
- Master fundamentals of secret and public cryptography
- Utilize the master protocols for security services
- Identify network security threats and counter-measures
- Propose network security designs using available secure solutions

UNIT- I

Basic Principles: Security Goals, Cryptographic Attacks, Services and Mechanisms, Mathematics of Cryptography

UNIT –II

Symmetric Encryption: Mathematics of Symmetric Key Cryptography, Introduction to Modern Symmetric Key Ciphers,Data Encryption Standard, Advanced Encryption Standard.

UNIT- III

Asymmetric Encryption: Mathematics of Asymmetric Key Cryptography, Asymmetric Key Cryptography

UNIT –IV

Data Integrity, Digital Signature Schemes & Key Management: Message Integrity and Message Authentication, Cryptographic Hash Functions, DigitalSignature, Key Management

UNIT-V

Network Security: Security at Application layer (PGP and S/MIME), Security at the Transport Layer(SSL and TLS), Security at the Network Layer(IPSec, System Security)

Text Book

1. Cryptography and Network Security, Behrouz A Forouzan, Debdeep Mukhopadhyay, (3e) Mc Graw Hill.

Reference Books

1. Cryptography and Network Security, William Stallings, (6e) Pearson.
2. Everyday Cryptography, Keith M.Martin, Oxford.
3. Network Security and Cryptography, Bernard Meneges, Cengage Learning
4. Cryptography and Network Security – by Atul Kahate – TMH.
5. Cyber Security Operations Handbook – by J.W. Rittiaghouse and William M.Hancok – Elseviers.
6. Khairol Amali Bin Ahmad , Khaleel Ahmad , Uma N. Dulhare ,Functional Encryption ,EAI/Springer Innovations in Communication and Computing,1st ed. 2021 Edition

PROFESSIONAL ELECTIVE –II

QUANTUM COMPUTING

PE621 CS

Instruction: 3L periods per weekDuration of SEE: 3 hours CIE:30marksSEE: 70marks
Credits: 3

Objectives:

1.To understand the fundamentals of quantum information processing.

2. To understand quantum computation.

3. Quantum cryptography.

4. Quantum information theory.

Outcomes:

Students must be able to:

1. Analyze the behavior of basic quantum algorithms
2. Implement simple quantum algorithms and information channels in the quantum circuit model.
3. Simulate a simple quantum error-correcting code.
4. Prove basic facts about quantum information channels.
5. Understand the framework of quantum computation, and how that be useful for future quantum technologies

UNIT-I

Introduction: Quantum Measurements Density Matrices, Positive-Operator Valued Measure, Fragility of quantum information: Decoherence, Quantum Superposition and Entanglement, Quantum Gates and Circuits.

UNIT-II

Quantum Basics and Principles: No cloning theorem & Quantum Teleportation, Bell's inequality and its implications, Quantum Algorithms & Circuits.

UNIT-III

Algorithms: Deutsch and Deutsch–Jozsa algorithms, Grover's Search Algorithm, Quantum Fourier Transform, Shore's Factorization Algorithm.

UNIT-IV

Performance, Security and Scalability: Quantum Error Correction: Fault tolerance; Quantum Cryptography, Implementing Quantum Computing: issues of fidelity; Scalability in quantum computing.

UNIT-V

Quantum Computing Models: NMR Quantum Computing, Spintronics and QED MODEL, Linear Optical MODEL, Nonlinear Optical Approaches; Limits of all the discussed approaches, Future of Quantum computing.

Suggested Readings:

1. Eric R. Johnston, NicHarrigan, Mercedes and Gimeno-Segovia "Programming Quantum Computers: Essential Algorithms And Code Samples, SHROFF/ O'Reilly
2. Dr. Christine Corbett Moran, Mastering Quantum Computing with IBM QX: Explore the world of quantum computing using the Quantum Composer and Qiskit, Kindle Edition Packet.
3. V.K Sahni, Quantum Computing (with CD), TATA McGraw Hill.
4. Chris Bernhardt, Quantum Computing for Everyone (The MIT Press).

ADVANCED COMPUTER ARCHITECTURE

PE622CS

Instruction: 3 periods per week
Duration of SEE: 3 hours
CIE: 30marks
SEE: 70marks
Credits: 3

Objectives:

An overview of computer architecture, which stresses the underlying design principles and the impact of these principles on computer performance. General topics include design methodology, processor design, control design, memory organization, system organization, and parallel processing.

Outcomes:

Student will be able to

1. Know the classes of computers, and new trends and developments in computer architecture
2. Understand pipelining, instruction set architectures, memory addressing.
3. Understand the performance metrics of microprocessors, memory, networks, and disks.
4. Understand the performance and efficiency in advanced multiple-issue processors.
5. Understand symmetric shared-memory architectures and their performance.

UNIT – I

Introduction - Introduction to computer architectureSoftware-hardware interface. Performance and Power. Performance metrics. Performance measurement. Benchmark programs.

UNIT – II

Instructions- Instruction Set. Operations. Operands and addressing modes. Role of compilers and system software. Understanding implementation of function calls and returns, array references, pointers.

UNIT – III

Computer Arithmetic- Signed integers. Floating point. Rounding and accuracy. Addition and Subtraction. Multiplication. Division

Processor - Data path elements. Data path control.

UNIT – IV

Pipelining - Speedup. Pipeline hazards. Stalling. Forwarding. Branch prediction. Exceptions. Speculation. Multiple issue.

Dynamic scheduling; Cache memory- Locality of reference. Cache organization and access. Multilevel caches. Performance. Cache coherence.

UNIT – V

Virtual Memory- Hardware support for address translation, page fault handling. Translation look aside buffer, Hardware-software interface.

Input/Output-Harddisk. Flash memory. I/O interfacing. Memory mapped I/O. Interrupt driven I/O. Direct memory access. Redundant arrays of inexpensive disks; Introduction to Multi-core architecture, Multi-processors. Clusters.

Suggested Readings:

1. David A. Patterson and John L. Hennessy, Computer Organization and Design: The Hardware and Software Interface, Morgan Kaufmann Publishers, 4th Edition.(2009).
2. John L. Hennessy and David A. Patterson, Computer Architecture: A Quantitative Approach, Morgan Kaufmann Publishers (2007).

IMAGE PROCESSING

PE 623 CS
Instruction: 3L periods per week
CIE:30marks
Credits: 3

Duration of SEE: 3 hours
SEE: 70marks

Prerequisite:
1. Data Structures
2. Discrete Mathematics

Objectives:

1. To introduce basics of visual perception, sampling, quantization and representation of digital images
2. To introduce spatial domain and frequency domain filtering techniques necessary for image processing operations.
3. To learn advanced image analysis techniques such as image restoration, image compression, image segmentation
4. To learn techniques of multi resolution methods, wavelets and morphological processing.
5. To understand the applications of image processing.

Outcomes:

1. Understand the basic image enhancement techniques in spatial & frequency domains.
2. Understand the basics of multi-resolution techniques.
3. Understand the basics of segmentation methods.
4. Apply this concept for image handling in various fields.
5. Knowledge about Morphological operations.

UNIT-I

Fundamentals of Image Processing: Introduction, examples, fundamental steps, components, elements of visual perception, light and electromagnetic spectrum, image sensing and acquisition, image sampling and quantization, basic relationships between pixels. **Intensity Transformations And Spatial Filtering**: Background, some basic intensity transformation functions, histogram processing,fundamentalsofspatialfiltering,smoothing
spatial filters, sharpening spatial filters, combining spatial enhancement methods.

UNIT-II

Filtering In The Frequency Domain: Background, preliminary concepts, sampling and Fourier transform of sampled functions, discrete Fourier transform (DFT) of one variable, extension to functions of two variables, some properties of the 2-D discrete Fourier transform,basicsoffilteringinthefrequencydomain,imagesmoothing,imagesharpening,
homo- morphic filtering.

UNIT –III

Image Restoration: Noise models, restoration in the presence of noise only-spatial filtering, periodic noise reduction by frequency domain filtering, linear degradation, position-invariant degradation, estimating the degradation function, inverse filtering, minimum mean square error filtering, constrained least squares filtering, geometric mean filter.

UNIT – IV

Wavelets And Multi Resolution Processing: Background, multi-resolution expansions, wavelet transforms in one dimension, the fast wavelet transform, wavelet transforms in two dimensions, wavelet packets.
Image Compression: Fundamentals, image compression models, elements of information theory, error free compression, lossy compression, image compressionstandards.

UNIT-V

Image Segmentation: Fundamentals, point, line and edge detection, thresholding, region-based segmentation, segmentation using morphological watersheds, the use of motion in segmentation.

Morphological Image Processing: Preliminaries, erosion and dilation, opening and closing, the Hit-or-Miss transformation, some basic morphological algorithms, some basic gray-scale morphological algorithms.

Suggested Readings:

1. Rafael C. Gonzalez and Richard E. Woods, *Digital Image Processing*, PHI Learning Pvt. Limited, 3rd Edition, 2008.
2. Rafael C.Gonzalez, Richard E.Woods and Steven L.Eddins, *Digital Image Processing Using MATLAB*, 2nd Edition, McGraw Hill, 2010.
3. AL. Bovik, *The Essential Guide to Image processing,* 2nd Edition, Elsevier, 2009.
4. Anil K.Jain, "Fundamentals of Digital Image Processing", PHI, 2006.
5. William K. Pratt, *Digital Image Processing*, John Wiley & Sons, Inc., 3rd Edition, 2001

SOFTWARE TESTING

PE624CS

Instruction: 3 periods per week Duration of SEE: 3 hours

CIE: 30marks SEE: 70marks

Credits: 3

Objectives:

1. To understand and Learn the basic concepts ofTesting.
2. To follow the methodology of White BoxTesting.
3. To Obtain knowledge of Integration and SystemTestings
4. To understand the concepts of Object OrientedTesting.

Outcomes:

Students must be able to:

1. Gain the basic knowledge of Testing.
2. Acquire the knowledge of White Box Testing methods
3. Test an application using Functional Testing.
4. Use Object Oriented Testing and Millennium Testing methods
5. Implement Integration and system testings.

UNIT – I

Basic definitions, Test cases, Insights from a Venn diagram, Identifying test cases, Error and fault taxonomies, Levels of testing. Examples: Generalized pseudo code, The triangle problem, The NextDate function, The commission problem, The SATM (Simple Automatic Teller Machine) problem, The currency converter, Saturn windshield wiper.

UNIT – II

Water fall model– V–model– Spiral model– Agile model ,Life cycle of testing, Static Testing dynamic testing White box testing ,Block box testing , Regression testing, Integration Testing ,System and Performance Testing – UsabilityTesting.

UNIT – III

Boundary Value Analysis, Equivalence Class Testing, Decision Table Based Testing, Cause Effect Graphing Technique, Path testing –Cyclomatic Complexity, Graph Metrics, Data Flow Testing, Slice based testing.

UNIT – IV

Test planning, cost–benefit analysis of testing, monitoring and control, test reporting, test control Specialized testing, Object Oriented Testing, Automated Tools for Testing, Tool Selection and Implementation, Challenges in test automation, GUITesting.

UNIT – V

Software Complexity, Exploratory Testing, Test-Driven Development, All Pairs Testing, Software Technical Reviews, Software Testing Excellence-Best practices.

Suggested Readings:

1. Paul C. Jorgensen, "Software Testing: A Craftsman's Approach", 4th Edition, CRC Press,2007
2. Boris Biezer,"Software Testing Techniques",2nd Edition,Dreamtech Press, 2003
3. Aditya P.Mathur,"Foundations of Software Testing", Pearson Education, 2013.
4. William E. Perry, "Effective Methods for Software Testing", 2nd Edition,Wiley.
5. Srinivasan Desikan, Gopalaswamy Ramesh, "Software Testing, Principles and Practices", 2006. PearsonEducation.

DATA MINING

PE625 CS

Instruction: 3 periods per week *Duration of SEE: 3 hours*

CIE:30marks *SEE: 70marks*

Credits: 3

Objectives:

1. To introduce the basic concepts of data Mining and its applications
2. Tounderstanddifferentdataminginglikeclassification,clusteringandFrequentPattern mining
3. To introduce current trends in data mining

Outcomes:

Student will be able to

1. Organize and Prepare the data needed for data mining using pre preprocessing techniques
2. Implement the appropriate data mining methods like classification, clustering or Frequent Pattern mining on a given data set
3. Explain the steps in KDD, Identify various pre-processing techniques and compute similarity among data objects
4. Compute associations and correlations among items by mining frequent patterns from transactional databases
5. Construct Multidimensional data models to represent data cubes and perform characterization & generalization tasks on data cubes.

UNIT – I

Introduction: introduction to Mining and Data Mining, What kinds of data can be mined? What kinds of patterns can be mined? Which technologies are used? Which kinds of applications are Targeted? Major issues in Data Mining. Getting to know your data: Data objects and attributed types. Basic statistical descriptions of data. Data visualization, Measuring data similarity and dissimilarity.

UNIT – II

Mining frequent patterns, Associations and correlations: Basic concepts and methods, Frequenct Item set Mining Methods, Pattern evaluation methods.

UNIT – III

Classification: Basic concepts, Decision tree induction, Bayes classification methods, Advance methods, Bayesian Belief Network, Classification by back propagation, Support vector machine.

UNIT – IV

Cluster Analysis: Concepts and Methods: Cluster Analysis, Partitioning Methods, Hierarchical Methods, Density-Based Methods, Grid-Based Methods, Evaluation of clustering.

UNIT – V

Data Mining Trends and Research Frontiers: Mining Complex Data Types, Other Methodologies of Data Mining, Data Mining Applications, Data Mining and Society, Data Mining trends.

Suggested Readings:

1. Jiawei Han, Micheline Kamber, Jin Pei, Data Mining: Concepts & Techniques, 3rd Edition., Morgon Koffman ,2011
2. Vikram Pudi, P. Radha Krishna,***DataMining***, Oxford University Press, 1st Edition, 2009
3. Ning Tan, Michael Steinbach, Vipin Kumar, 2008
Introduction to Data Mining,*Pearson Education.*

MOBILE COMPUTING

PE626 CS

Instruction: 3 periods per week *Duration of SEE: 3 hours*

CIE:30marks *SEE: 70marks*

Credits: 3

Objectives:

1.To introduce basics of wireless voice and data communication technologies
2.To build working knowledge on various telephone and satellite networks
3.To study the working principles of wireless LANs and standards
4.To study principles of adhoc networks and routing
5.To gain knowledge on integration of mobile networks into Internet
6.To build skills in working with wireless application protocols to develop mobile applications.

Outcomes:

1.Students must be able to:
2.Understand and apply various techniques involved in planning and construction stages.
3.Implement Adhoc Network Routing protocols.
4.Mini based project based on tracking, localization and routing in wireless networks.
5.Implement file transfer, access and authentication based applications for mobile computing.

UNIT-I

Introduction – Wireless transmission – Frequencies for radio transmission – Signals – Antennas – Multiplexing – Modulations – Spread spectrum, Cellular Wireless Networks,4G -Introduction, features and challenges, Applications of 4G, 4G Network architecture.

UNIT-II

Telecommunication systems – GSM – GPRS – DECT – UMTS – IMT-2000 – Satellite Networks - Basics – Parameters and Configurations – Capacity Allocation – FAMA and DAMA – Broadcast Systems – DAB – DVB.

UNIT-III

Wireless LAN – IEEE 802.11 - Architecture – services – MAC – Physical layer – IEEE 802.11a - 802.11b standards – HIPERLAN – Blue Tooth.

UNIT-IV

Routing Ad-hoc Network Routing Protocols- Ad-hoc Network Routing Protocols, Destination Sequenced Distance Vector Algorithm, Cluster Based Gateway Switch Routing, Global State Routing, Dynamic Source Routing, Ad-hoc on-demand Routing.
Mobile IP - Dynamic Host Configuration Protocol.
Traditional TCP - Classical TCP Improvements – WAP, WAP 2.0., Wireless TCP.

UNIT-V

Data Dissemination: Pull and Push Based Data Delivery models, Data Dissemination by Broadcast, Broadcast Disks, Directory Service in Air, Energy Efficient Indexing scheme for Push Based Data Delivery.

Suggested Readings:

1. Jochen Schiller, Mobile Communications, Pearson Education, 2nd Edition, 2009.
2. William Stallings,"Wireless Communications and Networks", PHI/Pearson Education, 2002.
3. KavehPahlavan, Prasanth Krishnamurthy, "Principles of Wireless Networks", Prentice Hall, 2003.
4. UweHansmann, LotharMerk, Martin S. Nicklons and Thomas Stober, "Principles of Mobile Computing", Springer, 2003. 5. Krzysztof Wesolowski, Mobile Communication Systems, John Wiley and Sons Ltd, 2002.
5. Kurnkum Garg, Mobile Computing, Pearson Education, 2010
6. Asoke K Talukder, Roopa R Yavagal, Mobile Computing, TMH 2008.
7. Raj Kamal, Mobile Computing, Oxford, 2009.
8. "A Survey of Mobile Transactions appeared in Distributed and Parallel databases" 16,193-230, 2004, Kluwer Academics Publishers.
9. S. Acharya, M. Franklin and S. Zdonil, "Balancing Push and Pull for Data Broadcast, Proceedings of the ACM SIGMOD", Tuscon, AZ, May 1997
10. S. Acharya, R. Alonso, M. Franklin and S. Zdonik, "Broadcast Disks: Data Management for Asymmetric Communication Environments, Proceedings of the ACM SIGMOD Conference", San Jose, CA, May 1995.

MACHINE LEARNING LAB

PC 651 CS
Instruction: 2 periodsperweek
CIE:25marks
Credits: 1

Duration of SEE: 3 hours
SEE: 50marks

Objectives:

1.Demonstration of different classifiers on different data.
2.Demonstrate ensembling of classifiers for solving real world problems
3. Make use of realworld data to implement machine learning models.

Outcomes:

1.Students must be able to:
2. Apply machine learning algorithms: dataset preparation, model selection, model building etc.

3. Evaluate various Machine Learning approaches.
4. Use scikit-learn, Keras and Tensorflow to apply ML techniques.

5. Apply unsupervised learning and interpret the results.

LIST OF EXPERIMENTS:

1. **Basic Data Preprocessing**
   a. Installation of python environment/Anaconda IDE for machine learning: installing python modules/Packages like scikit-learn, Keras and Tensorflow etc.
   b. Programs involving pandas, Numpy and Scipy libraries.

2.Programs for classification

1. Build models using linear regression and logistic regression and apply it to classify a new instance
2. Write a program to demonstrate the following classifiers. Use an appropriate data set for building the model. Apply the model to classify a new instance.
   a) Decision tree
   b) K nearest neighbour
   c) Naïve bayes
   d) Support vector machine

3.Demonstration of Clustering algorithms using

1. k-means
2.Hierarchical algorithms(agglomerative etc).
Interpret the clusters obtained.

4.Demonstrate ensemble techniques like boosting, bagging, random forests etc.

5.Build a classifier, compare its performance with an ensemble technique like random forest.

6.Evaluate various classification algorithms performance on a dataset using various measures like True Positive rate, False positive rate, precision, recall etc.

7.Demonstrate GA for optimization (minimization or maximization problem).

8.Case study on supervised/unsupervised learning algorithm:

a) Handwritten digits classification using CNN.

b) Text classification using python libraries.

DESIGN AND ANALYSIS OF ALGORITHMS LAB

PC 652 CS

Instruction: 2 periods per week *Duration of SEE: 3 hours*
CIE:25marks *SEE: 50marks*
Credits: 1

Prerequisite:

1. Problem Solving Skills
2. Data Structures
3. Discrete Structures

Objectives:

1. To learn the importance of designing an algorithm in an effective way by considering space and time complexity
2. To learn graph search algorithms.
3. To study network flow and linear programming problems
4. To learn the dynamic programming design techniques.
5. To develop recursive backtracking algorithms.

Outcomes:

After completing this course, the student will be able to:

1. Design an algorithm in a effective manner
2. Apply iterative and recursive algorithms.
3. Design iterative and recursive algorithms.
4. Implement optimization algorithms for specific applications.
5. Design optimization algorithms for specific applications.

List of Experiments:

SNo.	**Description of the program**
1	Print all the nodes reachable from a given starting node in a digraph using BFSmethod and Check whether a given graph is connected or not using DFS method.
2	Sort a given set of elements anddeterminethetimerequiredtosorttheelements using following algorithms: • Merge Sort • Quick Sort
3	Implement Knapsack problem using • Brute Force Approach • Greedy Method • Dynamic Programming
4	FindMinimumCostSpanningTreeofagivenundirectedgraph using • Kruskal's algorithm • Prim’s algorithm
5	From a given vertex in a weighted connected graph, find shortest paths to other vertices using Dijkstra's algorithm
6	Implement Travelling Salesperson Problem using • Brute Force Approach • Dynamic Programming
7	ImplementAll-PairsShortestPathsProblemusingFloyd'salgorithm
8	Implementthe following using BackTracking • NQueen'sproblem • Hamiltonian Cycle • Graph Coloring

MINI PROJECT

PC533CS
Instruction: 4 periods per week Duration of SEE: 3
hours CIE: 25 marks EE: 50 marks
Credits: 2
Objectives:
1. To enhance practical and professional skills.
2. To familiarize tools and techniques of systematic literature survey and documentation.
3. To expose the students to industry practices and teamwork.
4. To encourage students to work with innovative and entrepreneurial ideas.

Outcomes:
Student will be able to
1. Demonstrate the ability to synthesize and apply the knowledge and skills acquired in the academic program to the real-world problems.
2. Evaluate different solutions based on economic and technical feasibility.
3. Effectively plan a project and confidently perform all aspects of project management.
4. Develop and test the solution.
5. Demonstrate effective coding, written, presentation and oral communication skills.

I. CASE STUDY

The students are required to carry out mini projects in any of the areas such as Data Structures, Microprocessors and Interfacing, Database Management Systems, Operating Systems, Design and Analysis of Algorithms, Software Engineering, Data Communications, Web Programming& Services, Computer Networks, Compiler Construction, and Object- Oriented System Development.
Problems Statements are suggested to be taken from Smart India Hackathon (SIH) Portalinvited from the Ministries / PSUs / MNCs / NGOs to be worked out through.

The project could be classified as hardware, software, modeling, simulation etc. The projectshould involve one or many elements of techniques such as analysis, design, and synthesis.
The department will appoint a project coordinator who will coordinate the following:
1. Grouping of students (maximum of 3 students in a group)
2. Allotment of projects and project guides.
3. All projects allotment is to be completed by the 4th week of the semester so that the students get sufficient time for completion of the project.
4. Disseminate guidelines given by monitoring committee comprising of senior faculty members to the students and their guides.

Sessional marks are to be awarded by the monitoring committee.Common norms will be established for the final presentation and documentation of the project report by the respective departments. Students are required to submit a presentation and report on the mini project at the end of the semester.

OPEN ELECTIVES - I

OOP USING JAVA

OE 601 CS

Instruction: 3 periodsperweek *Duration of SEE: 3hours*
CIE:30marks *SEE: 70marks*
Credits: 3

Course Objective:

To understand fundamentals of object-oriented programming in Java which includes defining classes, invoking methods, difference between applet and application programs, using class libraries

Course Outcomes:

1. Define, understand, differentiate the Object Oriented concepts and Java Programming concepts
2. Use Exception handling and multithreading mechanisms to create efficient software applications.
3. Utilize modern tools and collection framework to create Java applications to solve real world problems.
4. Design and develop GUI based applications using awt for system based applications.
5. Apply object oriented concepts on real time scenarios.

UNIT-I

Object Oriented System Development: Understanding Object Oriented Development, Understanding Object Concepts, Benefits of Object Oriented Development.
Java Programming Fundamentals: Introduction. Overview of Java, Data Type, Variables and Arrays, Operators, Control statements, Classes, Methods , Inheritance, Packages and Interfaces.

UNIT-II

Exceptions Handling, Multithreaded Programming, I/O basics, Reading Console input and output, Reading and Writing Files, Print Writer Class, String Handling.

UNIT-III

Exploring Java Language, Collections Overview, Collections Interfaces, Collections Classes, Iterators, Random Access Interface, Maps, Comparators, Arrays, Legacy classes and interfaces, Sting tokenizer, BitSet, Date, Calendar, Timer.

UNIT-IV

Introducing AWT working With Graphics: AWT Classes, Working with Graphics.
Event Handling: Two Event Handling Mechanisms, The Delegation Event Model, Event Classes, Source of Events, Event Listener Interfaces. and AWT Controls

UNIT-V

Java I/O classes and interfaces, Files, Stream and Byte classes, Character Streams, Serialization.

Text Book

1. Herbert Schildt, The Complete Reference Java, 7th Edition, Tata McGraw Hill, 2005.

Suggested Reading:

1. James M Slack, Programming and Problem solving with JAVA, Thomson Learning, 2002
2. C Thomas Wu, An Introduction to Object Oriented programming with Java, Tata McGraw Hill, 2005.

DATA STRUCTURES AND ALGORITHMS

OE 602 CS

Instruction: 3 periodsper week *Duration of SEE: 3hours*
CIE:30marks *SEE: 70marks*
Credits: 3

Course Objectives:

1. To develop proficiency in the specification, representation, and implementation of abstract data types and data structures.
2. To discuss the linear and non-linear data structures and their applications
3. To introduce the creation, insertion and deletion operations on binary search trees and balanced binary search trees.
4. To introduce various internal sorting, searching techniques and their time complexities

Course Outcomes:

After completing this course, the student will be able to:

1. Understand the importance of abstract data type and implementing the concepts of data structure using abstract data type.
2. Evaluate an algorithm by using algorithmic performance and measures.
3. Distinguish between linear and non-linear data structures and their representations in the memory using array and linked list.
4. Apply the suitable data structure for a real world problem and think critically for improvement in solutions.
5. Determine the suitability of the standard algorithms: Searching, Sorting and Traversals

UNIT – I

Algorithms: Introduction, Algorithm Specifications, Recursive Algorithms, Performance Analysis of an algorithm- Time and Space Complexity, Asymptotic Notations.
Arrays: Arrays - ADT, Polynomials, Sparse matrices, Strings-ADT, Pattern Matching.

UNIT – II

Stacks and Queues: Stacks, Stacks using Arrays, Stacks using dynamic arrays, Evaluation of Expressions – Evaluating Postfix Expression, Infix to Postfix.
Queues: Queues ADT, operations, Circular Queues, Applications

UNIT – III

Linked Lists: Singly Linked Lists and Chains, Linked Stacks and Queues, Polynomials, Operations for Circularly linked lists, Equivalence Classes, Sparse matrices, Doubly Linked Lists.
Hashing: Static Hashing, Hash Tables, Hash Functions, Overflow Handling, Theoretical Evaluation of Overflow Techniques

UNIT – IV

Trees: Introduction, Binary Trees, Binary Tree Traversals, Heaps, Binary Search trees (BST) : Definition, Searching an element, Insertion into a BST, Deletion from a BST.
Efficient Binary Search Trees: AVL Trees: Definition, Searching an element, Insertion into a AVL

UNIT – V

Graphs: Graph Abstract Data Type, Elementary Graph operations (DFS and BFS), Minimum Cost Spanning Trees (Prim‘s and Kruskal‘s Algorithms).
Sorting and Searching: Insertion sort, Quick sort, Best computing time for Sorting,
Merge sort, Heap sort, shell sort, Sorting on Several Keys, List and Table Sorts, Summary of Internal Sorting, Linear and Binary Search algorithms.

Text Book:

1. Horowitz E, Sahni S and Susan Anderson-Freed, Fundamentals of Data structures in C, 2nd Edition (2008), Universities Press

Reference Books:

1. Mark A Weiss, Data Structures and Algorithm Analysis In C, Second Edition (2002), Pearson
2. Kushwaha D. S and Misra A.K, Data structures A Programming Approach with C, Second Edition (2014), PHl.
3. Gilberg R. F and Forouzan B. A, Data structures: A Pseudocode Approach with C, Second Edition (2007), Cengage Leaming
4. Tanenbaum A. M , Langsam Y. Augenstein M. J, Data Structures using C, Second Edition (2008), Pearson.
5. Thomas H. Cormen, Charles E. Leiserson, Ronald L Rivest, Clifford Stein, Introduction to Algorithms, Third Edition (2009), MIT Press
6. Yedidyah Langsam , Moshe J. Augenstein ,Aaron M. Tenenbaum, Data
7. Structures Using C and C++ , Second Edition (2009), PHI