With effect from the academic year 2016-2017

# Department of Mathematics

## Osmania University, Hyderabad

## M.Sc.[Computer Science]

Course under Choice Based Credit System

### SEMESTER – I

| Paper | Code | Paper Title | HpW | Marks | Credits |
|---|---|---|---|---|---|
| I | CS101T | Advanced Java Programming | 4 | 20 + 80 = 100 | 4 |
| II | CS102T | Operating Systems | 4 | 20 + 80 = 100 | 4 |
| III | CS103T | Software Engineering | 4 | 20 + 80 = 100 | 4 |
| IV | CS104T | Discrete Mathematics | 4 | 20 + 80 = 100 | 4 |
| V | CS105P | Advanced Java Lab | 6 | 75 | 3 |
| VI | CS106P | Operating Systems Lab | 6 | 75 | 3 |
| VII | CS107P | Software Engineering Lab | 4 | 50 | 2 |
| | | **Total** | **32** | **600** | **24** |

### SEMESTER – II

| Paper | Code | Paper Title | HpW | Marks | Credits |
|---|---|---|---|---|---|
| I | CS201T | Programming in Python | 4 | 20 + 80 = 100 | 4 |
| II | CS202T | Computer Networks | 4 | 20 + 80 = 100 | 4 |
| III | CS203T | Design and Analysis of Algorithms | 4 | 20 + 80 = 100 | 4 |
| IV | CS204T | Automata Theory | 4 | 20 + 80 = 100 | 4 |
| V | CS205P | Python Lab | 6 | 75 | 3 |
| VI | CS206P | Computer Networks Lab | 6 | 75 | 3 |
| VII | CS207P | Design and Analysis of Algorithms Lab | 4 | 50 | 2 |
| | | **Total** | **32** | **600** | **24** |

# Department of Mathematics

## Osmania University, Hyderabad

## M.Sc.[Computer Science]

Course under Choice Based Credit System

### SEMESTER – III

| Paper | Code | Paper Title | HpW | Marks | Credits |
|---|---|---|---|---|---|
| I | CS301T | Programming in C# | 4 | 20 + 80 = 100 | 4 |
| II | CS302T | Compiler Design | 4 | 20 + 80 = 100 | 4 |
| III | Elective<br>CS303T(A)<br>CS303T(B) | <br>Network Security<br>Big Data Analytics | 4 | 20 + 80 = 100 | 4 |
| IV | Elective<br>CS304T(A)<br>CS304T(B) | <br>Object Oriented Analysis and Design<br>Data Mining | 4 | 20 + 80 = 100 | 4 |
| V | CS305P | C# Lab | 6 | 75 | 3 |
| VI | CS306P | Compiler Design Lab | 6 | 75 | 3 |
| VII | Elective<br>CS307P(A)<br>CS307P(B) | <br>Network Security Lab<br>Big Data Analytics Lab | 4 | 50 | 2 |
| | | **Total** | **32** | **600** | **24** |

### SEMESTER – IV

| Paper | Code | Paper Title | HpW | Marks | Credits |
|---|---|---|---|---|---|
| I | CS401T | Computer Organization | 4 | 20 + 80 = 100 | 4 |
| II | CS402T | Cloud Computing | 4 | 20 + 80 = 100 | 4 |
| III | Elective<br>CS403T(A)<br>CS403T(B) | <br>Mobile Computing<br>Distributed Systems | 4 | 20 + 80 = 100 | 4 |
| IV | Elective<br>CS404T(A)<br>CS404T(B) | <br>Robotics and Artificial Intelligence<br>Internet of Things | 4 | 20 + 80 = 100 | 4 |
| V | CS405P | Project Work | 16 | 200 | 8 |
| | | **Total** | **32** | **600** | **24** |

# CS101T Advanced Java Programming

Theory: 4 Hours/Week Credits: 4

## Unit – I

Review of Core Java: Class Object, Object Oriented Concepts with respect to Java, Interfaces, Packages and Exception Handling, Applets, Overview of Collection Framework (No question to be set from above topics).
AWT: Introduction, AWT Class Hierarchy, Creating Container, Adding Components, Layout, Using Panel, Text Field, Text Area, List, Checkbox, Check Box Group, Choice, Event Handling, Dialog Boxes, ScrollBar, Menu.
Swing: Containment Hierarchy, Adding Components, JTextField, JPasswordField, JTable, JComboBox, JProgressBar, JList, JTree, JColorChooser, Dialogs.
Remote Method Invocation (RMI): Introduction, Remote Method Invocation, Java RMI Interfaces and Classes, an Application, Compiling the Program, Generating Stub Classes, Running the Program, Callback with an Application.

## Unit – II

Servlet: Server-Side Java, Servlet Alternatives, Servlet Strengths, Servlet Architecture, Servlet Life Cycle, GenericServlet, HttpServlet, Servlet Example, Passing Parameters to Servlets, Retrieving Parameters, Cookies, Filters.
Java Server Pages (JSP): Introduction, JSP Engines, How JSP Works, JSP and Servlet, Anatomy of a JSP Page, JSP Syntax, JSP Components, Beans, Session Tracking, Users Passing Control and Data between Pages, Sharing Session and Application Data.

## Unit – III

Java Database Connectivity (JDBC): Introduction, JDBC Drivers, JDBC Architecture, JDBC Classes and Interfaces, Loading a Driver, Making a Connection, Execute SQL Statement, SQL Statements, Retrieving Result, Getting Database Information, Scrollable and Updatable Resultset, Result Set Metadata.
Hibernate: Introduction, Writing POJO Class, Creating a Table, Writing a Hibernate Application, Compiling and Running Application, Book Application Using Annotation, Object Life Cycle, HQL, Using Native SQL Query, Named Queries, Generating DDL, Generator Class, Hibernate Tools.

## Unit – IV

Java Naming and Directory Interface (JNDI): Naming Concepts, Directory Concepts, Java Naming and Directory Interface, Specifying JNDI Properties, Name Servers, Naming Operations, Working with Directory.
Overview of J2EE, Introduction to JavaBeans, Advantages of JavaBeans, Properties of JavaBeans with examples, JavaBeans API, Enterprise JavaBeans (EJB), Applications using Session Beans and Entity Beans, Introduction to Struts Framework.
Java Server Faces (JSF): Introduction, Simple Application, Request Processing Life-Cycle, Tracing Phases, Managed Bean, Basic JSF Tags, Expression Language, Event Handling with Example, Page Navigation.

**Text** Uttam K. Roy, *Advanced Java programming*

**References**

1. Herbertt Schildt, *Java Complete Reference*
2. Cay S. Horstmans, Gray Coronell, *Core Java Vol. II – Advanced Features*
3. Sharanam Shah, Vaishali Shah, *Java EE 7 for Beginners*

# CS102T Operating Systems

Theory: 4 Hours/Week Credits: 4

## Unit – I

Introduction: Computer-System Architecture, Operating-System Structure, Operating-System Operations, Process Management, Memory Management, Storage Management, Protection- Security, Kernel Data Structures, Computing Environments, Open-Source Operating Systems.

Operating-System Structures: Operating-System Services, User Interface for Operating-System, System Calls, Types of System Calls, Operating-System Design and Implementation, Operating-System Structure, Operating-System Debugging.

Process Management: Process Concept, Process Scheduling, Operations on Processes, Inter process Communication, Examples of IPC Systems, Communication in Client–Server Systems.

Threads: Overview, Multithreading Models, Threading Issues.

Process Synchronization: Concept, Critical-Section Problem, Peterson's Solution, Synchronization, Classic Problems of Synchronization, Semaphores, Monitors.

## Unit – II

CPU Scheduling: Concepts, Scheduling Criteria, Scheduling Algorithms, Thread Scheduling, Real-Time CPU Scheduling, Algorithm Evaluation.

Deadlocks: System Model, Deadlock Characterization, Methods for Handling Deadlocks, Deadlock Prevention, Deadlock Avoidance, Deadlock Detection, Recovery from Deadlock.

## Unit – III

Memory Management: Main Memory, Swapping, Contiguous Memory Allocation, Segmentation, Paging, Structure of the Page Table. Virtual Memory: Demand Paging, Page Replacement, Allocation of Frames, Thrashing, Memory-Mapped Files.

Mass-Storage Structure: Overview, Disk Structure, Disk Scheduling, Disk Management, Swap-Space Management, RAID Structure, Stable-Storage Implementation.

## Unit – IV

File Systems: File Concept, Access Methods, Directory and Disk Structure, File-System Mounting, Protection. File-System Structure and Implementation, Directory Implementation, Allocation Methods, Free-Space Management, Recovery, Network File System.

Protection and Security: Goals of Protection, Principles of Protection, Domain of Protection,

Access Matrix, Access Control, Revocation of Access Rights, The Security Problem, Program Threats, System and Network Threats, Cryptography as a Security Tool, User Authentication, Implementing Security Defenses, Firewalling to Protect Systems and Networks, Computer-Security Classifications. Case Study: Windows 7 and Linux System.

**Text** Abraham Silberschatz, Peter Baer Galvin, Greg Gagne, *Operating System Concepts* (9e)

**References**

1. Thomas W. Doeppner, *Operating systems in depth*
2. Andrew S. Tanenbaum, *Modern Operating Systems*
3. William Stallings, *Operating Systems – Internals and Design Principles*
4. Dhananjay M. Dhandhere, *Operating Systems-A Concept Based Approach*

# CS103T Software Engineering

Theory: 4 Hours/Week Credits: 4

## Unit – I

Software Engineering: The Nature of Software, Changing Nature of Software, Defining the Discipline, Software Process, Software Engineering Practice.

The Software Process: A Generic Process Model, Defining a Framework Activity, Process Assessment and Improvement, Prescriptive Process Models, Specialized Process Models, Unified Process, Personal and Team Process Models. Defining Agility, Agile Process, Extreme Programming, Psychology of Software Engineering, Software Team Structures, Software Engineering Using the Cloud, Global Teams.

## Unit – II

Requirements: Core Principles of Modeling, Requirements Engineering, Establishing the Groundwork, Eliciting Requirements, Developing Use Cases, Building the Analysis Model, Requirements Analysis, UML Models That Supplement the Use Case, Identifying Analysis Classes, Specifying Attributes, Defining Operations, Class-Responsibility-Collaborator Modeling, Associations and Dependencies, Analysis Packages.

Design Concepts: Design within the Context of SE, Design Process, Design Concepts, Design Model, Software Architecture, Architectural Styles, Architectural Considerations, Architectural Design, Component, Designing Class-Based Components, Conducting Component-Level Design, Component-Based Development, User Interface Design Rules.

## Unit – III

Quality Management: Quality, Software Quality, Software Quality Dilemma, Achieving Software Quality, Defect Amplification and Removal, Reviews, Informal Reviews, Formal Technical Reviews, Elements of Software Quality Assurance, SQA Tasks, Goals, and Metrics, Software Reliability, A Strategic Approach to Software Testing, Test Validation Testing, System Testing, Debugging, Software Testing Fundamentals, White-Box Testing, Black-Box Testing, Path Testing, Control Structure Testing, Object-Oriented Testing Strategies& Methods, Security Engineering Analysis, Security Assurance, Security Risk Analysis.

## Unit – IV

Software Configuration Management, SCM Process, Product Metrics for Requirements Model, Design Model, Source Code, Testing and Maintenance.

Managing Software Projects: The Project Management Spectrum, $W^5HH$ Principle, Metrics in the Process and Project Domains, Software Measurement, Metrics for Software Quality, Integrating Metrics within the Software Process, Software Project Estimation, Decomposition Techniques, Project Scheduling – basics, scheduling, Software Risks, Risk Mitigation, Monitoring, and Management, Software Maintenance, Software Reengineering, Reverse Engineering, Forward Engineering.

**Text** Roger S Pressman, B R Maxim, *Software Engineering – A Practitioner's Approach (8e)*

**References**

1. Ian Sommerville, *Software Engineering*
2. Hans Van Vliet, *Software Engineering*
3. D. Bell, *Software Engineering for Students*
4. K.K. Aggarwal, Y. Singh, *Software Engineering*
5. R. Mall, *Fundamentals of Software Engineering*
6. Pankaj Jalote, An Integrated *Approach to Software Engineering*

# CS104T Discrete Mathematics

Theory: 4 Hours/Week Credits: 4

## Unit – I

Mathematical Logic: propositional logic, propositional equivalences, predicates & quantifiers, rule of inference, direct proofs, proof by contraposition, proof by contradiction.
Boolean Algebra: Boolean functions and its representation, logic gates, minimizations of circuits by using Boolean identities and K-map.

## Unit – II

Basic Structures: Sets representations, set operations, functions, sequences and summations.
Division algorithm, modular arithmetic, solving congruences, applications of congruences.
Recursion: Proofs by mathematical induction, recursive definitions, structural induction, generalized induction, recursive algorithms.

## Unit – III

Counting: Basic counting principle, inclusion-exclusion for two-sets, pigeonhole principle, permutations and combinations, Binomial coefficient and identities, generalized permutations and combinations.
Recurrence Relations: introduction, solving linear recurrence relations, generating functions, principle of inclusion-exclusion, applications of inclusion-exclusion.
Relations: relations and their properties, representing relations, closures of relations, equivalence relations, partial orderings.

## Unit – IV

Graphs: Graphs definitions, graph terminology, types of graphs, representing graphs, graph isomorphism, connectivity of graphs, Euler and Hamilton paths and circuits, Dijkstra's algorithm to find shortest path, planar graphs–Euler's formula and its applications, graph coloring and its applications
Trees: Trees definitions–properties of trees, applications of trees–BST, Haffman Coding, tree traversals: pre-order, in-order, post-order, prefix, infix, postfix notations, spanning tress–DFS, BFS, Prim's, Kruskal's algorithms.

**Text** Kenneth H. Rosen, *Discrete Mathematics and its Applications* (7e)

**References**

1. Ralph P. Grimaldi, *Discrete and Combinatorial Mathematics*
2. Stein, Drysdale, Bogart, *Discrete Mathematics for Computer Scientists*
3. J.P. Tremblay, R. Manohar, *Discrete Mathematical Structures with Applications to Computer Science*
4. Joe L. Mott, Abraham Kandel, Theoder P. Baker, *Discrete Mathematics for Computer Scientists and Mathematicians*

# CS105P Advanced Java Lab

Practical: 6 Hours/Week Credits: 3

1. Create GUI to present a set of choices for a user to select stationary products and display the price of Product after selection from the list.
2. Create GUI to demonstrate typical Editable Table which describing Employee for a software company.
3. Create GUI to demonstrate swing components using student registration form.
4. Create a Remote Object for simple arithmetic operators. Use AWT/SWING to create user interface.
5. Write an RMI application using call back mechanism
6. Develop Servlet Question-Answer Application using HttpServletRequest and HttpServletRequest interfaces.
7. Develop Servlet application to accept HTNO of a student from client and display the memorandum of marks from the server
8. JSP Programs
   a. Create a JSP page that prints temperature conversion (from Celsius to Fahrenheit) chart
   b. Create a JSP page to print current date and time
   c. Create a JSP page to print number of times page is referred after the page is loaded.
9. Write a simple JSP application to demonstrate the use of implicit object (at least 5).
10. Develop a Hibernate application to Store Feedback of Website Visitors in MySQL Database.
11. Develop a JSP Application to accept Registration Details from the user and store database table.
12. Develop a JSP Application to Authenticate User Login as per the Registration Details. If Login Success then forward User to Index Page otherwise show Login failure Message.
13. Develop a web Application to add items in the inventory using JSF.
14. Write EJB applications using stateless session beans and state-full session beans.
15. Develop a Room Reservation System Application using Entity Beans.
16. Create Three-tire application using Servlets, JSP, EJB.

# CS106P Operating Systems Lab

Practical: 6 Hours/Week Credits: 3

1. Write shell programs using ‘case’, ‘then’ and ‘if’ & ’else’ statements.
2. Write shell programs using while, do-while and for loop statements.
3. Write a program to create a child process using fork(), exec() system calls and use other system calls.
4. Write a program to convert upper case to lower case letters of a given ASCII file.
5. Write a program to program to search the given pattern in a file.
6. Write a program to implementation of Signals in UNIX.
7. Write a program to simulate UNIX commands like ls, grep, cp.
8. Write a program to demonstrate FCFS and SJF process schedules on the given data.
9. Write a program to demonstrate CPU Priority and Round Robin Scheduling on the given burst time and arrival times.
10. Write a program to simulate Inter Process Communication using pipes.
11. Write a program to implementing Producer and Consumer problem using Semaphores.
12. Write a program to simulate Bankers Algorithm for Dead Lock Avoidance
13. Write a program to simulate Bankers Algorithm Dead Lock Prevention.
14. Write a program to simulate Paging Techniques of memory management.
15. Write a program to simulate FIFO, LRU, LFU Page replacement algorithms.
16. Write a program to simulate Sequential, Indexed, and Linked file allocation strategies.

## CS107P Software Engineering Lab

Practical: 4 Hours/Week Credits: 2

1. Study of case tool
   Requirements
2. Implementation of requirements engineering activities such as elicitation, validation, management using case tools
   Analysis and Design
3. Implementation of Analysis and design using case tools
4. Study and usage of software project management tools such cost estimates and scheduling
5. Documentation generators –Study and practice of Documentation generators
6. Data Modeling using automated tools
7. Practice reverse engineering and re engineering using tools
8. Exposure towards test plan generators, test case generators, test coverage and software metrics.
9. Meta modeling and software life cycle management.
   Case Studies:
10. Structure charts, Data Flow Diagrams, Decision tables and ER diagrams for
    a. Banking System
    b. Railway Reservation System
    c. Hotel management system
    d. Inventory Control System
    e. Library management system

**Note:** The teacher should define the boundaries for the above case study problems and make the practice of problems mentioned.

# CS201T Programming in Python

Theory: 4 Hours/Week Credits: 4

## Unit – I

Introduction to Python Programming: How a Program Works, Using Python, Program Development Cycle, Input, Processing, and Output, Displaying Output with the Print Function, Comments, Variables, Reading Input from the Keyboard, Performing Calculations (Operators. Type conversions, Expressions), More about Data Output.
Decision Structures and Boolean Logic: if, if-else, if-elif-else Statements, Nested Decision Structures, Comparing Strings, Logical Operators, Boolean Variables.
Repetition Structures: Introduction, while loop, for loop, Calculating a Running Total, Input Validation Loops, Nested Loops.

## Unit – II

Functions: Introduction, Defining and Calling a Void Function, Designing a Program to Use Functions, Local Variables, Passing Arguments to Functions, Global Variables and Global Constants, Value-Returning Functions-Generating Random Numbers, Writing Our Own Value-Returning Functions, The math Module, Storing Functions in Modules.
File and Exceptions: Introduction to File Input and Output, Using Loops to Process Files, Processing Records, Exceptions.

## Unit – III

Lists and Tuples: Sequences, Introduction to Lists, List slicing, Finding Items in Lists with the in Operator, List Methods and Useful Built-in Functions, Copying Lists, Processing Lists, Two-Dimensional Lists, Tuples.
Strings: Basic String Operations, String Slicing, Testing, Searching, and Manipulating Strings.
Dictionaries and Sets: Dictionaries, Sets, Serializing Objects.
Recursion: Introduction, Problem Solving with Recursion, Examples of Recursive Algorithms.

## Unit – IV

Object-Oriented Programming: Procedural and Object-Oriented Programming, Classes, Working with Instances, Techniques for Designing Classes, Inheritance, Polymorphism.
GUI Programming: Graphical User Interfaces, Using the tkinter Module, Display text with Label Widgets, Organizing Widgets with Frames, Button Widgets and Info Dialog Boxes, Getting Input with Entry Widget, Using Labels as Output Fields, Radio Buttons, Check Buttons.

**Text** Tony Gaddis, *Starting Out With Python (3e)*

**References**

1. Kenneth A. Lambert, *Fundamentals of Python*
2. Clinton W. Brownley, *Foundations for Analytics with Python*
3. James Payne, *Beginning Python using Python 2.6 and Python 3*
4. Charles Dierach, *Introduction to Computer Science using Python*
5. Paul Gries, *Practical Programming: An Introduction to Computer Science using Python 3*

# CS202T Computer Networks

Theory: 4 Hours/Week Credits: 4

## Unit – I

Computer Networks Fundamentals: Overview, Network Hardware, Network Software, Reference models– OSI Model, TCP/IP Reference Model, Comparison of OSI and TCP/IP Reference Model, Example Networks, Network Standardization.

Physical Layer: Guided Transmission Media, Wireless Transmission, Multiplexing, Switching.

Data Link Layer: Design Issues, Error Detection and Correction, Data Link Layer Protocols, Sliding Window Protocol

## Unit – II

Multiple Access Sublayer: ALOHA, CSMA, Collision Free Protocols, Ethernet, Wireless LAN-802.11, Data Link Layer Switching–Repeaters, Hubs, Bridges, Switches, Routers, Gateways.

Network Layer: Design Issues, Routing Algorithms – Shortest path, Flooding, Distance Vector Routing, Link state Routing, Hierarchical, Broadcast Routing, Multicast Routing; Congestion Control Algorithms.

## Unit – III

Internetworking: Tunneling, Internetwork Routing, Fragmentation, IPv4 Vs IPv6Protocol, IP Addresses, CIDR, Internet Control Protocols–IMCP, ARP, RARP, DHCP.

Transport Layer: Services provided to the upper layers, Transport Protocols, Overview of Congestion Control.

## Unit – IV

The Internet Transport Protocols: Introduction to UDP&RPC, Real Time Transport Protocols, The Internet Transport Protocols–TCP, TCP Service Model, TCP protocol, TCP Segment Header, TCP Connection Establishment, TCP Connection Release, Modeling TCP Connection Management, TCP Sliding Window, TCP Time Management, TCP Congestion Control.

Application Layer: DNS, TELNET, E-Mail, FTP, HTTP, SSH, Overview of WWW.

**Text** Andrew S. Tanenbaum, David J Wetherall, *Computer Networks (5e)*

**References**

1. William Stallings, *Data and Computer Communications*
2. Behrouz A. Forouzan, *Data Communication and Networking*
3. Behrouz A Forouzan, Firouz Mosharraf, *Computer Networks A Top-Down Approach*

# CS203T Design and Analysis of Algorithms

Theory: 4 Hours/Week Credits: 4

## Unit – I

Introduction: Algorithm, Fundamentals of Algorithmic Problem Solving, Important Problem Types. Fundamentals of the Analysis of Algorithm: The Analysis Framework, Asymptotic Notations and Basic Efficiency Classes, Mathematical Analysis of Non-recursive & Recursive Algorithms. Brute Force Search: Selection Sort, Bubble Sort, Sequential Search, Brute-Force String Matching, Exhaustive Search, Depth-First Search, Breadth-First Search.

## Unit – II

Decrease–&–Conquer: Insertion Sort, Topological Sorting, Binary Search, Interpolation Search
Divide-and-Conquer: Merge Sort, Quick Sort, Multiplication of Large Integers, Strassen's Matrix Multiplication, Transform-and-Conquer: Presorting, Balanced Search Trees, Heaps and Heap Sort, Problem Reduction. Space and Time Trade-Offs: Hashing, B-Trees.

## Unit – III

Dynamic Programming: Knapsack Problem, Optimal Binary Search Trees, Warshall's and Floyd's Algorithms. Greedy Technique: Prim's Algorithm, Kruskal's Algorithm, Dijkstra's Algorithm, Huffman Trees and Codes. Iterative Improvement: Simplex Method, Maximum-Flow Problem.

## Unit – IV

Limitations of Algorithm Power: Lower-Bound Arguments, Decision Trees, P, NP, and NP-Complete Problems. Backtracking: n-Queens Problem, Hamiltonian Circuit Problem, Subset-Sum Problem, Branch-and-Bound: Assignment Problem, Knapsack Problem, Traveling Salesman Problem, Approximation Algorithms for the Knapsack Problem.

**Text** Anany Levitin, *Introduction to the Design and Analysis of Algorithms (3e)*

**References**

1. Richard Neapolitan, *Foundations of Algorithms*
2. Thomas H. Cormen, *Introduction to Algorithms*
3. E. Horowitz, S. Sahni, *Fundamentals of Computer Algorithms*
4. A.V. Aho, J.V. Hopcroft, J.D. Ullmann, *The Design and Analysis of Computer Algorithms*
5. Donald E Knuth, *The Art of Programming_ Volumes–1, 2, 3, 4*

# CS204T Automata Theory

Theory: 4 Hours/Week Credits: 4

## Unit – I

Fundamentals – alphabets, strings, languages, problems, graphs, trees, Finite State Systems, definitions, Finite Automaton model, acceptance of strings, and languages, Deterministic finite automaton and Nondeterministic finite automaton, transition diagrams, transition tables, proliferation trees and language recognizers, equivalence of DFA's and NFA's.

Finite Automata with $\varepsilon$-moves, significance, acceptance of languages, $\varepsilon$-closure, Equivalence of NFA's with and without $\varepsilon$-moves, Minimization of finite automata, Two-way finite automata, Finite Automata with output– Moore and Melay machines.

## Unit – II

Regular Languages: regular sets, regular expressions, identity rules, constructing finite automata for a given regular expressions, conversion of finite automata to regular expressions. Pumping lemma of regular sets and its applications, closure properties of regular sets.

Grammar Formalism: Regular grammars–right linear and left linear grammars, equivalence between regular linear grammar and finite automata, inter conversion, Context free grammar, derivation trees, sentential forms, right most and leftmost derivation of strings, ambiguity.

## Unit – III

Context Free Grammars: Simplification of Context Free Grammars, Chomsky normal form, Greiback normal form, Pumping lemma for context free languages and its applications, closure of properties of CFL (proofs omitted).

Push Down Automata: PDA definition, model, acceptance of CFL, acceptance by final state and acceptance by empty state and its equivalence. Equivalence of PDA's and CFL's, inter-conversion. (Proofs not required).

## Unit – IV

Membership Algorithm (CYK Algorithm) for Context Free Grammars.

Turing Machine: TM definition, model, design of TM, computable functions, unrestricted grammars, recursively enumerable languages. Church's hypothesis, counter machine, types of Turing machines (proofs omitted). Linear bounded automata and Context sensitive language.

Computability Theory: Chomsky hierarchy of languages, Introduction to DCFL, DPDA, LR(0) grammar, decidability and undecidable problems. Definitions of P and NP problems, NP complete and NP hard problems.

**Text** J. E. Hopcroft, J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*

**References**

1. Mishra, Chandrashekaran, *Theory of Computer Science*
2. ZviKohav, Niraj K Jha, *Switching and Finite Automata Theory*
3. Perter Linz, *An Introduction to Formal Languages and Automata*
4. John C. Martin, *Introduction to Languages and the Theory of Computation*

# CS205P Python Lab

Practical: 6 Hours/Week Credits: 3

1. Write a program that displays the following information: Your name, Full address, Mobile number, College name, Course subjects.
2. Write a program to find the largest three integers using if-else and conditional operator.
3. Write a program that asks the user to enter a series of positive numbers (The user should enter a negative number to signal the end of the series) and the program should display the numbers in order and their sum.
4. Write a program to find the product of two matrices [A]mxp and [B]pxr
5. Write recursive and non-recursive functions for the following:
    a. To find GCD of two integers.
    b. To find the factorial of positive integer
    c. To print Fibonacci Sequence up to given number n
6. Write a program to display two random numbers that are to be added, such as: 247 + 129, the program should allow the student to enter the answer. If the answer is correct, a message of congratulations should be displayed. If the answer is incorrect, a message showing the correct answer should be displayed.
7. Write recursive and non-recursive functions to display prime number from 2 to n.
8. Write a program that writes a series of random numbers to a file from 1 to n and display.
9. Write a program to create file, write the content and display the contents of the file with each line preceded with a line number (start with 1) followed by a colon.
10. In a program, write a function that accepts two arguments: a list and a number n. The function displays all of the numbers in the list that are greater than the number n.
11. Write a program with a function that accepts a string as an argument and returns the no. of vowels that the string contains. Another function to return number of consonants.
12. Write a program that opens a specified text file and then displays a list of all the unique words found in the file. (Store each word as an element of a set.)
13. Write a program to analyze the contents of two text files using set operations.
14. Write a program to implement the inheritance and dynamic polymorphism.
15. Write a GUI program that converts Celsius temperatures to Fahrenheit temperatures.
16. Write a GUI program that displays your details when a button is clicked.

**Note:** Handle the Exceptions raised from File Operations.

## CS206P Computer Networks Lab

Practical: 6 Hours/Week Credits: 3

1. Program to identify the category of the IP address for the given IP address
2. Program to implement sliding window protocol
3. Program for Socket pair system call usage in IPC
4. Program for Socket options using signals
5. Program to implement Echo concurrent Stream Server
6. Program to implement Echo concurrent stream client
7. Program to implement Listener and Talker
8. Program to implement TCP time service
9. Program to implement UDP time service
10. Program to implement Ping service
11. Program to implement Route tracing program
12. Program to implement File Transfer Protocol
13. Program to implement any Shortest path routing Algorithm
14. Program to implement Distance Vector Routing Implementation
15. Program to implement ICMP Error Message simulations
16. Program to implement Reverse Address Resolution Protocol

# CS207P Design and Analysis of Algorithms Lab

Practical: 4 Hours/Week Credits: 2

1. Write a program recursive and non-recursive function for the following:
   a) Factorial of an integer b) GCD of two integers c) Fibonacci Sequence
2. Write a program for sorting the given list using Insertion Sort, Topological Sort.
3. Write a program for sorting the given list using Selection Sort, BubbleSort.
4. Write a program for sorting the given list using Merge Sort.
5. Write a program for sorting the given list using Quick Sort.
6. Write a program for sorting the given list using Heap Sort.
7. Write a program to find the given number in a list using Sequential Search, Binary Search.
8. Write a program to find product of two matrices [A]mxp and [B]pxr
9. Write a program to create AVL tree.
10. Write a program to create B-tree.
11. Write a program to find the Euler circuit and the Hamiltonian circuit for a weighted graph.
12. Write a program to find the shortest path in a weighted graph using Dijkstra's Algorithm.
13. Write a program to solve travelling sales man problem.
14. Write a program to solve knapsack problem.
15. Write a program to find the minimum spanning tree for a weighted graph using Kruskal's Algorithm.
16. Write a program to find the minimum spanning tree for a weighted graph using Prim's Algorithm.

**Note:** Analyze all the above problems with respect to Time Complexity.

## MOOCs (Massive Online Open Courses) Free Resources

### E-Learning:

- NPTEL :nptel.ac.in [Core Subjects Certification]
- C++ INSTITUTE :cppinstitute.org [C++ Certification]
- ORACLEEDUCATION :education.oracle.com [Java, DBMS Certification]
- BIG DATA UNIVERSITY :bigdatauniversity.com [Big Data Certification]
- COURSERA :coursera.org [Core Subjects Certification]
- CODEACADEMY :codecademy.com [Coding Certification]
- KHANACADEMY :khanacademy.org [Core Subjects Certification]
- PIXAR IN A BOX :khanacademy.org/partner-content/pixar
- VIDEOLECTURES :videolectures.net
- YOUTUBEEDU :plus.google.com/+YouTubeEDU/posts
- DISNEY RESEARCH :disneyresearch.com
- ALISON :alison.com [Core Subjects Certification]
- INTERNET ARCHIVE :archive.org

### Freeware:

- SCILAB : scilab.org [MatLab Equivalent]
- GEOGEBRA :geogebra.org [Software for Class Room Teaching]

### Search Engine:

- WOLFRAM ALPHA :wolframalpha.com [Computing Engine]
- CITESEER :citeseerx.ist.psu.edu [Searching Research Articles]
- DOAJ :doaj.org [Open Access to Journals]