
B. E. SYLLABUS

INFORMATION SCIENCE & ENGINEERING

III & IV SEMESTER

With
Scheme of Teaching
& Examination
## DEPARTMENT: INFORMATION SCIENCE & ENGG

<table>
<thead>
<tr>
<th></th>
<th>Name</th>
<th>Degree</th>
<th>Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dr. Balasubramani R.</td>
<td>Ph.D.</td>
<td>Professor &amp; HOD</td>
</tr>
<tr>
<td>2</td>
<td>Dr. Uday Kumar Shenoy</td>
<td>Ph.D.</td>
<td>Professor</td>
</tr>
<tr>
<td>3</td>
<td>Mr. KarthikPai B. H.</td>
<td>M.Tech</td>
<td>Associate Professor II</td>
</tr>
<tr>
<td>4</td>
<td>Mrs. Ashwini B.</td>
<td>M.Tech</td>
<td>Associate Professor II</td>
</tr>
<tr>
<td>5</td>
<td>Mr. VasudevaPai</td>
<td>M.Tech</td>
<td>Assistant Professor II</td>
</tr>
<tr>
<td>6</td>
<td>Mr. Pranesh V. K.</td>
<td>M.Tech</td>
<td>Assistant Professor II</td>
</tr>
<tr>
<td>7</td>
<td>Mrs. Deepa J. Shetty</td>
<td>M.Tech</td>
<td>Assistant Professor II</td>
</tr>
<tr>
<td>8</td>
<td>Mr. Devidas</td>
<td>M.Tech</td>
<td>Assistant Professor II</td>
</tr>
<tr>
<td>9</td>
<td>Mrs. Rashmi Naveen</td>
<td>M.Tech</td>
<td>Assistant Professor II</td>
</tr>
<tr>
<td>10</td>
<td>Mr. Rakesh Joshi U.</td>
<td>M.Tech</td>
<td>Assistant Professor II</td>
</tr>
<tr>
<td>11</td>
<td>Mrs. Manasa S.</td>
<td>M.Tech</td>
<td>Assistant Professor II</td>
</tr>
<tr>
<td>12</td>
<td>Mr. Jason Elroy Martis</td>
<td>M.Tech</td>
<td>Assistant Professor II</td>
</tr>
<tr>
<td>13</td>
<td>Ms. Chinmai Shetty</td>
<td>M.Tech</td>
<td>Assistant Professor II</td>
</tr>
<tr>
<td>14</td>
<td>Mr. AbhirBhandary</td>
<td>M.Tech</td>
<td>Assistant Professor I</td>
</tr>
<tr>
<td>15</td>
<td>Mr. SrikanthBhat K.</td>
<td>M.Tech</td>
<td>Assistant Professor I</td>
</tr>
</tbody>
</table>
Vision:
The vision of the Dept. of ISE is to uniquely position the Dept. as a leader in innovation and excellence in information science and engineering through education, research and scholarship in a professional framework by addressing evolving global needs. Also the Dept. aims at creating top quality successful and sustainable programs and curricula for the students to address the emerging educational challenges and market demands.

Mission:
- To provide outstanding education and research training to the students for their productive careers in industry, academia and government.
- To provide a learning environment that promotes excellence and innovation, ethical practice and responsibility towards society.
- To prepare the students to practice their professions competently to meet the ever-changing needs of society and to continue learning their discipline, allowing them to move into other related fields.
- To promote active learning, critical thinking, and engineering judgment coupled with business and entrepreneurial skills.
Program: B.E. Information Science & Engineering

Programme Educational Objectives (PEOs) :
1. Graduates must gain both theoretical and practical knowledge to identify, formulate & solve challenges in Information Science & Engineering problems.
2. Graduates must work productively as Information Science Engineers, including supportive and leadership roles on multidisciplinary teams.
3. Graduates must communicate effectively, recognize and incorporate societal needs and constraints in their professional endeavors, and practice their profession with high regard to legal and ethical responsibilities.
4. Graduates must engage in life-long learning, such as graduate study, to remain current in their profession and be leaders in our technological society.

Programme Outcomes (POs)
BE (ISE) Engineering Program students must attain the following outcomes at the end of the course.
1. An ability to apply knowledge of mathematics, science and engineering
2. An ability to design and conduct experiments, as well as to analyze and interpret data
3. An ability to design a system, component or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability and sustainability
4. An ability to function on multidisciplinary teams
5. An ability to identify, formulate and solve engineering problems
6. An understanding of professional and ethical responsibility
7. An ability to communicate effectively
8. The broad education necessary to understand the impact of engineering solutions in global, economic environmental and societal context
9. A recognition of the need for and an ability to engage in lifelong learning
10. A knowledge of contemporary issues
11. An ability to use the techniques, skills and modern engineering tools necessary for engineering practice and
12. An ability to strengthen the knowledge and understanding in the areas of computer networking and software engineering.
### DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING
### SCHEME OF TEACHING AND EXAMINATION

#### III SEMESTER B.E.

<table>
<thead>
<tr>
<th>Sl. No.</th>
<th>Code</th>
<th>Subject</th>
<th>Theory/Tuto./Prac./ Self Study</th>
<th>Total Hrs./Week</th>
<th>C.I.E.</th>
<th>S.E.E.</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14IS301</td>
<td>Generating Functions and Transform Techniques</td>
<td>4+0+0+0</td>
<td>4</td>
<td>50</td>
<td>50</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>14IS302</td>
<td>Digital Design</td>
<td>3+2+0+0</td>
<td>5</td>
<td>50</td>
<td>50</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>14IS303</td>
<td>Discrete Mathematical Structures and Number Theory</td>
<td>4+0+0+0</td>
<td>4</td>
<td>50</td>
<td>50</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>14IS304</td>
<td>Computer Organization and Architecture</td>
<td>4+0+0+S</td>
<td>4</td>
<td>50</td>
<td>50</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>14IS305</td>
<td>Data Structures</td>
<td>3+2+0+0</td>
<td>5</td>
<td>50</td>
<td>50</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>14IS306</td>
<td>Object Oriented Programming with C++</td>
<td>3+0+0+0</td>
<td>3</td>
<td>50</td>
<td>50</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>14IS307</td>
<td>Digital Design Lab</td>
<td>0+0+2+0</td>
<td>2</td>
<td>50</td>
<td>50</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>14IS308</td>
<td>Data Structures Lab</td>
<td>0+0+2+0</td>
<td>2</td>
<td>50</td>
<td>50</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>14IS309</td>
<td>Object Oriented Programming with C++ Lab</td>
<td>0+0+2+0</td>
<td>2</td>
<td>50</td>
<td>50</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>TOTAL</strong></td>
<td></td>
<td><strong>31</strong></td>
<td><strong>450</strong></td>
<td><strong>450</strong></td>
<td><strong>26</strong></td>
</tr>
</tbody>
</table>
# IV Semester B.E.

<table>
<thead>
<tr>
<th>Sl. No.</th>
<th>Code</th>
<th>Subject</th>
<th>Theory/Tuto./Prac./ Self Study</th>
<th>Total Hrs./Week</th>
<th>C.I.E.</th>
<th>S.E.E.</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14IS401</td>
<td>Probability Theory and Numerical Methods</td>
<td>4+0+0+0</td>
<td>4</td>
<td>50</td>
<td>50</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>14IS402</td>
<td>Analysis and Design of Algorithms</td>
<td>3+2+0+0</td>
<td>5</td>
<td>50</td>
<td>50</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>14IS403</td>
<td>Finite Automata and Formal Languages</td>
<td>4+0+0+0</td>
<td>4</td>
<td>50</td>
<td>50</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>14IS404</td>
<td>Data Communication</td>
<td>3+0+0+0</td>
<td>3</td>
<td>50</td>
<td>50</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>14IS405</td>
<td>Software Engineering</td>
<td>4+0+0+S</td>
<td>4</td>
<td>50</td>
<td>50</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>14IS406</td>
<td>Unix and Shell Programming</td>
<td>3+0+0+0</td>
<td>3</td>
<td>50</td>
<td>50</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>14IS407</td>
<td>Soft Skills (IEL)</td>
<td>0+4+0+0</td>
<td>4</td>
<td>50</td>
<td>50</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>14IS408</td>
<td>Analysis and Design of Algorithms Lab</td>
<td>0+0+2+0</td>
<td>2</td>
<td>50</td>
<td>50</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>14IS409</td>
<td>Unix and Shell Programming Lab</td>
<td>0+0+2+0</td>
<td>2</td>
<td>50</td>
<td>50</td>
<td>1</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td></td>
<td><strong>32</strong></td>
<td><strong>32</strong></td>
<td><strong>450</strong></td>
<td><strong>450</strong></td>
<td></td>
<td><strong>26</strong></td>
</tr>
</tbody>
</table>
GENERATING FUNCTIONS AND TRANSFORM TECHNIQUES

Sub Code: 14IS301  Credits : 04
Hrs/Week: 4+0+0+0  Total Hours : 52

Course Outcomes:
At the end of the course the successful students will be able to:
1. Apply generating functions, permutations and combinations to solve problems on real life situations.
2. Formulate problems in terms of mathematical equations from the engineering/physical problems.
3. Develop a thorough understanding of Fourier series and Fourier transform so that it can be applied to engineering problems.
4. Apply graph theoretic approach or algorithm to engineering problems.

UNIT – I

Introduction to Graph Theory: Definitions and examples, Subgraphs, Complements, Graph Isomorphism, Euler Trails and Circuits, Hamiltonian paths and Cycles, Planar Graphs, Graph Colouring, Trees: Definitions, Properties and Examples.

10 Hours

UNIT - II

Fourier Analysis: Periodic functions, Euler’s formulae, Fourier series of odd and even functions, functions with arbitrary period, half range series. Harmonic Analysis. Fourier integral theorem, Fourier Transforms, Inverse Fourier transform, Convolution theorem and Parseval’s identity. Fourier sine and Fourier cosine transforms, Inverse Fourier sine and Inverse Fourier cosine transforms (simple problems) 12 Hours
UNIT - III

**Z transforms:** Z-transform, standard forms, linearity property, damping rule, shifting rule. Inverse Z-transform, Finite differences and difference equations, Solving Difference equations using Z-transforms.

10 Hours

UNIT - IV

Combinations with Repetition, Non-negative integer solution for linear equation, positive integer solution for linear equation. Fibonacci numbers, Catalan Numbers, The Principle of Inclusion and Exclusion, Generalization of the Principle, Derangements – Nothing is in its Right place, Rook Polynomials.

10 Hours

UNIT - V

Generating Functions: Definition and Examples – Calculational Techniques, Partitions of Integers, the Exponential Generating Function, and the Summation Operator. The method of Generating Functions to solve Recurrence relations.

10 Hours

TEXT BOOKS:
3. Narsing Deo, Graph theory
4. Graham knuth & Patashnik, Concrete Mathematics

REFERENCE BOOKS:
2. Harary, Graph theory, Narosa Publishing House, 1988
DIGITAL DESIGN

Sub Code: 14IS302
Credits : 04
Hrs/Week: 3+2+0+0
Total Hours : 52

Course Outcomes:
1. Comprehend and analyze basic logic gates and the digital logic circuit design concepts.
2. Understand combinational logic and logic design minimization techniques.
3. Remember CPU design and compare the alternatives
4. Apply the sequential logic.
5. Implement and design the sequential circuits.
6. Understand the register and counter operations.
7. Apply different methods digital conversion techniques.
8. Understand the Verilog HDL.
9. Evaluate Input output model and test bench.
10. Create the implementations of different models in HDL.

UNIT - I

Chapter 1: Digital Logic
Overview of Basic Gates and Universal Logic Gates, AND-OR-Invert Gates, Positive and Negative Logic

Chapter 2: Combinational Logic Circuits
Boolean Laws and Theorems, Sum of Products Method, Truth Table to Karnaugh Map, Pairs, Quads, and Octets, Karnaugh simplifications, Don’t Care Conditions, Product-of-sums method, Product-of-sums simplification, Simplification by Quine-McClusky Method, Hazards and Hazard cover.

8 Hours

UNIT - II

Chapter 3: Data-Processing Circuits
Multiplexers, Demultiplexers, 1-of-16 Decoder, BCD-to-Decimal Decoders, Seven segment Decoders, Encoders, EX-OR gates, Parity Generators and Checkers, Magnitude Comparator, Read-Only-Memory, Programmable Array Logic, Programmable Logic, Troubleshooting with a Logic Probe.

**Chapter 4: Arithmetic Circuits**
Binary Addition, Binary Subtraction, Unsigned Binary Numbers, Sign-Magnitude Numbers, 2’s Complement Arithmetic, Arithmetic Building Blocks, The Adder-Subtractor, Fast Adder, Arithmetic Logic Unit, Binary Multiplication and Division..  **8 Hours**

**UNIT - III**

**Chapter 5: Flip-Flops:**
RS Flip-flops, Gated Flip-flops, Edge-triggered RS, D, JK Flip-flops, Flip-flop timing, JK Master-slave Flip-flops, Switch Contact Bounce Circuits, Various Representations of Flip-flops, Analysis of Sequential Circuits, Conversion of Flip-flops – a synthesis example. Design of Sequential Circuit: Design and Analysis of Synchronous and Asynchronous Sequential Circuits with examples..  **8 Hours**

**UNIT - IV**

**Chapter 6: Registers**
Types of Registers, Serial In – Serial Out, Serial In – Parallel Out, Parallel In- Serial Out, Parallel In – Parallel Out, Applications of Shift Register.

**Chapter 7: Counters**
Asynchronous Counters, Decoding Gates, Synchronous Counters, Changing the Counter Modulus, Decade Counters, Pre-settable Counters, Counter Design as a Synthesis Problem..  **8 Hours**

**UNIT - V**

**Chapter 8: D/A Conversion and A/D Conversion**
Variable Resistor Networks, Binary Ladders, D/A Converters, D/A Accuracy and Resolution, A/D Converter – Simultaneous Conversion,


**HDL Implementations**
Introduction to Verilog HDL, specify input output, module body, prepare test bench, Various HDL Implementation models, HDL Implementations of Basic gates, Multiplexers, Shift registers, counters, flip-flops.

7 Hours

**TEXT BOOK :**

**REFERENCE BOOKS:**

**********

**DISCRETE MATHEMATICAL STRUCTURES AND NUMBER THEORY**

Sub Code: 14IS303  
Credits : 04  
Hrs/Week: 4+0+0+0  
Total Hours : 52

**Course Outcomes:**
1. Understand sets, relations and functions.
2. Design and formulate recursive formula.
3. Understand posts and lattice.
4. Deduce and infer logical statements.
5. Properties of integers and prime numbers.
UNIT - I

Set Theory and Counting:
Sets and subsets, Operations on sets, The Pigeonhole principle and Recurrence Relations.

Relations and its Properties:
Product sets and Partitions, Relations and Digraphs, Paths in relations and Digraphs. 10 Hours

UNIT - II

Relations and its Properties Contd:
Properties of relations, Equivalence relations, Computer representation of Relations and Digraphs and Transitive closure and Warshall’s algorithm.

Functions:
Definition, Types of functions, Invertible functions, Functions for computer science and Permutation functions. 10 Hours

UNIT - III

Order relations and Structures:
Partially Ordered Sets, Extremal elements of Partially ordered sets, Lattices, Semigroups and Groups. Semigroups, Isomorphism and homomorphism, Groups-Abelian groups. 10 Hours

UNIT - IV

Fundamentals of logic:
Propositions and Logical Operations, Conditional Statements, Biconditional statements, The laws of logic, Logical implication- Rules of inference, Quantifiers, Mathematical induction. 10 Hours

UNIT - V

Introduction to number Theory:
Prime and Relatively prime numbers, Properties of integers, Modular arithmetic, Fermat’s and Eulers Theorems, Testing for primality, Euclids
Algorithm, The Chinese Remainder Theorem. 

**12 Hours**

**TEXT BOOKS:**

**REFERENCE BOOKS:**

**********
COMPUTER ORGANIZATION AND ARCHITECTURE

Sub Code: 14IS304  
Credits : 04
Hrs/Week: 4+0+0+S*  
Total Hours : 39

*Self Study to be exercised under the supervision of course instructor and to be restricted to not more than 10% of the total teaching hours.

Course Outcomes:

1. Understand the types of operating Systems and its classifications
2. Apply the Concept of performance and its Measurement
3. Apply security and layered approach in hardware and its operation
4. Processing
5. Analyze the caching and buffering with respect to memory systems

UNIT - I


UNIT - II

Input / Output Organization: Accessing I/O Devices, Interrupts – Interrupt Hardware, Enabling and Disabling Interrupts, Handling Multiple Devices, Controlling Device Requests, Exceptions,

8 Hours

UNIT - III
Memory System: Basic Concepts, Semiconductor RAM Memories, Read Only Memories, Speed, Size, and Cost,


7 Hours

UNIT - IV


8 Hours

UNIT - V
Basic Processing Unit: Some Fundamental Concepts, Execution of a Complete Instruction, Multiple Bus Organization. 
Hard-wired Control and Micro programmed Control.  

7 Hours

TEXT BOOKS:

REFERENCE BOOKS:

**********

DATA STRUCTURES

Sub Code: 14IS305 Credits : 04
Hrs/Week: 3+2+0+0 Total Hours: 39

Course outcomes:

1. Understand & analyze various data organization.
2. To identify & implement the appropriate data structure and modify it if required for modeling a given problem and perform various operations on it.
3. Classify & Examine linear and non-linear data structures.
4. Demonstrate & Practice iterative and recursive solutions for elementary problems.
5. Formulate algorithms and programs that use data structures such as: arrays, linked lists, stacks, queues, trees.

UNIT – I

Chapter 1: Introduction And Overview
Definitions, Concepts of data structures, types, Overview of data structures.. 1Hours

Chapter 2: Pointers.
Definition and Concepts, Accessing variables through pointers, Pointers and functions, Arrays and pointers, Array of pointers, Pointer arithmetic. 2Hours

Chapter 3: Linear Data Structures – Stacks

5 Hours

UNIT – II

Chapter 4: Linear Data Structures – Queues
Introduction and Definition Representation of Queue: Array and Structure representation of queue, various queue Structures: ordinary queue, circular queue, priority queue.

5 Hours

Chapter 5: Linear Data Structures - Singly Linked lists
Memory allocation functions. Definition and concepts singly Linked List: Representation of link list in memory, Operations on singly Linked List.

UNIT – III

Chapter 6: Linear Data Structures – Circular & Doubly Linked lists

8 Hours

UNIT – IV

Chapter 7: Nonlinear Data Structures: Tree data structures 1

7 Hours
UNIT – V

Chapter 8: Nonlinear Data Structures - Tree data structures 2
Threaded Binary Trees, Heaps , Binomial Heaps ,Expression Tree:
Evaluating expression tree, Constructing expression tree from postfix

5 Hours

Hashing: Definition, collision in hashing, Resolving hash clashes by
open addressing, separate chaining.

2 Hours

TEXT BOOKS:
1. Fundamentals of Data Structures In C; 2nd Edition; Ellis
2. Data Structures and Program Design in C++, R.Kruse, C.L
3. Yedidyah Langsam Moshe J. Augenstein and Aaron M.
   Tenenbaum; Data Structures using C and C++, PHI/Pearson, 2nd
4. Data Structures Using C And C++; 2nd Edition; Langsam
   Yedidyah, Augenstein Moshe J., Tenenbaum Aaron M; Prentice-
   Hall 2009.

REFERENCE BOOKS:
1. Classic Data Structures; D.Samanta.

2. Data Structures and Program Design in C; R. Kruse etal, , Pearson
   Education Asia, Delhi-2002.
   Second Edition; Behrouz A. Forouzan and Richard F.
   Gilberg;Thomson, 2003.
OBJECT ORIENTED PROGRAMMING USING C++

Sub Code: 14IS306  
Credits : 04

Hrs/Week: 3+0+0+0  
Total Hours : 39

Course Outcomes:

1. Understand the basic concepts, its benefits and application of object oriented programming
2. Understand the C++ programming language using object oriented approach
3. Understand the C++ program paradigm and its basic syntax and constructs
4. Apply the function overloading concepts of C++ in writing programs
5. Apply the concepts of classes and objects to problems
6. Apply the concepts of constructors and destructors to problems
7. Apply the operator overloading concepts
8. Apply inheritance concept to share the properties of a class
9. Apply virtual functions and polymorphism to problems
10. Apply templates to the programs to handle different data types
11. Apply exception handling concepts to handle exceptions caused in a program
12. Apply the basic file handling operations to file related problems

UNIT-I

Chapter 1: Principles of object – oriented programming
A look at Procedure Oriented Programming, Object Oriented Programming Paradigm, Basic Concepts of OOP, Benefits of OOP, and Object oriented languages, Application of OOP.  

Chapter 2: Beginning with C++
What is C++, Applications of C++, Structure of C++ program.
Chapter 3 : Tokens, Expressions and Control Structures
Basic Data types, derived data types, user defined data types, variables in C++, dynamic initialization of variables, reference variables, operators in c++, scope resolution operator, memory management operators, type cast operators, manipulators, namespace.

3 Hours

UNIT – II

Chapter 4 : Functions in C++
Function prototyping , Infinite Functions, Default Arguments, Function Overloading.

2 Hours

Chapter 5:Classes and objects
Introduction, C Structure Revisited, Specifying a Class, Defining Member Functions, Static Data Members, and Static Member Functions. Arrays of Objects, Objects as Functions Arguments, this pointer, Friend Functions, Returning Objects, Constant Member Functions.

5 Hours

UNIT-III

Chapter 6:Constructors and Destructors
Introduction, Constructors, Parameterized Constructors, Multiple Constructors in a Class. Constructors with Default Arguments, Copy Constructors, Dynamic Constructors, Constant Objects, Destructors.

5 Hours

Chapter 7:Operator Overloading and type Conversions
Introduction, Defining Operator Overloading, overloading the Various Operators, Overloading the Increment and the Decrement Operators (Prefix and Postfix), Overloading the Unary Minus and the Unary Plus Operator, Overloading the Arithmetic Operators. Overloading the Relational Operators, Overloading the Assignment Operator, Overloading the Insertion and Extraction Operators, Rules for overloading operators. Type conversions.

5 Hours
UNIT-IV

Chapter 8: Inheritance
Introduction, Defining Derived Classes, single Inheritance, Protected Access Specifier, Multilevel Inheritance, Multiple Inheritance, Hierarchical Inheritance, Hybrid Inheritance, Virtual Base Classes, Abstract Classes, Constructors in Derived Classes, Nesting of Classes.

5 Hours

Chapter 9: Pointers, Virtual Functions and Polymorphism
Introduction, Pointers, Pointers to objects, pointers to derived classes, Virtual Functions, Pure Virtual Functions.

3 Hours

UNIT-V

Chapter 10: Templates
Introduction, Function Templates, Class Templates, Overloading of template Function.

3 Hours

Chapter 11: Exception Handling
Basics of exception Handling, Exception Handling Mechanism, Limitation of Exception Handling.

3 Hours

Chapter 12: Working with files
Classes for Files Stream Operations, Opening and Closing a File, Error Handling during File Operations.

2 Hours

TEXT BOOK:
1. E. Balaguruswamy. Object – Oriented Programming with C++ ,third Edition, Tata McGraw Hill. (Chapters 1.3 to 1.8, 2.1, 2.2, 2.6, 3.5 to 3.7, 3.10 to 3.18, 4.3, 4.6 to 4.9, 5, 6, 7, 8, 9, 11, and 12)

REFERENCE BOOKS:

DIGITAL DESIGN LAB

Sub Code: 14IS307  
Credits : 01  
Hrs/Week :0+0+2+0  
Total Hours : 13*3

Course Outcomes:

PART A (Combinational Circuits)

2. Realization of half/full adder and half/full Subtractor using logic gates.
3. Realization of Binary to Gray code conversion and vice-versa.
4. Design and implementation of
   i) Half/Full Adder using IC 74153.
   ii) Half/Full Subtractor using IC 74153.
5. To design and set up the following circuit
   I. 8:1 Multiplexer (MUX) using only 4:1 Multiplexer. (74153)
   II. 1:4 De multiplexer (DE-MUX) using only NAND gates.
6. Design and implementation of a Full adder and a Full Subtractor using 3:8 decoder and 4 input NAND gates.
7. Design 1-bit and 2-bit Comparators circuit.
8. Use decoder chip to drive LED display. (7447 & 7-segment LED display)

PART B (Sequential Circuits)

9. Design and implementation of clocked S-R flip flop using Nor gates.
10. Design and implementation of D flipflop.
11. Design and implement following applications of JK flip flop.
   i. 4-bit ripple counter
   ii. BCD counter
12. Design and implementation of a Mod-N (N<8) Synchronous up counter using J-K flip flop ICs.
13. Design and implementation of an Asynchronous counter using a Decade counter IC to count up from 0 to n (n<=9) (7490)
14. Design and implementation of the following using 4-bit shift register
   i) Ring counter
   ii) Johnson counter
15. Design and study the operation of Sequence Generator. (7495,7486)

**PART C (Simulation)**
1. Simulation of 8:1 and 4:1 multiplexer.
2. Simulation of half adder and full adder.
3. Simulation of the following
   1. Multiplexer
   2. De multiplexer
4. Simulation of the following:
   a. Ring counter
   b. Johnson counter

**********

**DATA STRUCTURES LAB**

Sub Code: 14IS308  Credits  :  01
Hrs/Week :0+0+2+0  Total Hours :  26

The students need to write, execute and test programs covering the entire syllabus of Data Structures.
Typical programs would be:
1. Managing records using structures, pointers – writing functions for insert, search and other operations. Same program to be implemented using dynamic memory allocation.
2. Dynamic memory allocation programs – searching, sorting, sum of n numbers.
3. Stack – Basic operations, infix to postfix, infix to prefix, evaluating a postfix expression, validity of a mathematical expression.
4. Recursive program examples including Tower of Hanoi.
5. Queues – Ordinary Queue, circular queue, priority queue, double ended queue implementation.
6. Linked list creation, insertion /deletion and display using pointers.
7. Linked implementation of stacks and queues.
8. Doubly linked list and Circular List implementation.
9. Files – managing records using files, command line arguments. Programs need to be written in C programming language.

**************************

OBJECT ORIENTED PROGRAMMING WITH C++ LAB

Sub Code: 14IS309  Credits : 01
Hrs/Week : 0+0+2+0  Total Hours : 26

Students have to write, execute and test programs covering the syllabus of 14IS306. Typical programs:

1. Program highlighting inheritance feature
2. Program using virtual functions
3. Program highlighting polymorphism feature
4. Program for string handling
5. Program for file handling
6. Programs using Functions & Operator Overloading

************
PROBABILITY THEORY AND NUMERICAL METHOD

Sub Code: IS401  
Credits : 04  
Hrs/Week : 4+0+0+0  
Total Hours : 52

Course Outcome:
At the end of the course the student will be able to
1. Understand and appreciate probabilistic models for situations involving chance effect.
2. Learn some probability distributions both discrete and continuous and its applications in real life problems.
3. Apply numerical methods to solve engineering problems where the analytical solutions for some functions are not possible.

UNIT – I
Introduction to probability: finite sample space, conditional probability and independence. Baye’s theorem (Overview). One dimensional random variable, pdf, cdf, expectation and variance. Two and higher dimensional random variables, joint pdf, marginal pdf, covariance, correlation coefficient. 12 Hours

UNIT - II
Distributions: Binomial, Poisson, uniform, normal, Gamma, exponential & Chi-square Simple problems.

Sampling Theory-Random samples, sampling distribution, t and F distributions correlation and regression. 12 Hours

UNIT – III

**Stochastic Process:** Definition, Classifications, Types of Stochastic processes, Markov process, Markov Chains, Transition probabilities, Higher transition probabilities, C-K equations. 8 Hours

**UNIT - IV**

**Numerical Analysis:** Finite differences, Newton-Gregory forward and backward difference interpolation formulae, Lagrange’s interpolation formula, Lagrange’s Inverse interpolation formula. Numerical differentiation using Newton’s forward & backward formulae.

Numerical integration: General quadrature formula, Trapezoidal rule, Simpson’s one third rule, Simpson,s three eighth rule. 12 Hours

**UNIT – V**

Roots of algebraic and transcendental equations, Numerical solution of first order ordinary differential equations: Taylor’s series Method, Modified Euler’s method, Runge – Kutta 4th order Method. Milne’s predictor-corrector methods. 8 Hours

**TEXT BOOKS:**

**REFERENCE BOOKS:**
2. Medhi: Stochastic Process

************
ANALYSIS AND DESIGN OF ALGORITHMS

Sub Code : 14IS402
Credits : 04
Hrs/Week : 3+2+0+0
Total Hours : 52

Course Outcomes:

1. To learn about the fundamentals of the algorithm and problem solving and to understand about the different problem types with relevant examples
2. To understand different data structures
3. To learn how to analyze the efficiencies of various algorithms
4. To learn different asymptotic notations such as big theta, big omega and big oh notations
5. To mathematically analyze non recursive algorithms such as finding factorial of a number and finding gcd of two numbers
6. To mathematically analyze recursive algorithms such as Fibonacci Numbers and Tower of Hanoi
7. To learn how to analyze and write algorithms such as selection sort, bubble sort, Sequential Search and Brute-Force String Matching using brute force technique
8. To learn how to analyze and solve problems like travelling salesman problem and assignment problem using exhaustive search technique.
9. To learn how to analyze and write algorithms such as merge sort, quick sort, Binary search and binary tree traversals using divide and conquer technique
10. To learn how to multiply two large integers using divide and conquer technique and using Strassen’s matrix multiplication method
11. To learn how to analyze and write algorithms such as Insertion Sort, DFS, BFS, Topological Sorting and Algorithms for Generating Combinatorial Objects using decrease and conquer technique
12. To learn how to analyze and write algorithms such as Presorting, Balanced Search Trees, Heaps and Heap sort, Problem Reduction using transform and conquer technique
13. To learn how to analyze and write algorithms such as Sorting by Counting, Input Enhancement in String Matching – Horspool’s algorithm and Bayer-Moore algorithm using space and time tradeoffs technique
14. To learn how to analyze and write algorithms such as Warshall’s algorithm, Floyd’s Algorithms and The Knapsack Problem using dynamic programming technique
15. To learn how to analyze and write algorithms such as Prim’s Algorithm, Kruskal’s Algorithm and Dijkstra’s Algorithm using greedy technique
16. To learn how to analyze and write algorithms such as n queens problem and subset-sum problem using backtracking technique
17. To learn how to analyze and write algorithms such as Assignment problem and Knapsack problem using branch and bound technique and Definition of NP Completeness

UNIT – I


5 Hours

Fundamentals of the Analysis of Algorithm Efficiency: Analysis Framework,
Asymptotic Notations and Basic Efficiency Classes, Mathematical Analysis of Non recursive and Recursive Algorithms – Fibonacci Numbers and Tower of Hanoi.

5 Hours
UNIT – II

Brute Force: Selection Sort and Bubble Sort, Sequential Search and Brute-Force String Matching, Exhaustive Search. 5 Hours

Divide and Conquer: Merge sort, Quick sort, Binary Search, Binary tree traversals and related properties, Multiplication of large integers and Strassen’s Matrix Multiplication. 5 Hours

UNIT – III

Decrease and Conquer: Insertion Sort, Depth First Search, Breadth First Search, Topological Sorting, Algorithms for Generating Combinatorial Objects. 5 Hours

Transform and Conquer: Presorting, Balanced Search Trees, Heaps and Heap sort, Problem Reduction. 5 Hours

UNIT – IV

Space and Time Tradeoffs: Sorting by Counting, Input Enhancement in String Matching – Horspool’s algorithm and Bayer-Moore algorithm. 6 Hours

Dynamic Programming: Computing a Binomial Coefficient, Warshall’s and Floyd’s Algorithms, The Knapsack Problem and memory functions. 5 Hours

UNIT – V

Greedy Technique: Prim’s Algorithm, Kruskal’s Algorithm, Dijkstra’s Algorithm, Huffman Trees. 5 Hours

Backtracking: n queens problem, subset-sum problem

Branch and Bound: Assignment problem, Knapsack problem Definition of NP Completeness. 6 Hours
TEXT BOOK:
1. Introduction to the Design and Analysis of Algorithms, Second Edition by Anany Levitin

REFERENCE BOOKS:

**********

FINITE AUTOMATA AND FORMAL LANGUAGE

Sub Code: 14 IS403 Credits : 04
Hrs/Week : 4+0+0+0 Total Hours : 52

Course Outcomes:
1. Understand the basic concepts of theory of computation
2. Review the properties of Regular Languages, Context free languages
3. Apply the concept of regular expression and Context Free Grammar to implement Lexical and Syntax Analysis phases of compilers
4. Analyze the working of Finite Automata, PDA, and Turing Machine
5. Design Finite Automata and regular expressions for different problem sets
6. Create Context free Grammar for different problem sets
7. Construct Push down Automata for different problem sets
8. Design Turing Machine for different problem sets
9. Evaluate hierarchy of Formal Languages and Automata
10. To prove non regular languages, linear languages, and non context free languages using pumping lemma

UNIT - I
Introduction to the theory of computation:
Mathematical preliminaries and notation, Three basic concepts Some applications.

3 Hours

Finite Automata:

7 Hours

UNIT - II
Regular Languages and Regular Grammars:
Regular expressions, Languages associated with regular expressions, Regular expressions denote regular languages, Regular expressions for regular languages, Regular grammars-Right and Left linear grammars. Right and Left Linear grammar examples. Right linear grammar generates regular languages, Right linear grammars for regular languages.

7 Hours

Properties of Regular Languages:
Closure properties of regular languages-Closure under simple set operations. Closure under other operations. Identifying non regular languages-Using the A pumping lemma.
UNIT - III

Context-Free languages.

3 Hours

Simplification of CFG and Normal Forms:

UNIT – IV

Properties of Context-Free Languages:
Two pumping lemmas: A pumping lemma for Context-Free languages. A pumping lemma for Linear languages.

Pushdown Automata
Nondeterministic pushdown automata-Definition, Language accepted by a PDA. PDA for Context Free languages, Context free grammars for pushdown automata, Deterministic pushdown automata.

UNIT - V

Turing Machines:
The standard Turing machine-Definition, examples, Turing machine as language accepter. Turing machine as Transducers. Combining Turing machines for complicated tasks. Universal turing machine. Other models of Turing machines: Multitape turing machines, Nondeterministic turing machines etc.

A Hierarchy of Formal Languages and Automata:
Recursive and recursively enumerable languages, The Chomsky hierarchy.

Limits of Algorithmic Computation:

Some problems that cannot be solved by Turing machines, The post correspondence problem.  

**TEXT BOOKS:**
1. An Introduction to formal languages and Automata, Peter, Linz 4th edition

**REFERENCE BOOKS:**
1. J.P.Hopcroft, Rajeev Motwani, J.D.Ullman, Introduction to Automata Theory, Languages and Computation, Pearson Education

**********

**DATA COMMUNICATION**

<table>
<thead>
<tr>
<th>Sub Code</th>
<th>Credits</th>
<th>Total Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>14IS404</td>
<td>03</td>
<td>39</td>
</tr>
<tr>
<td>Hrs/Week</td>
<td>3+0+0+0</td>
<td></td>
</tr>
</tbody>
</table>

**Course Outcomes:**
1. Define and discuss evolution of Network architecture and services
2. Understand the concept of layering. Discuss methodology that evolved with these two protocol stack architecture for computer networks.
3. Understand the concept of digital representation of information and basic properties of digital transmission systems
4. Study the characteristics and limits in digital transmission.
5. Study various modulation techniques and solve related exercises.
6. Understand the techniques used for error detection and correction and solve related exercises.
7. Study various multiplexing techniques and understand their characteristics.
8. Understand the need for synchronization and discuss techniques for flow control.
9. Study various historical and current protocols used in medium access control layer of network protocol stack.

UNIT – I

Introduction to data communications:
Communication Networks and Services: Telegraph Networks and Message Switching, Telephone Networks and Circuit Switching, the Internet, Computer Networks and Packet Switching.

Applications and Layered Architectures:

7 Hours

UNIT – II


8 Hours

UNIT – III

Digital Transmission Fundamentals – II:


UNIT – IV

Multiplexing and Peer to Peer Protocols:
Multiplexing: Frequency Division Multiplexing, Wavelength Division Multiplexing, Time Division Multiplexing. SONET: SONET Multiplexing, SONET frame structure. Framing, HDLC Data Control, Flow and Error Control: Stop-and-Wait ARQ, Go-Back-N ARQ, Selective Repeat ARQ, Piggybacking. 8 Hours

UNIT - V

Medium Access Control:

TEXT BOOKS:
REFERENCE BOOKS:

SOFTWARE ENGINEERING

Sub Code: 14IS405
Credits : 04
Hrs/Week : 4+0+0+S*
Total Hours : 52

*Self Study to be exercised under the supervision of course instructor and to be restricted to not more than 10% of the total teaching hours.

Course Outcomes:
1. To learn the notable changes in software engineering
2. To study different types of software process models
3. To learn the requirements gathering and analysis.
4. To know the various approaches to software design and their differences
5. To understand about data flow diagrams and structured design
6. To know the concepts of UML diagrams
7. To study UI design methodology
8. To learn program analysis tools and test object oriented programs.
9. To know SEI maturity model.
10. To learn COCOMO model and risk management
UNIT -1

Chapter 1: Overview
Introduction: FAQ's about software engineering, Professional and ethical responsibility.  

Chapter 2: Socio-Technical systems
Emergent system properties; Systems engineering; Organizations, people and computer systems; Legacy systems.

Chapter 3: Critical Systems
A simple safety-critical system; System dependability; Availability and reliability.

Chapter 4: Software Processes

Chapter 5: Software Requirements
Functional and Non-functional requirements; User requirements; System requirements; Interface specification; The software requirements document.

Chapter 6: Requirements Engineering Processes
Feasibility studies; Requirements elicitation and analysis; Requirements validation; Requirements management.

UNIT – II

Chapter 7: System Models
Context models; Behavioral models; Data models; Object models; Structured methods.

Chapter 8: Project Management
Management activities; Project planning; Project scheduling; Risk management.  

UNIT – III

Chapter 9: Architectural Design  Architectural design decisions; System organization; Modular decomposition styles; Control styles.  

Chapter 10: Object-Oriented design  Objects and Object Classes; An Object-Oriented design process; Design evolution.  

UNIT - IV

Chapter 11: Rapid Software Development:  Agile methods; Extreme programming; Rapid application development.  

Chapter 12: Software Evolution:  Program evolution dynamics; Software maintenance; Evolution processes; Legacy system evolution.  

Chapter 13: Verification and Validation:  Planning; Software inspections; Automated static analysis; Verification and formal methods.  

Chapter 14: Software testing:  System testing; Component testing; Test case design; Test automation.  

UNIT – V

Chapter 15: Managing People:  Selecting staff; Motivating people; Managing people; The People Capability Maturity Model.  

Chapter 16: Software Cost Estimation:  Productivity; Estimation techniques; Algorithmic cost modeling, Project duration and staffing.  
TEXT BOOK:

REFERENCE BOOKS:

**********

UNIX AND SHELL PROGRAMMING
Sub Code: 14IS406  Credits : 03
Hrs/Week : 3+0+0+0  Total Hours : 39

Course Outcomes:
1. Comprehend and analyze Unix features–Unix architecture and basic unix commands
2. Understand the Unix file system and file handling commands, vi editor and shell.
3. Remember file attributes and chmod, umask, find commands
4. Understand the process mechanism and process related commands
5. Implement the simple filters
6. Analyze the regular expression and implement filters grep & sed
7. Understand the basics of shell programming
8. Create the different implementations of shell programs.
9. Understand and create the different implementations of awk script.
10. Understand and create the different implementations of perl script.

UNIT – I

Background and some basic commands
cal: The Calendar, date: Displaying and System date, echo: Displaying a Message, printf: An Alternative to echo, bc: The Calculator, script: Recording a session, passwd: Changing your password, who: who are the users?, uname: Knowing your machine’s characteristics, tty: Knowing your terminal, stty: Displaying and setting Terminal characteristics.

The filesystem and some file handling commands
cat: Dispalying and creating Files, cp: Copying a File, rm: Deleting files, mv: Renaming files, more: Paging output, The lp Subsystem: Printing a file, file: Knowing the file types, wc: Counting lines, words and characters, od: Displaying data in octal, cmp: Comparing two Files, comm.: What is common?, diff: Converting one file to other, dos2unix and unix2dos: Converting between DOS and UNIX, Compressing files, gzip, gunzip, zip and unzip commands.

The vi Editor
Vi Basics, Input mode- Entering and replacing text, Saving text and Quoting- The ex mode, Navigation, Editing Text, Undoing last editing instructions (u and U), Repeating the last command(.), Searching for a pattern(/ and ?), Substitution- Search and Replace(:s), Customizing vi.

THE SHELL

2 Hours
The shell’s interpretive cycle, Pattern matching- The wild-cards, Escaping and Quoting, Redirection: The three standard file, /dev/null and /dev/tty: Two special files, pipes, tee: Creating a Tee, Command Substitution, Shell variables.

UNIT – II

FILE ATTRIBUTES


THE PROCESS

Process basics, ps: Process status, System Processes (-e or -a), Mechanism of process creation, Internal and External commands, Running jobs in background, nice: Job execution with lower priority, Killing process with signals, Job control, fg and bg commands at and batch: Execute later, cron: Running jobs periodically, time: Timing Processes.

UNIT - III

SIMPLE FILTERS


FILTERS USING REGULAR EXPRESSIONS – grep AND sed

Searching for a pattern, Basic regular expressions (BRE) – An Introduction, Extended regular expressions (ERE) and egrep. The stream
editor, Line addressing, Using multiple instructions (-E and -F), Context addressing, Writing selected lines to the file (w), Text editing, Substitution.

4 Hours

UNIT - IV

SHELL PROGRAMMING

Shell scripts, read: Making scripts interactive, Using command line arguments, exit and Exit status of Command, The logical operators && and || - Conditional Execution, The if conditional, Using test and [] to evaluate expressions, The case conditional, expr: Computation and string handling, $0: Calling a script by different names, while: Looping, for: looping with a list, set and shift: Manipulating the positional parameters, The here document(<<), trap: Interrupting a program, Debugging shell script with –x


8 Hours

UNIT – V

awk – AN ADVANCED FILTER

Simple awk Filtering, Splitting a Line into Fields, printf: Formatting output, Variables and Expressions, The Comparison operators, Number Processing, Variables, The –f option: storing awk programs in a file, The BEGIN and END sections, Built in variables, Arrays, Functions, Control flow,- the if statement, Looping with for, Looping with while.

4 Hours

Perl-THE MASTER MANIPULATOR

Perl preliminaries, The chop function: Removing the last character, Variables and operators, The string handling functions, Specifying filenames in command line, $_: The default variable, Current line number($.) and the Range operator(..), Lists and arrays, foreach: Looping through a list, split: Splitting into a list or array, join: Joining a list,
dec2bin.pl: converting a Decimal Number to binary, grep: Searching an array for a pattern, Associative arrays, Regular expressions and substitution, file handling, File tests, Subroutines.

4 Hours

TEXT BOOKS:

REFERENCE BOOKS:

**********

INDIVIDUAL EFFECTIVENESS LABS (IEL)

Introduction
Entry Edge (E²) is an industry readiness program designed for technology undergraduates to help them enhance important individual behavior & skills, and become productive from the very beginning of their corporate carrier. The program places a high emphasis on the pedagogy of learning by doing.

As part of the program, students first go through individual behavior & skill labs (Individual Effectiveness Labs) in their II year of engineering curriculum and then participate in “hands on” and “minds on” team activities in a simulated work environment, to accomplish tasks and to solve real-world organizational issues during a week long Immersive Group Workshop (IGW) held in the III year of their engineering course.

This document provides the syllabus and evaluation framework for Individual Effectiveness Labs (IEL).

**********
INDIVIDUAL EFFECTIVENESS LABS (IEL)

Sub Code: 14IS407
Credits : 02
Hrs / Week: 0+4+0+0
Total Hours: 52

Course Outcome:
1. To help the students understand themselves. Identify and analyze personality/behavioral attributes of personal effectiveness – exploratory orientation, self-disclosure, receptivity to feedback and sensitivity to others.
2. To help the students identify their primary and secondary motivators – what drives them for achievement?
   a. Understanding the student’s need for achievement
   b. Understanding how positive expectations lead to positive results.
3. To help the students to develop a goal driven mindset and to take the first steps into individual personal planning, controlling and measuring results.
4. To make the students aware of importance of communication and typical barriers to communication.
5. To help the students develop effective oral communication skills.
6. To help the students develop effective written communication skills.
7. To help the students develop listening skills.
8. To help the students participate in group discussions.
9. To help the students develop effective business presentation skills.
10. To help the students receive feedback with an open mind, respond to feedback and take the action on them.
11. To help the students develop time management and organization skills.

Contents

Module 1: Know Yourself
Self assessment profilers to identify and assess the following – Identify and analyze personality/behavioral attributes of personal effectiveness – exploratory orientation, self disclosure, receptivity to feedback, sensitivity to others.

8 Hours
Module 2: Achievement Motivation & Goal Setting
- Identifying primary and secondary motivators using a motivational profiler.
- Understanding need for achievement.
- Developing goal driven mindset.
- First steps into career planning. 8 Hours

Module 3: Communication Skills
- Effective oral communication
- Effective written communication
- Constructing effective messages (memo, letters, e-mails)
- Writing persuasively
- Correspondence etiquettes – letters & email
- Importance of listening responsively
- Handling conversations
- Effective group discussions 15 Hours

Module 4: Presentation Skills
- Understanding audience, presentation objectives, best practices & tools in preparation of presentation.
- Improving quality of presentation through better use of voice, eyes, gestures, visual aids.
- Presenting to groups
- Presenting one-on-one. 13 Hours

Module 5: Handling Feedback
- Seeking feedback
- Accepting feedback with an open mind
- Responding to feedback
- Actionizing feedback 6 Hours

Module 6: Time Management
- Introduction to Time Management and importance of managing self
- Beating procrastination
- Action plans-starting to achieve in a small way

- Scheduling skills 6 Hours

REFERENCE BOOKS:
2. Online reference materials provided as part of the Entry Edge program.

DESIGN AND ANALYSIS OF ALGORITHMS LAB

Sub Code: 14IS408 Credits : 01
Hrs/week : 0+0+2+0 Total hours : 26

1. Implement Euclid’s algorithm, Middle School Procedure and Consecutive Integer Checking to compute GCD (m,n) and determine the time required.
2. Sort a given set of elements using Merge sort method and determine the time required to sort the elements.
3. Sort a given set of elements using Quick sort method and determine the time required sort the elements.
4. Sort a given set of elements using the Heapsort method and determine the time required to sort the elements.
5. Print all the nodes reachable from a given starting node in a digraph using BFS method and determine the time required.
6. Implement Horspool’s algorithm for String Matching and determine the time required.
7. Implement 0/1 Knapsack problem using dynamic programming and determine the time required.
8. Find Minimum Cost Spanning Tree of a given undirected graph using Prim’s algorithm and determine the time required.
9. Find Minimum Cost Spanning Tree of a given undirected graph using Kruskal's algorithm and determine the time required.
10. Find a subset of a given set $S = \{s_1, s_2, \ldots, s_n\}$ of $n$ positive integers whose sum is equal to a given positive integer $d$. For example, if $S = \{1, 2, 5, 6, 8\}$ and $d = 9$ there are two solutions $\{1, 2, 6\}$ and $\{1, 8\}$. A suitable message is to be displayed if the given problem instance doesn't have a solution and determine the time required.
11. From a given vertex in a weighted connected graph, find shortest paths to other vertices using Dijkstra's algorithm and determine the time required.
12. Implement Recursive Binary search and determine the time required to search an element.
13. Implement Sequential search and determine the time required to search an element.
14. Sort a given set of elements using Selection sort and determine the time required to sort elements.
15. Sort a given set of elements using the Bubble sort method and determine the time required to sort the elements.
16. Implement Brute Force String Matching Technique and determine the time required.
17. Sort a given set of elements using the Insertion sort method and determine the time required to sort the elements.
18. Implement Strassen’s matrix multiplication and determine the time required.
19. Check whether a given graph is connected or not using DFS method and determine the time required.
20. Find the Topological sequence of vertices for the given graph and determine the time required.
21. Implement N Queen's problem using Back Tracking method and determine the time required.
22. Find the Binomial Co-efficient using Dynamic Programming and determine the time required.
23. Compute the transitive closure of a given directed graph using Warshall's algorithm and determine the time required.
24. Implement Floyd’s algorithm for the All-Pairs-Shortest-Paths problem and determine the time required.
25. Sort a given set of elements using the Sorting by counting method and determine the time required to sort the elements.
26. Sort a given set of elements using the Distribution counting method and determine the time required to sort the elements.
27. Generate prime numbers using Sieve’s algorithm and determine the time required to sort the elements.

**********

UNIX SHELL & PROGRAMMING LAB

Sub Code: 14IS409  Credits : 01
Hrs/week : 0+0+2+0  Total hours : 13*2

Course Outcomes:

C-Programs

1. Write a C program to do the following using fork().
Create a child process. Child process prints its own proc id & id of its parents and exits. The parent process waits for its child to finish & prints its own process id & id of its child and exits.
2. Write a C program to prompt the user to the name of environment variable and prints its value if it is defined & suitable message otherwise.

Shell scripts

3. Write a shell script that accepts 2 file names as arguments. Check the permissions of these files are identical. If they are then output is common permission otherwise output permission of each file.
4. Write a program to display last modification time of file.
5. Write a shell script that accepts the pathname & creates all the components in that pathname as directory.
6. Write a non-recursive shell script that accepts any no. Of arguments & prints them in reverse order.
7. Write a shell script that displays all the links of the file.
8. Write a shell function that takes a valid directory name as an argument and calculate the maximum length of file in that directory.
9. Write a shell script that uses awk function to find total size of an ordinary file.
10. Write a shell script that accepts valid login name as an argument & print their corresponding home directory.
11. Write a shell code to accept a string from terminal & echo a suitable message if it doesn’t have at least 10 characters. Use case & expr.
12. Devise a shell script that accepts 2 directories and delete the files which are similar from 2nd directory.
13. Write a shell code to accept file name as an argument and checks whether it is present in current directory.If it doesn’t then convert its name to uppercase only if file with new name doesn’t exist.
14. Devise a shell script that looks at every components of path and checks whether the directory exist & is also accessible.

**AWK Scripts**

15. Write an awk script to fill asterisk to the end of the line till the length of the line is 127 characters.
16. Write an awk script to print the line and addition of no’s present in that line where a file contains fields in the form of digits separated with space.
17. Write an awk script that folds the long line into 40 columns. If line exceeds 40 then it must be broken after 40 to next line.
18. Write an awk script to delete duplicate lines from a text file.

**PERL Scripts**
19. Write a perl script that echoes its command line argument 1 per line after translating the lower case letters to upper case.
20. Write a perl program to convert unsigned binary into decimal.
21. Write a perl program to convert decimal to binary.
22. Write a perl program which asks user to repeatedly enter a no. and if he enters 0 it should print total of all the no’s entered so far.
23. Write a perl program that takes filename as an argument, checks whether file exist & binary and prints binary if it is binary.
24. Write a perl program that prompts user to input a string & no. and prints the string that many times.

**********