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 Hours/W	CIE	SEE	Duration in Hrs.	
Theory Courses										
1	PC501CS	Software Engineering	3	-	-	3	30	70	3	3
2	PC502CS	Principles of Programming Languages	3	-	-	3	30	70	3	3
3	PC503CS	Compiler Design	3	-	-	3	30	70	3	3
4	PC504CS	Essentials of Artificial Intelligence	3	-	-	3	30	70	3	3
5	PC505CS	Computer Networks	3	-	-	3	30	70	3	3
6	PE	Professional Elective-I	3	-	-	3	30	70	3	3
7	PC506CS	Design and Analysis of Algorithms	3	-	-	3	30	70	3	3
Practical / Laboratory Courses										
8	PC551CS	Computer Networks Lab	-	-	2	2	25	50	3	1
9	PC553CS	Artificial Intelligence Lab	-	-	2	2	25	50	3	1
10	PC552CS	Design and Analysis of Algorithms Lab	-	-	2	2	25	50	3	1
TOTAL			**21**	**-**	**06**	**27**	**285**	**640**	**30**	**24**

Professional Elective -I	
Course Code	**Course Title**
PE511CS	Object Oriented Analysis and Design
PE512DS	Data Science
PE513CS	Queueing Theory and Modelling
PE514CS	Information Theory and Coding
PE515CS	Computer Graphics
PE516CS	Software Reuse Techniques

SCHEME OF INSTRUCTION & EXAMINATION

B.E (Computer Science and Engineering)

SEMESTER-VI

S. No.	Course Code	Course Title	Scheme of Instruction				Scheme of Examination			Credits
			L	T	D/P	Contact Hours/WK	CIE	SEE	Duration in Hrs.	
Theory Courses										
1	PC601CS	Quantum Computing	3	-	-	3	30	70	3	3
2	PC602CS	Nano Technology and Computing	3	-	-	3	30	70	3	3
3	PC603CS	Machine Learning & Deep Learning	3	-	-	3	30	70	3	3
4	PC604CS	Cryptography and Network Security	3	-	-	3	30	70	3	3
5	PE	Professional Elective-II	2	-	-	2	30	70	3	2
6	PC605CS	Computer Ethics	3			3	30	70	3	3
Practical / Laboratory Courses										
7	PC651CS	Quantum Computing Lab	-	-	2	2	25	50	3	1
8	PC652CS	Machine Learning & Deep Learning Lab	-	-	2	2	25	50	3	1
9	PW653CS	Mini Project	-	-	4	4	25	50	3	2
10	SI671CS	Summer Internship*	-	-	-	-	-	-	-	-
TOTAL			**17**	**-**	**08**	**25**	**255**	**570**	**27**	**21**

Professional Elective -II	
Course Code	**Course Title**
PC601 IT/PE	Embedded Systems
PE622CS	Design Thinking
PE623CS	Software Testing
PE624CS	Advanced Computer Architecture
PE625CS	Image Processing
PE511DS	Information Retrieval System
Open Elective - I	
Course Code	**Course Title**
OE601CS	OOP using JAVA (Not for CSE & IT Students)
OE602CS	Data Structure and Algo. (Not for CSE & IT Students)

SOFTWARE ENGINEERING

PC501CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30marks SEE: 70marks
Credits: 3	

Objectives:
1. To introduce the fundamental concepts of software engineering process frameworks, process models, and agile methodologies to select appropriate models for given project scenarios.
2. To illustrate software engineering principles and systematic requirements engineering techniques to gather, analyze, and validate user requirements effectively.
3. To demonstrate analysis and design modelling approaches including data, object-oriented, scenario-based, and behavioural models to transform requirements into design.
4. To describe architectural styles, component-level design, and user interface design principles to develop modular, maintainable, and user-centric software systems.
5. To impart testing strategies, debugging techniques, software metrics, and quality assurance standards like ISO 9000 to ensure delivery of reliable software products.
Outcomes:
Student will be able to
1. Acquire working knowledge of alternative approaches and techniques for each phase of software development
2. Judge an appropriate process model(s) assessing software project attributes and analyze necessary requirements for project development eventually composing SRS
3. 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
4. Concede product quality through testing techniques employing appropriate metrics by understanding the practical challenges associated with the development of a significant software system
5. Apply software engineering principles of agility, user interface design, and deployment to develop user-centric software solutions while adapting to changing requirements and ensuring effective product delivery.

UNIT-1
Introduction to Software Engineering: A generic view of Process: Software Engineering, Process Framework, CMM Process Patterns, Process Assessment. **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 and Process. **An Agile view of Process:** Introduction to Agility and Agile Process, Agile Process Models.
UNIT-2
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, Requirements Engineering Tasks, Initiating Requirements Engineering Process, Eliciting Requirements,

Developing Use- Cases, Building the Analysis Model, Negotiating Requirements, Validating Requirements.
UNIT-3
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-4
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-5
Testing: Strategies: A Strategic Approach to Conventional Software Testing, Test Strategies for O-O Software. **Tactics:** Software Testing Fundamentals, Black-box and White-box Testing, Basis Path Testing, Control Structure Testing, O-O Testing Methods. **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. **DevOps: Introduction to DevOps, what is DevOps, Importance and benefits of DevOps, Principles** **and practices, DevOps lifecycle for business Agility and continuous testing.**
Suggested Readings:
1. Roger S. Pressman, Software Engineering: A Practitioner's Approach, 9th Edition, McGraw Hill, 2020
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
4. Deepak Gaikwad, Viral Thakkar, "DevOps Tools from Practitioner's viewpoint", Wiley Publications, 2019.

PRINCIPLES OF PROGRAMMING LANGUAGES

PC502CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30marks SEE: 70marks
Credits: 3	

Objectives :

1. Describe the fundamental concepts, paradigms, syntax and semantics of programming languages.
2. Analyze language design principles, language evaluation criteria, and implementation techniques.
3. Explain concepts of data types, bindings, scope, type checking, expression, and control structures in programming language design.
4. Examine object-oriented, concurrent, logic, and functional programming languages.
5. Compare programming language features and select appropriate paradigms and languages for solving real-world computing problems.

Outcomes:

Student will be able to:

1. Explain programming language concepts, paradigms, syntax representation methods and semantic models.
2. Analyze data types, bindings, type systems, expressions, and control structures used in programming languages.
3. Apply concepts of subprograms, parameter passing mechanisms, scope rules, and abstraction techniques in program design.
4. Demonstrate object-oriented, concurrent and logic programming concepts using appropriate languages and tools.
5. Evaluate and compare functional and scripting language solutions using languages such as, F#, Python, Clojure & Scala and understand their suitability for different applications.

UNIT I

Preliminary Concepts: Reasons for studying, concepts of Programming languages, Programming domains, Language Evaluation Criteria. Influence on Language design, Language categories,Programming Paradigms, 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.

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. Expression and Statements: Arithmetic relational and Boolean expressions, Short circuit evaluation mixed mode assignment, Assignment Statements, Control Structures, Selection, Iteration.

UNIT III

Subprograms: Fundamental of Subprograms, Design issues of subprogram and operations, local referencing environments, parameter passing methods, overloaded subprograms, Generic subprograms, Design issues for functions, user defined overloaded operators, Coroutines.

UNIT IV

Abstract types: Introduction to Data Abstraction and encapsulation, design issues for Abstract Data types (ADT), language examples, parameterized Abstract data types ADT, object oriented programming inC++, Java, C#, Kotlin &Rust. Concurrency: Introduction to subprogram level concurrence, Semaphores, monitors, message passing, Java threads, C# threads. Exception handling in C++, C#, Python, Java&Rust.

Logic Programming language: Introduction and overview of logic programing, basic elements of prolog, application of logic programming.

UNIT V

Functional Programming Languages: Introduction, fundamentals of FPL, F#, Scala, **Clojure**,and Python functional programming. 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.

Suggested Readings:

1) Concepts of Programming languages Robert W. Sebesta 8/e, Pearson Education, 2008 / 11th Edition, 2020
2) Programming Language Design Concepts, D. A. Watt, Wiley Dreamtech, 2007 / Reprint 2015

References:

1) Programming languages 2nd Edition, A. B. Tucker, R. E. Noonan, TMH
2) Programming Kotlin Venkat Subramaniam, Pragprog publications, 2019
3) The Rust Programming language Steve Klabnik, Carol Nichols. Nostarch Press, 2019
4) Functional Python Programming (PythonFP) Steven F. Lott Packtpub publication 3rd Edition, 2024
5) Programming Scala Martin odersky, Lex Spoon, Bill Venners, Oreilly publisher, 3rd Edition 2020
6) **Programming Clojure, Alex Miller, Stuart Holloway, Aaron Bedra, pragprog publication, 3rd Edition, 2018**

COMPILER DESIGN

PC503CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30marks SEE: 70marks
Credits: 3	

Objectives:
1. To understand and list the different stages in the process of compilation and identify different methods of lexical analysis
2.Design top-down and bottom-up parsers
3. Identify synthesized and inherited attributes
4. Develop syntax directed translation schemes
5. Develop algorithms to generate code for a target machine
Outcomes:
Upon completion of the course, the students will be able to:
1. Develop the lexical analyzer, for a given grammar specification.
2. Design top-down and bottom-up parsers, for a given parser specification.
3. Develop syntax directed translation schemes.
4. Generate intermediate code for programming constructs
5. Develop algorithms to generate code for target machine.

UNIT-1
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-2
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 Generator Yaac.
UNIT-3
Syntax-Directed Translation: Syntax-Directed Definitions, Evaluation Orders for SDD's Applications of Syntax-Directed Translation. **Symbol Table:** Structure, Operations, Implementation and Management.
UNIT-4
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-5
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, Introductionto 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.

ESSENTIALS OF ARTIFICIAL INTELLIGENCE

PC504CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	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 and illustrate basic principles of AI in solutions that require problem solving, search, Inference
2. Represent natural language/English using Predicate Logic to build knowledge through various representation mechanisms.
3. Demonstrate understanding of steps involved in building of intelligent agents, expert systems, Bayesian networks.
4. Differentiate between learning paradigms to be applied for an application.
5. Compare and contrast pros and cons of various machine learning techniques and to get an insight when to apply particular machine learning approach.

.

UNIT-1
Problem Solving & Search: Introduction- introduction to intelligence Foundations of artificial intelligence (AI). History of AI, Structure of Agents. **Problem Solving -** Formulating problems, problem types, states and operators, state space. **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-2
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-3
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-4
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-5
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.
Suggested Readings:
1.Stuart Russell and Peter Norvig. Artificial Intelligence – A Modern Approach, 3rd Edition, Pearson Education Press, 2009.1.
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.
4. Machine Learning: An Algorithmic Perspective, Stephen Marsland, Second Edition (Chapman & Hall/Crc Machine Learning & Pattern Recognition) (2014)
5. Machine Learning, Tom Mitchell, McGraw-Hill Science/Engineering/Math; (1997).

COMPUTER NETWORKS

PC505CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30marks SEE: 70marks
Credits: 3	

Objectives:
1.Toexplain the basics of communication model and its standards and modern Network architectures from a design and performance perspective.
2. To discuss Data Transmission standards and MAC protocols.
3. To introduce network layer and its functions, IP layer concepts, and IP protocols, Routing techniques.
4. To familiarize with functions of transport layer, congestion control and QoS techniques.
5. To introduce different application layer protocols and the basic concepts of Cryptography.
Outcomes:
Students will be able to:
1. Identify Data Communication components, explain the functions of the different layer of the OSI and TCP/IP Protocol.
2. Explain Data Transmission standards and MAC protocols.
3. Differentiate between types of networks, explain different functions of network layer.
4. Describe the functions of transport layer, TCP and congestion and quality of service aspects.
5. Understand different application layer protocol and describe about the different encryption Techniques.

UNIT-1
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-2
Data Link Layer and Medium Access Sub Layer: **Flow Control and Error control protocols:** Stop Error Detection and Error Correction - Fundamentals, Block coding, Hamming Distance, CRC and Wait, Go back – N ARQ, Selective Repeat ARQ, Sliding Window, Piggybacking. Multiple access protocols: Pure ALOHA, Slotted ALOHA, CSMA/CD,CDMA/CA

UNIT-3
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-4
Transport Layer: Process to Process Communication, Elements of transport protocol **Internet Transport Protocols:** UDP, TCP. **Congestion and Quality of Service:** QoS improving techniques.
UNIT-5
Application Layer: Domain Name Space (DNS), EMAIL, SNMP, Bluetooth **Basic concepts of Cryptography:** Network Security Attacks, Firewalls, Symmetric Encryption, Data encryption Standards, public key Encryption (RSA), Hash Functions, Message Authentication, 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 Prentice Hall India.
3. W. Richard Stevens, Unix Network Programming, Prentice Hall / Pearson Education, 2009

DESIGN AND ANALYSIS OF ALGORTIHMS

PC506CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30marks SEE: 70marks
Credits: 3	

Objectives:
1. To review elementary data structures, order notation and algorithm analysis
2. To learn algorithm design strategies such as Divide-and-Conquer
3. To learn algorithm design strategies such as greedy method and dynamic programming
4. To learn algorithm design strategies such as back tracking and branch & bound technique
5. To understand the concepts of NP-hard and NP-complete
Outcomes:
Students will be able to:
1. Analyze the time and space complexity of algorithms using asymptotic notations and recurrence relations.
2. Design algorithms for various computing problems using Divide-and-Conquer technique.
3. Critically analyze the different algorithm design techniques for optimization problems.
4. Modify existing algorithms to improve efficiency using state-space search techniques.
5. Design algorithms for various computing problems and classify them based on computational complexity classes.

UNIT-1
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 data structures–Graphs:** BFS, DFS, Bi-Connected Components. Sets: representation, UNION, FIND operations.
UNIT-2
Divide-and-Conquer Method: The general method, Binary search, Finding maximum and minimum, Merge sort, Quick sort. **Brute Force:** Knapsack, Traveling salesman problem, Convex-Hull
UNIT-3
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-4
Back tracking: N-queens problem, Graph coloring, Hamiltonian cycles **Branch-and-bound:** FIFO & LC branch and Bound methods, 0/1 Knapsack problem, Traveling salesperson

UNIT-5
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:** Introduction, models for parallel computing
Suggested Readings:
1.Horowitz E, Sahni S, Fundamentals of Computer Algorithms, 2ndEdition, Universities Press, 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.Michael T. Goodrich, Roberto Tamassia, Algorithm Design: Foundations, Analysis and InternetExamples, John Wiley & Sons,2002

PROFESSIONAL ELECTIVE – I

OBJECT ORIENTED ANALYSIS AND DESIGN

PE511CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30marks SEE: 70marks
Credits: 3	

Objectives:
1. To understand OOAD concepts, UML and Agile/UP processes.
2. To learn elaboration artifacts and domain modeling.
3. To understand UML diagrams and design to code mapping.
4. To learn GRASP, GoF patterns and architectural design.
5. To understand TDD, architecture documentation and case studies.
Outcomes:
Students will be able to:
1. Create use-cases, activity and sequence diagrams for requirements.
2. Develop domain models and class diagrams for OO design.
3. Apply UML interaction/state diagrams and map design to code.
4. Apply GRASP and design patterns for solving design problems.
5. Document architecture and execute OOAD case study using Agile/UP.

UNIT-1
Object-Oriented Analysis and Design: Introduction, UML **Iterative, Evolutionary, and Agile:** Introduction, Unified Process, Agile Modeling, Agile Unified Process. Case Studies: Basics **Inception& Use cases:** Introduction, Evolutionary Requirements, Use Cases, Usecase Diagrams, Activity Diagrams.
UNIT-2
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-3
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-4
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-5
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. MeilirPage-Jones(2000),Fundamentals of Object Oriented Designin 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.

PROFESSIONAL ELECTIVE – I

DATA SCIENCE

PE512DS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30marks SEE: 70marks
Credits: 3	

Objectives:
1. To understand mathematical foundations and linear algebra for data science.
2. To learn statistical modeling and inference techniques.
3. To understand predictive modeling using regression techniques.
4. To learn R programming for data manipulation and analysis.
5. To understand classification, clustering and real-world data science applications.
Outcomes:
Students will be able to:
1. Apply linear algebra and dimensionality reduction for data preprocessing.
2. Perform statistical hypothesis testing and Bayesian inference on datasets.
3. Build linear and logistic regression models for prediction.
4. Write R programs for data analysis using data frames and functions.
5. Implement KNN, K-Means, Logistic Regression in R and apply to case studies.

UNIT-1
Data Science: Introduction to data science, Linear Algebra for data science, Linear Equation, Distance, Eigen values, Eigenvectors, Dimensionality reduction.
UNIT-2
Statistical Modeling, Random variables, sample statistics, Inference: Statistical Hypothesis Testing, Confidence Intervals, P hacking, Bayesian Inference.
UNIT-3
Predictive Modeling: Linear Regression, Simple Linear Regression model building, Multiple Linear Regression, Logistic regression.
UNIT-4
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 andSimulation in R.
UNIT-5
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.

PROFESSIONAL ELECTIVE – I

QUEUEING THEORY AND MODELLING

PC513CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30 marks SEE: 70 marks
Credits: 3	

Objectives:
1. To provide necessary basic concepts in probability and random processes for applications like random signals, linear systems in communication engineering.
2. To understand basic concepts of probability, one and two dimensional random variables and introduce standard distributions applicable to engineering which can describe real life phenomenon.
3. To understand the concept of queueing models and apply in engineering.
4. To understand significance of advanced queueing models.
5. To provide mathematical support in real life problems and develop probabilistic models which can be used in science and engineering.
Outcomes:
Students will be able to:
1. Understand fundamental knowledge of probability and standard distributions which describe real life phenomenon.
2. Understand basic concepts of one and two dimensional random variables and apply in engineering applications.
3. Apply the concept of random processes in engineering disciplines.
4.Acquire skills in analyzing queueing models.
5. Understand and characterize phenomenon which evolve with respect to time in a probabilistic manner.

UNIT-1
Probability and Random Variables: Probability – Axioms of probability – Conditional probability – Baye's theorem – Discrete and continuous random variables – Moments – Moment generating functions – Binomial, Poisson, Geometric, Uniform, Exponential and Normal distributions.
UNIT-2
Two-Dimensional Random Variables: Joint distributions – Marginal and conditional distributions – Covariance – Correlation and linear regression – Transformation of random variables – Central limit theorem (for independent and identically distributed random variables).
UNIT-3
Random Processes: Classification – Stationary process – Markov process – Poisson process – Discrete parameter Markov chain – Chapman Kolmogorov equations – Limiting distributions
UNIT-4
Queueing Models: Markovian queues – Birth and death processes – Single and multiple server queueing models – Little's formula – Queues with finite waiting rooms – Queues with impatient customers: Balking and reneging
UNIT-5
Advanced Queueing Models: Finite source models – M/G/1 queue – Pollaczek Khinchin formula – M/D/1 and M/EK/1 as special cases – Series queues – Open Jackson networks

Suggested Readings:
1. Gross, D., Shortle, J.F., Thompson, J.M and Harris, C.M. – Fundamentals of Queueing Theory, Wiley Student Edition 4th Edition, 2014.
2. Ibe, O.C. – Fundamentals of Applied Probability and Random Processes, Elsevier, 1st Indian Reprint, 2007
References:
1.Hwei Hsu – Schaum's Outline of Theory and Problems of Probability, Random Variables and Random Processes, Tata McGraw Hill Edition, New Delhi, 2004
2. Taha, H.A. – Operations Research, 9th Edition, Pearson India Education Services, Delhi, 2016.
3. Trivedi, K.S. – Probability and Statistics with Reliability, Queueing and Computer Science Applications, 2nd Edition, John Wiley and Sons, 2002.
4. Yates, R.D. and Goodman, D. J. – Probability and Stochastic Processes, 2nd Edition, Wiley India Pvt. Ltd., Bangalore, 2012.

PROFESSIONAL ELECTIVE – I

INFORMATION THEORY AND CODING

PC514CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30 marks SEE: 70 marks
Credits: 3	

Objectives:
1. To understand measure of information, entropy and information rate of sources.
2. To learn source coding techniques and Huffman coding.
3. To understand information channels and channel capacity.
4. To learn error control coding and linear blocks codes.
5. To understand cyclic codes and convolution codes with Viterbi decoding.
Outcomes:
Students will be able to:
1. Compute entropy and information rate for dependent & independent sources.
2. Design Huffman codes using Shannon's algorithm and KMI property.
3. Calculate mutual information and channel capacity of communication channels.
4. Design encoder/decoder for Linear Block codes and Hamming codes.
5. Encode data using cyclic & convolutional codes and decode using Viterbi algorithm.

UNIT-1
Information Theory: Introduction, Measure of information, Information content of message, Average Information content of symbols in Long Independent sequences, Average Information content of symbols in Long dependent sequences, Markov Statistical Model for Information Sources, Entropy and Information rate of Mark off Sources
UNIT-2
Source Coding: Encoding of the Source Output, Shannon's Encoding Algorithm, Source coding theorem, Prefix Codes, Kraft McMillan Inequality property- KMI, Huffman codes
UNIT-3
Information Channels: Communication Channels, Discrete Communication channels, Channel Matrix, Joint probability Matrix, Binary Symmetric Channel, System Entropies, Mutual Information, Channel Capacity, Channel Capacity of Binary Symmetric Channel
UNIT-4
Error Control Coding: Introduction, Examples of Error control coding, methods of Controlling Errors, Types of Errors, types of Codes, Linear Block Codes: matrix description of Linear Block Codes, Error detection & Correction capabilities of Linear Block Codes, Single error correction Hamming code, Table lookup Decoding using Standard Array
UNIT-5
Binary Cyclic Codes: Algebraic Structure of Cyclic Codes, Encoding using an (n-k) Bit Shift register, Syndrome Calculation, Error Detection and Correction **Convolution Codes:** Convolution Encoder, Time domain approach, Transform domain approach, Code Tree, Trellis and State Diagram, The Viterbi Algorithm

Suggested Readings:

1. K. Sam Shanmugam – Digital and Analog Communication Systems, John Wiley India Pvt Ltd, 1996
2. Simon Haykin – Digital Communication, John Wiley India Pvt Ltd, 2008
3. Ranjan Bose – ITC and Cryptography, TMH, II edition, 2007
4. J. Das, S.K. Mullick, P. K. Chatterjee – Principles of Digital Communication, Wiley, 1986 – Technology & Engineering
5. Bernard Sklar – Digital Communications: Fundamentals and Applications, Second Edition, Pearson Education, 2016, ISBN: 9780134724058
6. Hari Bhat, Ganesh Rao – Information Theory and Coding, Cengage, 2017
7. Todd K Moon – Error Correction Coding, Wiley Std. Edition, 2006

PROFESSIONAL ELECTIVE – I

COMPUTER GRAPHICS

PE515CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	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.
5. To understand hierarchical modeling, scene graphs and animation.
Outcomes:
Students 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. Create realistic images of 3-d objects that involve lighting shading aspects.
5. Create hierarchical models and animated scenes using scene graphs.

UNIT-1
Graphics Systems and Models: Graphics system, Images, Physical and Synthetic, Imaging system, Synthetic camera model, Programming interface, Graphics architectures, Programmable pipelines. **Graphics Programming:** Programming two-dimensional applications, OpenGL API, Primitives and attributes, Color, Viewing and Control functions.
UNIT-2
Input and Interaction: Input devices, Display lists & modeling, Programming event-driven input, Picking, building interactive models, Animating interactive programs, Logic operations. GeometricObjects: Three-dimensional primitives, Coordinate systems and frames,Framesin OpenGL, Modeling colored cube.
UNIT-3
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-4
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-5
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:
Edward Angel, Interactive Computer Graphics: A Top-Down Approach Using OpenGL, Pearson Education, 5th edition, 2009.
Francis S Hill Jr., Stephen M Kelley, Computer Graphics using OpenGL, Prentice-Hall Inc., 3rd Edition, 2007.
Jim X. Chen, Foundations of 3D Graphics Programming using JOGL and Java3D, Springer Verlag, 2006.
Hearn Donald, Pauline M Baker, Computer Graphics, 2nd edition, 1995.

PROFESSIONAL ELECTIVE – I

SOFTWARE REUSE TECHNIQUES

PE516CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30marks SEE: 70marks
Credits: 3	

Objectives:
1.Demonstration of patterns related to object oriented design.
2.Describe the design patterns that are common in software applications.
3. Analyze a software development problem and express it
4.Design a module structure to solve a problem, and evaluate alternatives
5.Implement a module so that it executes efficiently and correctly
Outcomes:
Students will be able to:
1. Construct a design consisting of a collection of modules.
2. Exploit well-known design patterns such as Iterator, Observer, Factory and Visitor.
3. Distinguish between different categories of design patterns.
4. Ability to understand and apply common design patterns to incremental/iterative development.
5. Ability to identify appropriate patterns for design of given problem and design the software using Pattern Oriented Architectures.

UNIT-1
Software Reuse Design Patterns: Software reuse success factors, Reuse driven software engineering as business, Object oriented software engineering, Applications and Component subsystems, Use case components, Object components **Intro:** What is a Design Pattern?, Design Patterns in Smalltalk MVC, Describing Design Patterns, The Catalog of Design Patterns, Organizing the Catalog, How Design Patterns Solve Design Problems, How to Select a Design Pattern
UNIT-2
A Case Study – Designing a Document EditorTopics: Design Problems, Document Structure, Formatting, Embellishing the User Interface, Supporting Multiple Look-and-Feel Standards, Supporting Multiple Window Systems
UNIT-3
Creational Patterns: Abstract Factory, Builder, Factory Method, Prototype, Singleton, Discussion of Creational Patterns
UNIT-4
Structural Patterns: Adapter, Bridge, Composite, Decorator, Façade, Flyweight, Proxy

UNIT-5
Behavioral Patterns: Chain of Responsibility, Command, Interpreter, Iterator, Mediator, Memento, Observer, State, Strategy, Template Method, Visitor, Discussion of Behavioral Patterns, What to Expect from Design Patterns
Suggested Readings:
Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides – Design Patterns: Elements of Reusable Object-Oriented Software, Pearson Education 1995
Ivar Jacobson, Martin Griss, Patrick Johnson – Software Reuse: Architecture, Process and Organization for Business Success, ACM Press 1997
Mark Grand – Pattern's in JAVA Vol-I, Wiley DreamTech
Mark Grand – Pattern's in JAVA Vol-II, Wiley DreamTech
Mark Grand – JAVA Enterprise Design Patterns Vol-III, Wiley DreamTech
Eric Freeman – Head First Design Patterns, O'Reilly

COMPUTER NETWORKS LAB

PE551CS	
Instruction: 2 periods per week	Duration of SEE: 3 hours
CIE: 25 marks	SEE: 50 marks
Credits: 1	

Pre-requisites:
1. Data Communications
Objectives:
1. Learn to communicate between 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 to analyze the performance of various network protocols.
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 various routing algorithms.
5. Analyze the performance of various network protocols.

List of Programs:

S. No.	Description of Program
1.	Running and using services/commands like telnet, 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 Examples: 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 Simulation tools.
7.	Programming using raw sockets.
8.	Programming using RPC.

ARTIFICIAL INTELLIGENCE LAB

PC553CS	
Instruction: 2 periods per week	Duration of SEE: 3 hours
CIE: 25 marks	SEE: 50 marks
Credits: 1	

Pre-requisites:
1. Basics of programming in Python
Objectives:
1. To apply programming skills to formulate the solutions for computational problems.
2. To study implementation first order predicate calculus using Prolog
3. To familiarize with basic implementation of NLP with the help of Python libraries NLTK
4. To understand python library scikit-learn for building machine learning models
5. To enrich knowledge to select and apply relevant AI tools for the given problem
Outcomes:
CO1: Design and develop solutions for informed and uninformed search problems in AI.
CO2: Demonstrate reasoning in first order logic using Prolog
CO3: Apply machine learning algorithms: dataset preparation, model selection, model building etc.
CO4: Demonstrate and enrich knowledge to select and apply python libraries to synthesize information and develop supervised learning models
CO5:Develop a case study in multidisciplinary areas to demonstrate use of AI

List of Programs:	
S. No.	**Description of Program**
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.	Basic Data Preprocessing Installation of python environment/Anaconda IDE for machine learning: a. installing python modules/Packages like scikit-learn, Keras and Tensorflow etc. b. Programs involving pandas, Numpy and Scipy libraries.

5.	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
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	

DESIGN AND ANALYSIS OF ALGORITHMS LAB

PE552CS	
Instruction: 2 periods per week	Duration of SEE: 3 hours
CIE: 25 marks	SEE: 50 marks
Credits: 1	

Pre-requisites:
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. Apply graph searching techniques
2. Design analgorithm in a effective manner
3. Apply iterative and recursive algorithms.
4. Implement optimization algorithms for specific applications.
5. Design optimization algorithms for specific applications.

List of Programs:	
S. No.	
1.	Print all the nodes reachable from a given starting node in a digraph using BFS methodand Check whether a given graph is connected or not using DFS method.
2.	Sort a given set of elements and determine the time required to sort the elements using following algorithms: • Merge Sort • Quick Sort
3.	Implement Knapsack problem using • Brute Force Approach • Greedy Method • Dynamic Programming
4.	Find Minimum Cost Spanning Tree of a given undirected graph 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.	Implement All-Pairs Shortest Paths Problem using Floyd's algorithm
8.	Implement the following using BackTracking • NQueen'sproblem • Hamiltonian Cycle • Graph Coloring

SCHEME OF INSTRUCTION & EXAMINATION

B.E (Computer Science and Engineering)

SEMESTER-VI

S. No.	Course Code	Course Title	Scheme of Instruction				Scheme of Examination			Credits
			L	T	D/P	Contact Hours/WK	CIE	SEE	Duration in Hrs.	
Theory Courses										
1	PC601CS	Quantum Computing	3	-	-	3	30	70	3	3
2	PC602CS	Nano Technology and Computing	3		-	3	30	70	3	3
3	PC603CS	Machine Learning & Deep Learning	3	-	-	3	30	70	3	3
4	PC604CS	Cryptography and Network Security	3	-	-	3	30	70	3	3
5	PC62XCS	Professional Elective-II	2	-	-	2	30	70	3	2
6	PC605CS	Computer Ethics	3			3	30	70	3	3
Practical / Laboratory Courses										
7	PC651CS	Quantum Computing Lab	-	-	2	2	25	50	3	1
8	PC652CS	Machine Learning & Deep Learning Lab	-	-	2	2	25	50	3	1
9	PW653CS	Mini Project	-	-	4	4	25	50	3	2
10	SI671CS	Summer Internship*	-	-	-	-	-	-	-	-
TOTAL			**17**	**-**	**08**	**25**	**255**	**570**	**27**	**21**

Professional Elective -II	
Course Code	**Course Title**
PC601 IT/PE	Embedded Systems
PE622CS	Design Thinking
PE623CS	Software Testing
PE624CS	Advanced Computer Architecture
PE625CS	Image Processing
PE511DS	Information Retrieval System
Open Elective - I	
Course Code	**Course Title**
OE601CS	OOP using JAVA (Not for CSE & IT Students)
OE602CS	Data Structure and Algo. (Not for CSE & IT Students)

PC601CS	QUANTUM COMPUTING					
Prerequisites			L	T	P	C
			3	-	-	3
Evaluation	CIE	30 Marks	SEE		70 Marks	

Course Objectives:

1. Understand the mathematical foundations and physical principles that form the basis of quantum computing.
2. Gain knowledge of fundamental computer science concepts, information theory, and quantum mechanics relevant to quantum information processing.
3. Explore quantum architectures, qubits, quantum gates, circuits, and hardware technologies used in quantum computers.
4. Study and analyze major quantum algorithms and their advantages over classical computational methods.
5. Examine the impact of quantum computing on modern cryptographic systems and understand emerging quantum-resistant security techniques.

Course Outcomes:

After successful completion of the course, students will be able to:

1. Apply linear algebra, complex numbers, and quantum physics principles to model and analyze quantum systems.
2. Explain the foundations of computer science, information theory, and quantum mechanics relevant to quantum computing.
3. Analyze quantum architectures, qubits, quantum gates, circuits, and hardware technologies.
4. Evaluate and apply fundamental quantum algorithms to solve computational problems efficiently.
5. Assess the impact of quantum computing on modern cryptography and emerging quantum-safe security solutions.

UNIT – I

Linear Algebra Basics: Basic Algebra, Matrix Math, Vectors and Vector Spaces, Complex Numbers: Definition of Complex Numbers, Algebra of Complex Numbers, Complex Numbers Graphically, Vector Representations of Complex Numbers, Pauli Matrice, Transcendental Numbers.

Basic Physics for Quantum Computing: Quantum Physics Essentials, Basic Atomic Structure, Hilbert Spaces, Uncertainty, Quantum States, Entanglement.

UNIT – II

Computer Science for Quantum Computing: Computational Complexity, Coding Theory, Logic Gates, Information Theory: Basic Probability, Set Theory, Information Theory, Quantum Information.

Quantum Theory: Quantum Mechanics, Quantum Decoherence, Quantum Electrodynamics, Quantum Chromodynamics, Feynman Diagram, Quantum Entanglement and QKD.

UNIT – III

Quantum Architecture: Qubits, Quantum Gates, quantum logic, Quantum Circuits, The D-Wave Quantum Architecture.

Quantum Hardware: Addressing Decoherence, Topological Quantum Computing, Quantum Essentials, Quantum Networking.

UNIT – IV

Quantum Algorithms: Deutsch's Algorithm, Deutsch-Jozsa Algorithm, Bernstein- Vazirani Algorithm, Simon's Algorithm, Shor's Algorithm, Grover's Algorithm.

UNIT – V

Quantum Computing and Cryptography: Current Asymmetric Algorithms, RSA, Diffie-Hellman, Elliptic Curve. The Impact of Quantum Computing on Cryptography: Asymmetric Cryptography, Specific Algorithms and Applications.

Suggested Readings:

1. Dr. Chuck Easttom, Quantum Computing Fundamentals, Pearson.

References:

1. Nielsen M. A., Quantum Computation and Quantum Information, Cambridge University Press
2. Quantum Computing for Computer Scientists by Noson S. Yanofsky and Mirco A. Mannucci
3. Benenti G., Casati G. and Strini G., Principles of Quantum Computation and Information, Vol. Basic Concepts. Vol. Basic Tools and Special Topics, World Scientific.
4. Pittenger A. O., An Introduction to Quantum Computing Algorithms.

PC 602CS	NANO TECHNOLOGY & COMPUTING					
Prerequisites	Engineering Physics, Basic Electronics		L	T	P	C
			3	1	-	4
Evaluation	CIE	30 Marks	SEE		70 Marks	

Course Objectives

After successful completion of this course, the students will be able to:

1. Understand the limitations of classical CMOS technology, thermodynamic constraints, and the need for emerging nano-computing paradigms.
2. Explain the fundamental principles of quantum mechanics and their significance in nanoscale electronic devices.
3. Analyze the operation and characteristics of nanoscale electronic devices, including Single-Electron Transistors (SETs), Carbon Nanotube FETs (CNTFETs), and Graphene Nanoribbon (GNR)-based circuits.
4. Examine the principles of spintronic devices, memristive systems, and Quantum-dot Cellular Automata (QCA) for next-generation computing applications.
5. Design combinational and sequential logic circuits using Quantum-dot Cellular Automata (QCA) for nano-computing systems.

Course Outcomes

Upon successful completion of this course, students will be able to:

1. Explain the physical and thermodynamic limitations of classical CMOS technology and justify the need for nano-computing technologies.
2. Apply the principles of quantum mechanics to analyze the behavior of nanoscale electronic devices and quantum systems.
3. Analyze the operation, characteristics, and applications of SETs, CNTFETs, Graphene Nanoribbons, and other nanoscale electronic devices.
4. Evaluate the performance and applications of spintronic devices, memristors, and Quantum-dot Cellular Automata in emerging computing architectures.
5. Design QCA-based combinational and sequential logic circuits, including majority gates, memory elements, shift registers, and arithmetic logic units (ALUs), for nano-computing applications.

UNIT-I:

Foundations and Limits of Classical Computing Limits of Silicon CMOS: Sub-threshold leakage, short-channel effects, electromigration and the physical limits of Moore's Law. Thermodynamic Limits: Landauer's principle, power dissipation limits, and irreversible vs. reversible computing.

UNIT-II:

Quantum Mechanics Fundamentals: Wave-particle duality, Schrodinger wave equation, quantum tunnelling and electron confinement. Heisenberg Uncertainty Principle, Quantum Entanglement, Superposition, Spin & Angular Momentum, Perturbation Theory, and Feynman Path Integrals.

UNIT-III:

Nanoscale Electronic Devices Single-Electron Transistors (SET): Coulomb blockade, single-electron tunneling, and SET based logic gates. Carbon Nanotube Electronics: Carbon Nanotube FETs (CNTFETs), ballistic transport, Graphene Nanoribbons (GNR), and CNT-based logic circuits.

UNIT-IV:

Spintronics & Memristors: Spin-transfer torque (STT) devices, giant magnetoresistance (GMR), and memristive systems for neuromorphic nano-computing. Quantum-dot Cellular Automata (QCA) QCA Basics: QCA cell structure, polarization states, electrostatic interactions, and clocking schemes (Switch, Hold, Release, Relax).

UNIT-V:

QCA Logic Design: Majority gates, invertors, wire crossings (coplanar and multilayer), and full-adder design. Sequential QCA: Memory cell design, shift registers, and QCA-based arithmetic logic units (ALUs).

Reference Books:

1. Nanoelectronics and Information Technology by Rainer Waser.
2. Introduction to Nanocomputing by Sahni, V.
3. Quantum-dot Cellular Automata by Michael Niemier.

MACHINE LEARNING AND DEEP LEARNING

PC603CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30marks SEE: 70marks
Credits: 3	

Objectives:
1.To learn ensemble techniques and various unsupervised learning algorithms.
2.To explore Neural Networks and Deep learning basics.
3. Explore advanced concepts of CNN and importance of transfer learning.
4. Expertise object detection using advanced techniques.
5. Generate new images using computer vision techniques.
Outcomes:
Students will be able to:
1.Apply ensemble techniques for improvement of classifiers.
2. Classify images using CNN.
3. Use advance CNN Architectures and transfer learning techniques.
4. Explore object detection techniques and generate images with masking of various classes.
5. Generate new images with GAN and other computer vision techniques.

UNIT-1
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-2
Neural Networks: Introduction, Artificial Neural Networks, Single-Layer Feed-Forward Networks, Multi-Layer Feed-Forward Networks, Multilayer Perceptron, Back-propagation algorithm, Training strategies, Activation Functions, Gradient Descent For Machine Learning, Radial basis functions, Hopfield network, Recurrent Neural Networks. **Convolution neural networks:** Image classification using MLP, CNN architecture, Basic components of a CNN, Image classification using CNNs, Adding dropout layers to avoid overfitting, Convolution over color images (3D images), Case study: Image classification for color images.
UNIT-3
Advanced CNN architectures: CNN design patterns, LeNet-5, AlexNet, VGGNet, Inception and GoogleNet, ResNet Transfer learning: Problems that transfer learning solves, Transfer learning, Transfer learning working, Transfer learning approaches, Choosing the appropriate level of transfer learning, Open source datasets, Case study: A pretrained network as a feature extractor.

UNIT-4
Object detection with R-CNN, SSD, and YOLO: General object detection framework, Region-based convolutional neural networks (R-CNNs), Single-shot detector (SSD), You only look once (YOLO). Segmentation, Mask R-CNN and Instance Segmentation, U-Net and Semantic Segmentation Reducing Noise with Autoencoders: Creating a simple fully connected autoencoder, creating a convolutional autoencoder, Denoising images with autoencoders, Spotting outliers using Autoencoders, Variational Autoencoders, creating an inverse image search index with deep learning, Implementing a variational autoencoder.
UNIT-5
Generative adversarial networks: GAN architecture, Evaluating GAN models, Popular GAN applications. DeepDream and neural style transfer: How convolutional neural networks see the world, DeepDream, Neural style transfer **Deep Learning on Edge Devices with CPU/GPU:** Optimization, Overview of deep learning on edge devices, Techniques used for GPU/CPU optimization, Overview of MobileNet.
Suggested Readings:
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.Ian Goodfellow, Yoshua Bengio, Aaron Courville - Deep Learning – Second Edition-MIT Press , 2016.
8.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

PE604CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30marks SEE: 70marks
Credits: 3	

Objectives:
1. Understand security concepts, Ethics in Network Security.
2. Obtain knowledge on mechanisms to encounter threats
3. Appreciate and apply relevant cryptographic techniques
4. Apply authentication services and security mechanisms
5. Comprehend computer network access control and ethics in network security.
Outcomes:
Students will be able to:
1. Develop familiarity with cryptography and security techniques
2. Master fundamentals of secret and public cryptography
3. Utilize the master protocols for security services
4. Identify network security threats and counter-measures
5. Propose network security designs using available secure solutions

UNIT-1
Basic Principles: Security Goals, Cryptographic Attacks, Services and Mechanisms, Mathematics of Cryptography
UNIT-2
Symmetric Encryption: Mathematics of Symmetric Key Cryptography, Introduction to Modern Symmetric Key Ciphers, Data Encryption Standard, Advanced Encryption Standard.
UNIT-3
Asymmetric Encryption: Mathematics of Asymmetric Key Cryptography, Asymmetric Key Cryptography
UNIT-4
DataIntegrity,DigitalSignatureSchemes&KeyManagement:MessageIntegrityandMessage Authentication,CryptographicHashFunctions,DigitalSignature,KeyManagement
UNIT-5
NetworkSecurity: SecurityatApplicationlayer(PGPandS/MIME),SecurityattheTransport Layer(SSLandTLS),SecurityattheNetworkLayer(IPSec,SystemSecurity)
Suggested Readings:
1.Cryptography and Network Security, Behrouz A Forouzan, Debdeep Mukhopadhyay, (3e) Mc Graw Hill
2. Cryptography and Network Security, William Stallings, (6e) Pearson.
3. Everyday Cryptography, Keith M.Martin, Oxford.
4. Network Security and Cryptography, Bernard Meneges, Cengage Learning
5. Cryptography and Network Security – by Atul Kahate – TMH.
6. Cyber Security Operations Handbook – by J.W. Rittiaghouse and William M.Hancok – Elseviers.
7. Khairol Amali Bin Ahmad , Khaleel Ahmad , Uma N. Dulhare ,Functional Encryption , EAI/Springer Innovations in Communication and Computing,1st ed. 2021 Edition

PC 605 CS	Computer Ethics					
Prerequisites			L	T	P	C
			3	-	-	3
Evaluation	CIE	30 Marks	SEE		70	

Course Objectives (COBs)

After successful completion of this course, students will be able to:

1. Understand the basic concepts of morality, ethics, and law in computing.
2. Learn professional ethics and ethical responsibilities of computing professionals.
3. Understand privacy, security, anonymity, and ethical issues in the digital world.
4. Gain knowledge of intellectual property rights, cybercrime, and computer ethics.
5. Examine the ethical and social impact of emerging technologies such as AI, IoT, and social media.

Course Outcomes (COs)

After successful completion of this course, the students will be able to:

CO1: Explain the concepts of morality, ethics, law, ethical theories, and their significance in computing and information technology.

CO2: Apply professional codes of conduct and ethical decision-making principles to resolve ethical issues encountered in computing professions.

CO3: Analyze ethical, legal, and social issues related to anonymity, privacy, civil liberties, and data protection in digital environments.

CO4: Analyze intellectual property rights, computer crimes, digital evidence, and ethical responsibilities in computer forensic investigations.

CO5: Evaluate the ethical and societal implications of emerging technologies such as Artificial Intelligence, IoT, online social networks, biometrics, surveillance, and digital media.

UNIT-I:

Morality and Ethics: Moral code of behaviour, Purpose of Law, Penal code, Morality and the law; Ethical theories, Functional definition of ethics, Ethical reasoning and decision making, codes of ethics, technology and values.

UNIT-II:

Ethics and Professions: Professional code of conduct, Professional decision making and ethics, Whistleblowing, harassment and discrimination, ethical and moral implications of profession.

UNIT-III:

Anonymity, Privacy and Civil Liberties: Anonymity in the Internet, its advantages and disadvantages, Types and value of Privacy, Privacy protection and civil liberties, implications for privacy of the Aadhaar database,Workplace privacy and surveillance, Ethics for privacy, Ethical and legal basis for privacy. Case studies of Cambridge Analytica, SECC in India.

UNIT-IV:

Intellectual Property Rights, Computer Crime and Ethics: Computer Products and services, foundations of Intellectual Property Rights (IPR), Ownership, IPR crimes, Protecting computer software under IPR, Transnational issues. Digital evidence, preserving digital evidence, analysis of digital evidence, ethical implications and responsibilities in computer forensic investigations.

UNIT-V:

Social Implications of Information Technology: Advances in Artificial Intelligence, AI and ethics, bias in AI algorithms, Online social networks, Ethical issues in online social networks, security and crimes in online social networks; Internet of Things (IoT) and Location tracking, ethics of tracking, ethics of personalised search/ advertisements in online social networks and search engines, self-driving vehicles and ethical/legal issues, surveillance and ethical issues, use of biometrics for authentication, ethical implications of biometric technologies, fake news

and implications of its spread via social media platforms, digital divide and its implications, sexual predators and pornography on the Internet and its implications.

Text Book:

1. Ethical and Social Issues in the Information Age, Joseph Migga Kizza, 6th ed. 2017 Edition, ISBN-10: 1447149890, ISBN-13: 978-1447149897

Reference Books:

[1]https://theprint.in/politics/exclusive-inside-story-cambridge-analytica-actually india/44012/
[2] https://thewire.in/tech/what-we-know-and-what-we-dont-about-what-cambridge analytica-did-in-india
[3] https://www.huffingtonpost.in/entry/aadhaar-national-social-registry-database-modi in 5e6f4d3cc5b6dda30fcd3462
[4] https://www.huffingtonpost.in/entry/one-year-after-aadhaar-judgement-we-are-still-not listening in 5d8bb628e4b0ac3cdda24659
[5] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6596313/
[6] https://yjolt.org/sites/default/files/21_yale_j.l._tech._106_0.pdf
[7] https://hbr.org/2019/10/what-do-we-do-about-the-biases-in-ai
[8] https://www.nytimes.com/2019/11/19/technology/artificial-intelligence-bias.html
[9] https://aibusiness.com/three-notable-examples-of-ai-bias/
[10] https://thewire.in/tech/artificial-intelligence-has-a-gender-bias-problem-just-ask-siri
[11] https://www.theguardian.com/books/2019/feb/02/age-of-surveillance-capitalism shoshana-zuboff-review
[12] https://www.nature.com/articles/d41586-018-07135-0
[13] https://www.newyorker.com/science/elements/a-study-on-driverless-car-ethics-offers-a troubling-look-into-our-values
[14] https://towardsdatascience.com/how-ethical-is-facial-recognition-technology 8104db2cb81b [15] https://www.technologyreview.com/f/615068/facial-recognition-european-union temporary-ban-privacy-ethics-regulation/
[16]https://www.meity.gov.in/writereaddata/files/Information%20Technology%20(Intermediary%20Guielines%20and%20Digital%20Media%20Ethics%20Code)%20Rules%2C%202021%20(updated%2006.04. 2023)-.pdf
[17] https://www.meity.gov.in/content/cyber-laws
[18]https://www.tec.gov.in/pdf/Studypaper/AI%20Policies%20in%20India%20A%20status%20Paper%20 final.pdf

PROFESSIONAL ELECTIVE – II
EMBEDDED SYSTEMS

PC601IT/PE	Duration of SEE: 3 hours
Instruction: 2 periods per week	CIE: 30marks SEE: 70marks
Credits: 2	

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.
4. To learn IPC, synchronization and task management in embedded systems.
5. To understand RTOS services, task scheduling and embedded software tools.
Outcomes:
Students will be able to:
1. Classify embedded systems and explain design process with examples.
2.Interface devices using serial/parallel buses and wireless protocols.
3. Develop interrupt-driven device drivers with DMA support.
4. Implement IPC and synchronization for multi-task embedded applications.
5. Design RTOS based real-time systems and evaluate scheduling performance.

UNIT-1
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-2
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 systems network protocols, Wireless and mobile system protocols.
UNIT-3
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-4
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, Inter-process communication, Signal function, Semaphore functions, Message Queue functions, Mailbox functions, Pipe functions, Socket functions, RPC functions.

UNIT-5
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.

PROFESSIONAL ELECTIVE – II
DESIGN THINKING

PE622CS	Duration of SEE: 3 hours
Instruction: 2 periods per week	CIE: 30marks SEE: 70marks
Credits: 2	

Objectives:
1. To familiarize students with design thinking concepts and principles
2. To ensure students can practice the methods, processes and tools of design thinking
3. To ensure students can apply the design thinking approach and have ability to model real world situations
4. To enable students to analyze primary and secondary research in the introduction to design thinking.
5.To learn tools for prototype and testing.
Outcomes:
Students will be able to:
1. Understand the basics of Design thinking.
2. Learn the steps involved in design process
3. Understand the different phases of design thinking
4. Generate ideas and find solutions
5. Use tools for generation of ideas

UNIT-1
DESIGN THINKING FOR INNOVATION Introduction to Design Thinking, Understanding the principles of Design Thinking, Business Model Innovation, Challenges Best-Suited for Design Thinking, Product Life Cycle
UNIT-2
PROCESS OF DESIGN Introduction Design Process, Four Step, Five Step, Twelve Step, Creativity and Innovation in Design Process, Design limitation, Creative Thinking, Lean Canvas Model and other Business Models
UNIT-3
PHASES IN DESIGN THINKING Understand, Observe, Define, Ideate, Prototype, Test, Reflect Problem Statement, Empathy, The 5 Whys, Stakeholder map, Empathy map, Personas, Peer observation, Trend analysis
UNIT-4
SOLUTION/IDEA GENERATION Story Telling, Context mapping, Critical items diagram, Brainstorming, Matrix and Voting methods, Analogies, Benchmarking, Utility maps
UNIT-5
TOOLS AND TECHNIQUES FOR PROTOTYPE AND TEST Types of Prototype, Exploration Map, Blueprint, MVP, Testing Sheets, Solution Feedback Capturing Tools, Structured Usability Testing, A/B Testing, Design Thinking Applications Case Studies

Suggested Readings:
1.Gavin Ambrose and Paul Harris – Design Thinking, AVA Publishing, 2010
2.David Raizman – History of Modern Design, 2nd edition, Laurence King Publishing Ltd., 2010
3.Michael Lewrick, Patrick Link, Larry Leifer – The Design Thinking Toolbox: A Guide to Mastering the Most Popular and Valuable Innovation Methods, Wiley Publishing
4.Design Thinking for Dummies – Wiley
5.Tom Kelley, Jonathan Littman – Ten Faces in Innovation, Currency Books, 2006
6.G. Pahl, W. Beitz, J. Feldhusen, KH Grote – Engineering Design: A Systematic Approach, 3rd edition, Springer, 2007
7.The Field Guide to Human-Centered Design – by Design Kit

PROFESSIONAL ELECTIVE – II
SOFTWARE TESTING

PE623CS	Duration of SEE: 3 hours
Instruction: 2 periods per week	CIE: 30marks SEE: 70marks
Credits: 2	

Objectives:
1. To understand and Learn the basic concepts of Testing
2. To follow the methodology of White Box Testing.
3. To Obtain knowledge of Integration and System Testing
4. To understand the concepts of Object Oriented Testing.
5. To understand advanced testing concepts like TDD, exploratory testing and reviews.
Outcomes:
Students will be able to:
1.Gain the basic knowledge ofTesting.
2.Acquire the knowledge of White Box Testing methods
3.Test an application using Functional Tesing
4.Use Object Oriented Testing and Millennium Testing methods
5. Apply TDD, exploratory testing and best practices for quality assurance.

UNIT-1
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-2
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 – Usability Testing.
UNIT-3
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-4
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, GUI Testing.
UNIT-5
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.

PROFESSIONAL ELECTIVE – II
ADVANCED COMPUTER ARCHITECTURE

PE624CS	Duration of SEE: 3 hours
Instruction: 2 periods per week	CIE: 30marks SEE: 70marks
Credits: 2	

Objectives: Students will be able to:

1. Understand computer architecture fundamentals and performance evaluation.
2. Analyze instruction sets, addressing modes, and system software role.
3. Learn computer arithmetic and processor datapath design.
4. Study pipelining, cache organization, and performance improvement techniques.
5. Understand virtual memory, I/O systems, and multi-processor architectures

Outcomes:

Students 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-1

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

UNIT-2

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-3

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

Processor - Data path elements. Data path control.

UNIT-4

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-5

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

Input/Output-Hard-disk. Flash memory, I/O interfacing, Memory-mappedI/O, Interruptdriven 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).

PROFESSIONAL ELECTIVE – II
IMAGE PROCESSING

PE625CS	Duration of SEE: 3 hours
Instruction: 2 periods per week	CIE: 30marks SEE: 70marks
Credits: 2	

Pre-requisites:
1. Data Structures2. 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:
Students will be able to:
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-1
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, fundamentals of spatial filtering, smoothingspatial filters, sharpening spatial filters, combining spatial enhancement methods.
UNIT-2
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-3
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 squareerror filtering, constrained least squares filtering, geometric mean filter.

UNIT-4
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 compression standards.
UNIT-5
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

PROFESSIONAL ELECTIVE – II
INFORMATION RETRIEVAL SYSTEM

PE511DS	Duration of SEE: 3 hours
Instruction: 2 periods per week	CIE: 30marks SEE: 70marks
Credits: 2	

Objectives:
1. To provide the knowledge on information retrieval system capabilities
2. To delve into the concepts of Cataloging and Indexing.
3. To explore into Automatic Indexing's diverse classes and Document/Term Clustering
4. To educate learners about various user search techniques.
5. To discuss about information visualization and system evaluation.
Outcomes:
Students will be able to:
1. Understand various functionalities and capabilities of Information Retrieval System.
2. Gain knowledge on cataloging and data structure methodology for IRS.
3. Differentiate various clustering algorithms and indexing.
4. Differentiate various user search techniques and system search techniques.
5. Understand the concepts of information visualization and text search

UNIT-1
Introduction: Definition, Objectives, Functional Overview, Relationship to DBMS, Digital Libraries and Data Warehouses **Information Retrieval System Capabilities:** Search, Browse, Miscellaneous
UNIT-2
Cataloging and Indexing: Objectives, Index Process, Automatic Indexing, Information Extraction **Data Structures:** Introduction, Stemming Algorithms, Inverted File Structure, N-Gram Data Structure, PAT Data Structure, Hypertext Data Structure
UNIT-3
Automatic Indexing: Classes of Automatic Indexing, Statistical Indexing, Natural Language, Concept Indexing **Document and Term Clustering:** Introduction, Thesaurus Generation, Item Clustering, Hierarchy of clusters

UNIT-4
User Search Techniques: Search Statements and Binding, Selective Dissemination of Information Search, Weighted Searches of Boolean Systems, Searching the Internet and Hypertext
UNIT-5
Information Visualization: Introduction, Cognition and Perception, Information Visualization Technologies **Text Search Algorithms:** Introduction, Software Text Search Algorithms, Hardware Text Search Systems
Suggested Readings:
Kowalski, Gerald, Mark T Maybury – INFORMATION STORAGE AND RETRIEVAL SYSTEMS: Theory and Implementation, Second Edition, Kluwer Academic Press
References:
Gerald Kowalski – INFORMATION RETRIEVAL: Architecture and Algorithms
Frakes, W.B., Ricardo Baeza-Yates – Information Retrieval Data Structures and Algorithms, Prentice Hall, 1992
Modern Information Retrieval by Ricardo Baeza-Yates – Pearson Education
Information Storage & Retrieval by Robert Korfhage – John Wiley & Sons
e-Learning Resources:
https://class.coursera.org/nlp/lecture/178
http://cosmolearning.org/courses/database-design-417/video-lectures/
http://nptel.ac.in/video.php?subjectId=106102064

PC605CS	COMPUTER ETHICS					
Prerequisites			L	T	P	C
			3	-	-	-
Evaluation	CIE	50 Marks	SEE		-	

Course Objectives (COBs)

After successful completion of this course, students will be able to:

6. Understand the basic concepts of morality, ethics, and law in computing.
7. Learn professional ethics and ethical responsibilities of computing professionals.
8. Understand privacy, security, anonymity, and ethical issues in the digital world.
9. Gain knowledge of intellectual property rights, cybercrime, and computer ethics.
10. Examine the ethical and social impact of emerging technologies such as AI, IoT, and social media.

Course Outcomes (COs)

After successful completion of this course, the students will be able to:

CO1: Explain the concepts of morality, ethics, law, ethical theories, and their significance in computing and information technology.

CO2: Apply professional codes of conduct and ethical decision-making principles to resolve ethical issues encountered in computing professions.

CO3: Analyze ethical, legal, and social issues related to anonymity, privacy, civil liberties, and data protection in digital environments.

CO4: Analyze intellectual property rights, computer crimes, digital evidence, and ethical responsibilities in computer forensic investigations.

CO5: Evaluate the ethical and societal implications of emerging technologies such as Artificial Intelligence, IoT, online social networks, biometrics, surveillance, and digital media.

UNIT-I:

Morality and Ethics: Moral code of behaviour, Purpose of Law, Penal code, Morality and the law; Ethical theories, Functional definition of ethics, Ethical reasoning and decision making, codes of ethics, technology and values.

UNIT-II:

Ethics and Professions: Professional code of conduct, Professional decision making and ethics, Whistleblowing, harassment and discrimination, ethical and moral implications of profession.

UNIT-III:

Anonymity, Privacy and Civil Liberties: Anonymity in the Internet, its advantages and disadvantages, Types and value of Privacy, Privacy protection and civil liberties, implications for privacy of the Aadhaar database [4],Workplace privacy and surveillance, Ethics for privacy, Ethical and legal basis for privacy. Case studies of Cambridge Analytica [1] [2], SECC in India [3].

UNIT-IV:

Intellectual Property Rights, Computer Crime and Ethics: Computer Products and services, foundations of Intellectual Property Rights (IPR), Ownership, IPR crimes, Protecting computer software under IPR, Transnational issues. Digital evidence, preserving digital evidence, analysis of digital evidence, ethical implications and responsibilities in computer forensic investigations.

UNIT-V:

Social Implications of Information Technology: Advances in Artificial Intelligence, AI and ethics, bias in AI algorithms [7], [8], [9], [10], Online social networks, Ethical issues in online social networks, security and crimes in online social networks; Internet of Things (IoT) and Location tracking [5], ethics of tracking, ethics of personalised search/

advertisements in online social networks and search engines, self-driving vehicles and ethical/legal issues [12] [13], surveillance and ethical issues [11], use of biometrics for authentication, ethical implications of biometric technologies [14] [15], fake news and implications of its spread via social media platforms, digital divide and its implications, sexual predators and pornography on the Internet and its implications.

Text Book:

2. Ethical and Social Issues in the Information Age, Joseph Migga Kizza, 6th ed. 2017 Edition, ISBN-10: 1447149890, ISBN-13: 978-1447149897

Reference Books:

[1]https://theprint.in/politics/exclusive-inside-story-cambridge-analytica-actually india/44012/
[2] https://thewire.in/tech/what-we-know-and-what-we-dont-about-what-cambridge analytica-did-in-india
[3] https://www.huffingtonpost.in/entry/aadhaar-national-social-registry-database-modi in 5e6f4d3cc5b6dda30fcd3462
[4] https://www.huffingtonpost.in/entry/one-year-after-aadhaar-judgement-we-are-still-not listening in 5d8bb628e4b0ac3cdda24659
[5] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6596313/
[6] https://yjolt.org/sites/default/files/21_yale_j.l._tech._106_0.pdf
[7] https://hbr.org/2019/10/what-do-we-do-about-the-biases-in-ai
[8] https://www.nytimes.com/2019/11/19/technology/artificial-intelligence-bias.html
[9] https://aibusiness.com/three-notable-examples-of-ai-bias/
[10] https://thewire.in/tech/artificial-intelligence-has-a-gender-bias-problem-just-ask-siri
[11] https://www.theguardian.com/books/2019/feb/02/age-of-surveillance-capitalism shoshana-zuboff-review
[12] https://www.nature.com/articles/d41586-018-07135-0
[13] https://www.newyorker.com/science/elements/a-study-on-driverless-car-ethics-offers-a troubling-look-into-our-values
[14] https://towardsdatascience.com/how-ethical-is-facial-recognition-technology 8104db2cb81b [15] https://www.technologyreview.com/f/615068/facial-recognition-european-union temporary-ban-privacy-ethics-regulation/
[16]https://www.meity.gov.in/writereaddata/files/Information%20Technology%20(Intermediary%20Guielines%20and%20Digital%20Media%20Ethics%20Code)%20Rules%2C%202021%20(updated%2006.04. 2023)-.pdf
[17] https://www.meity.gov.in/content/cyber-laws
[18]https://www.tec.gov.in/pdf/Studypaper/AI%20Policies%20in%20India%20A%20status%20Paper%20 final.pdf

OPEN ELECTIVE – I

OOP using JAVA (Not for CSE & IT Students)

OE601CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30marks SEE: 70marks
Credits: 3	

Objectives:
1. Understand OOP concepts and apply them using Java. 2. Learn core Java programming including classes, inheritance, and interfaces. 3. Implement exception handling and multithreading in Java programs. 4. Work with Java I/O, Collections and String handling for data processing. 5. Design GUI applications using AWT and event handling.
Outcomes:
Students will be able to: **CO1** Explain and apply Object Oriented concepts. **CO2** Write, compile and execute Java programs. **CO3** Handle exceptions and implement multithreading. **CO4** Use Java Collections, I/O streams, and File handling. **CO5** Develop GUI based applications using AWT components.

UNIT-1
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-2
Exceptions Handling, Multithreaded Programming, I/O basics, Reading Console input and output, Reading and Writing Files, Print Writer Class, String Handling.
UNIT-3
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-4
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-5
Java I/O classes and interfaces, Files, Stream and Byte classes, Character Streams, Serialization.
Suggested Readings:
1.Herbert Schildt, The Complete Reference Java, 7th Edition, Tata McGraw Hill, 2005.
2. James M Slack, Programming and Problem solving with JAVA, Thomson Learning, 2002
3. C Thomas Wu, An Introduction to Object Oriented programming with Java, Tata McGraw Hill, 2005.

OPEN ELECTIVE – I
DATA STRUCTURE AND ALGO. (NOT FOR CSE & IT STUDENTS)

OE602CS	Duration of SEE: 3 hours
Instruction: 3 periods per week	CIE: 30marks SEE: 70marks
Credits: 3	

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
5. To understand graph representations and apply graph traversal and Minimum Spanning Tree algorithms to solve real-world problems.
Outcomes:
Students 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-1
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-2
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-3
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-4
Trees: Introduction,BinaryTrees,Binary Tree Traversals,Heaps. **BinarySearch trees(BST):** Definition,Searchinganelement,InsertionintoaBST,DeletionfromaBST. **EfficientBinarySearchTrees:**AVLTrees:Definition,Searchinganelement,InsertionintoaAVL

UNIT-5
Graphs: GraphAbstractDataType,ElementaryGraphoperations(DFSandBFS),MinimumCost Spanning Trees (Prim‘s and Kruskal‘s Algorithms). **SortingandSearching:**Insertionsort,Quicksort,BestcomputingtimeforSorting, Mergesort,Heapsort,shellsort,SortingonSeveralKeys,ListandTableSorts,SummaryofInternal Sorting, Linear and Binary Search algorithms.
Suggested Readings:
1.Horowitz E, Sahni S and Susan Anderson-Freed, Fundamentals of Data structures in C, 2nd Edition (2008), Universities Press
2. Mark A Weiss, Data Structures and Algorithm Analysis In C, Second Edition (2002), Pearson
3. Kushwaha D. S and Misra A.K, Data structures A Programming Approach with C, Second Edition (2014), PHl.
4. Gilberg R. F and Forouzan B. A, Data structures: A Pseudocode Approach with C, Second Edition (2007), Cengage Leaming
5. Tanenbaum A. M , Langsam Y. Augenstein M. J, Data Structures using C, Second Edition (2008), Pearson.
6. Thomas H. Cormen, Charles E. Leiserson, Ronald L Rivest, Clifford Stein, Introduction to Algorithms, Third Edition (2009), MIT Press
7. Yedidyah Langsam , Moshe J. Augenstein ,Aaron M. Tenenbaum, DataStructures Using C and C++ , Second Edition (2009), PHI

QUANTUM COMPUTING LAB

PC651CS

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

Objectives:

1. To familiarize students with the principles of quantum computation and quantum information processing.
2. To provide hands-on experience in designing and simulating quantum circuits using IBM Quantum Composer.
3. To develop skills in implementing quantum algorithms using the Qiskit programming framework.
4. To enable students to analyze quantum gate operations, measurement processes, and quantum circuit behavior.
5. To expose students to practical applications of quantum computing in cryptography, search, optimization, and communication.

Outcomes:

1. Demonstrate the operation of single-qubit and multi-qubit quantum gates using quantum circuit simulators.
2. Design and simulate quantum circuits and interpret measurement results using IBM Quantum Composer.
3. Develop quantum programs using Python and Qiskit for implementing fundamental quantum operations.
4. Implement and analyze quantum algorithms such as Deutsch's, Deutsch–Jozsa, Grover's, and Shor's algorithms.
5. Evaluate the performance and applications of quantum computing techniques in solving computational problems.

LIST OF EXPERIMENTS:

1. Single Qubit Gate Simulation - Quantum Composer
2. Multiple Qubit Gate Simulation - Quantum Composer
3. Composing Simple Quantum Circuits with Q-Gates and Measuring Output into Classical Bits
4. IBM Qiskit Platform Introduction
5. Implementation of Shor's Algorithm
6. Implementation of Grover's Algorithm
7. Implementation of Deutsch's Algorithm
8. Implementation of Deutsch-Jozsa Algorithm

Note: Using Python and Qiskit

MACHINE LEARNING AND DEEP LEARNING LAB

PC652CS

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

Objectives:

1. Demonstration of different classifiers on different data.
2. Demonstrate ensembling of classifiers for solving real world problems
3. Make use of real world data to implement machine learning models.
4. Learn and apply Transfer learning approach for pretrained model.
5. Implement CNN model to classify images.

Outcomes:

Students must be able to:

1. Apply unsupervised learning and interpret the results.
2. Implement neural network architectures for various tasks such as image classification, object detection.
3. Apply techniques for hyperparameter tuning to improve model performance.
4. Apply transfer learning techniques to leverage pre-trained models for specific tasks.
5. Generate images with masking of various classes.

LIST OF EXPERIMENTS:

1. Demonstration of Clustering algorithms using k-means & Hierarchical algorithms(agglomerative etc). Interpret the clusters obtained.

2. Demonstrate ensemble techniques like boosting, bagging, random forests etc.Build a classifier, compare its performance with an ensemble technique like random forest. Evaluate various classification algorithms performance on a dataset using various measures like True Positive rate, False positive rate, precision, recall etc.

3. Write a program to train and validate the following classifiers for given data (scikit-learn):

a) Multi-layer Feed Forward neural network
b) Backpropagation techniques
c) Techniques to avoid overfitting

4. Design and implement to classify 32x32 images using MLP using tensorflow / keras and check the accuracy.

5. Design and implement a CNN model to classify multi category JPG images with tensorflow / keras and check accuracy. Predict labels for new images.

6. Design and implement a CNN model to classify multi category tiff images with tensorflow / keras and check the accuracy. Check whether your model is overfit / underfit / perfect fit and apply the techniques to avoid overfit and underfit like regularizers, dropouts etc.

7. Implement an image classification model using transfer learning techniques and check accuracy. Tune the required hyperparameters.

8. Implement R-CNN model for object detection. Check with Fast and Faster R-CNN models.

1. Implement a model to mask various categories with Semantic Segmentation.
2. Implement an Autoencoder to de-noise image.
3. Implement a GAN application to convert images.

MINI PROJECT

PW653CS	
Instruction: 2 periods per week	Duration of SEE: 3 hours
CIE: 25 marks	SEE: 50 marks
Credits: 1	

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.
5.To demonstrate developed work
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. To perform systematic literature survey using tools
3. Evaluate different solutions based on economic and technical feasibility.
4. Effectively plan a project and confidently perform all aspects of project management.
5. Demonstrate effective coding, written, presentation and oral communication skills.

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) Portal invited from the Ministries / PSUs / MNCs / NGOs to be worked out through.

The project could be classified as hardware, software, modeling, simulation etc. The project should 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.