Credit Structure for B.Tech. (Information Technology)

(For newly admitted students from session 2020-21)

Credit Courses											
	Category Semesters	I	II	III	IV	V	VI	VII	VIII	Total	Min. Req.
Undergraduate Core Courses (158 min. credits)	Basic Sciences & Maths (BSM)	9	14	9	4	-	-	-	-	**36**	**36**
	Engineering Fundamentals (EF)	11	7	6	2	-	-	-	-	**26**	**24**
	Department Core (DC)	-	-	10	15	19	24	10	4	**82**	**78**
	Management (M)	-	-	-	3	3	-	-	-	**6**	**6**
	Humanities & Social Science Core (HSSC)	4	-	-	-	-	-	-	-	**4**	**4**
	Project (P)	-	-	-	-	-	-	5	5	**10**	**10**
Undergraduate Programme Electives (22 min. credits)	Programme Electives (PE)	-	-	-	-	-	-	8	8	**16**	**16**
	Open Electives (OE)	-	-	-	-	-	-	-	4	**4**	**3**
	Humanities & Social Science Electives (HSSE)	-	3	-	-	-	-	-	-	**3**	**3**
Min. Credits Required (158+22=180)	**Total**	**24**	**24**	**25**	**24**	**22**	**24**	**23**	**21**	**187**	**180**

Audit Courses		
	Total	Min. Req.
(Min. 3 Credits audit subjects from other departments will be offered during Semester I-V)	21	15
Seminar	3	3
Industrial/Practical Training (IPT)	1	1

Course Structure of BTech (Information Technology)

Credit Value

1 Period Lecture=1 Credit **1 Period Tutorial=1 Credit** **2 Period Practical=1 Credit**

Freshman Year, Semester-I

S.N.	Category	Paper Code	Subject	L	T	P	Credit
1.	BSM	BAS-01	Engineering Mathematics-I	3	1	0	4
2.	BSM	BAS-02	Engineering Physics-I	3	1	2	5
3.	EF	BIT-01	Fundamentals of Information Technology	3	1	-	4
4.	EF	BEE-01	Principles of Electrical Engineering	3	1	2	5
5.	HSSC	BAS-03	Professional Communication	3	1	0	4
6.	EF	BIT-02	Software Tools-I	0	0	4	2
7.	AC		Audit Course				-
			Total	**15**	**5**	**8**	**24**

Freshman Year, Semester-II

S.N.	Category	Paper Code	Subject	L	T	P	Credit
1.	BSM	BAS-07	Engineering Mathematics-II	3	1	0	4
2.	BSM	BAS-08	Engineering Physics-II	3	1	2	5
3.	BSM	BAS-24	Applied Computational Methods	3	1	2	5
4.	EF	BIT-03	Programming Fundamentals	3	1	2	5
5.	HSSE	BAS-**	Humanities & Social Science Electives	2	1	0	3
6.	EF	BCE-10	Engineering Graphics	0	0	4	2
7.	AC		Audit Course				-
			Total	**14**	**5**	**10**	**24**

Sophomore Year, Semester-III

S.N.	Category	Paper Code	Subject	L	T	P	Credit
1.	BSM	BAS-01	Discrete Mathematics	3	1	0	4
2.	BSM	BAS-14	Graph Theory	3	1	2	5
3.	EF	BIT-11	Switching Theory & Logic Design	3	1	-	4
4.	DC	BIT-12	Data Structures	3	1	2	5
5.	DC	BIT-13	Object Oriented Programming	3	1	2	5
6.	EF	BIT-14	Software Tools-II	0	0	4	2
7.	AC		Audit Course				-
			Total	**15**	**5**	**10**	**25**

Sophomore Year, Semester-IV

S.N.	Category	Paper Code	Subject	L	T	P	Credit
1.	BSM	BAS-26	Optimization Techniques	3	1	0	4
2.	M	MBA-113	Management Information System	2	1	-	3
3.	DC	BIT-15	Design & Analysis of Algorithm	3	1	2	5
4.	DC	BIT-16	Computer Organization & Architecture	3	1	2	5
5.	DC	BIT-17	Database Management System	3	1	2	5
6.	EF	BIT-18	Software Tools-III	0	0	4	2
7.	AC		Audit Course				-
			Total	**14**	**5**	**10**	**24**

Junior Year, Semester-V

S.N.	Category	Paper Code	Subject	L	T	P	Credit
1.	M	MBA-02	Engineering & Managerial Economics	2	1	0	3
2.	DC	BIT-26	Operating System	3	1	2	5
3.	DC	BIT-27	Computer Networks	3	1	2	5
4.	DC	BIT-28	Software Engineering	3	1	2	5
5.	DC	BIT-29	Automata Theory	3	1	-	4
6.	AC		Audit Course				-
			Total	**14**	**5**	**06**	**22**

Junior Year, Semester-VI

S.N.	Category	Paper Code	Subject	L	T	P	Credit
1.	DC	BIT-31	Data Mining & Ware Housing	3	1	0	4
2.	DC	BIT-32	Artificial Intelligence	3	1	2	5
3.	DC	BIT-33	Machine Learning	3	1	2	5
4.	DC	BIT-34	Wireless Sensor Network & IoT	3	1	2	5
5.	DC	BIT-35	Network Security & Cryptography	3	1	2	5
6.	AC	BIT-30	Seminar	-	-	6	-
			Total	**15**	**5**	**08**	**24**

Senior Year, Semester-VII

S.N.	Category	Paper Code	Subject	L	T	P	Credit
1.	DC	BIT-41	Graphics & Visual Computing	3	1	2	5
2.	DC	BIT-42	Mobile Computing	3	1	2	5
3.	PE-1	BIT-*	Programme Elective-1	3	1	0	4
4.	PE-2	BIT-*	Programme Elective-2	3	1	0	4
5.	P	BIT-40	Project Part-1	0	0	10	5
6.	AC	BIT-45	Industrial/Practical Training	0	0	2	-
			Total	**12**	**4**	**14**	**23**

Senior Year, Semester-VIII

S.N.	Category	Paper Code	Subject	L	T	P	Credit
1.	DC	BIT-43	Distributed System	3	1	0	4
2.	PE-3	BIT-*	Programme Elective-3	3	1	0	4
3.	PE-4	BIT-*	Programme Elective-4	3	1	0	4
4.	OE	BOE-*	Open Elective Offered by other dept.	3	1	0	4
5.	P	BIT-50	Project Part-2	0	0	10	5
			Total	**12**	**4**	**10**	**21**

Engineering Fundamentals & Departmental Core (Information Technology)

Sr. No.	Paper Code	Subject	Prerequisite	L	T	P	Credit
1.	BIT-01	Fundamentals of Information Technology	-	3	1	0	4
2.	BIT-02	Software Tools-I	-	0	0	4	2
3.	BIT-03	Programming Fundamentals	-	3	1	2	5
4.	BIT-11	Switching Theory & Logic Design	-	3	1	0	4
5.	BIT-12	Data Structures	-	3	1	2	5
6.	BIT-13	Object Oriented Programming	-	3	1	2	5
7.	BIT-14	Software Tools-II	-	0	0	4	2
8.	BIT-15	Design & Analysis of Algorithm	-	3	1	2	5
9.	BIT-16	Computer Organization & Architecture	-	3	1	2	5
10.	BIT-17	Database Management System	-	3	1	2	5
11.	BIT-18	Software Tools-III	-	0	0	4	2
12.	BIT-26	Operating System	-	3	1	2	5
13.	BIT-27	Computer Networks	-	3	1	2	5
14.	BIT-28	Software Engineering	-	3	1	2	5
15.	BIT-29	Automata Theory	-	3	1	0	4
16.	BIT-31	Data Mining & Ware Housing	-	3	1	0	4
17.	BIT-32	Artificial Intelligence	-	3	1	2	5
18.	BIT-33	Machine Learning	-	3	1	2	5
19.	BIT-34	Wireless Sensor Network & IoT	-	3	1	2	5
20.	BIT-35	Network Security & Cryptography	-	3	1	2	5
21.	BIT-30	Seminar	-	0	0	6	0
22.	BIT-41	Graphics & Visual Computing	-	3	1	2	5
23.	BIT-42	Mobile Computing	-	3	1	2	5
24.	BIT-40	Project Part-1	-	0	0	10	5
25.	BIT-45	Industrial/Practical Training	-	0	0	2	1
26.	BIT-43	Distributed System	-	3	1	0	4
27.	BIT-50	Project Part-2	Project-1	0	0	10	5

Programme Electives (Information Technology)

Sr. No.	Paper Code	Subject	Prerequisite	L	T	P	Credit
		PE-1 & PE-2					
1.	BIT-51	.Net Technology	-	3	1	0	4
2.	BIT-52	Advanced JAVA	-	3	1	0	4
3.	BIT-53	Real Time System	-	3	1	0	4
4.	BIT-54	Artificial Intelligence Search Methods for problem Solving	-	3	1	0	4
5.	BIT-55	Aspect Oriented Programming	-	3	1	0	4
6.	BIT-56	Big Data Computing	-	3	1	0	4

7.	BIT-57	Blockchain Architecture Design and Use Cases	-	3	1	0	4
8.	BIT-58	Cloud Computing and Distributed Systems	-	3	1	0	4
9.	BIT-59	Compiler Design	-	3	1	0	4
10.	BIT-60	Computer Vision: Foundations and Applications	-	3	1	0	4
11.	BIT-61	Functional Programming	-	3	1	0	4
12.	BIT-62	Data Science for Engineers	-	3	1	0	4
13.	BIT-63	Database Administration with ORACLE	-	3	1	0	4
14.	BIT-64	Deep Learning	-	3	1	0	4
		PE-3 & PE-4					
15.	BIT-65	Android Programming	-	3	1	0	4
16.	BIT-66	Embedded System	-	3	1	0	4
17.	BIT-67	Hardware Modelling using Verilog	-	3	1	0	4
18.	BIT-68	Hardware Security	-	3	1	0	4
19.	BIT-69	High Performance Computing	-	3	1	0	4
20.	BIT-70	Introduction to Parallel Programming in Open MP	-	3	1	0	4
21.	BIT-71	Linux Administration & Networking	-	3	1	0	4
22.	BIT-72	Digital Signal Processing	-	3	1	0	4
23.	BIT-73	Multi-Core Computer Architecture – Storage and Interconnects	-	3	1	0	4
24.	BIT-74	Network Programming	-	3	1	0	4
25.	BIT-75	Parallel Algorithms	-	3	1	0	4
26.	BIT-76	Scalable Data Science	-	3	1	0	4
27.	BIT-77	Software Design, Construction & Quality Management	-	3	1	0	4
28.	BIT-78	Software Verification & Validation	-	3	1	0	4

Open Electives for other department

Sr. No.	Paper Code	Subject	Prerequisite	L	T	P	Credit
1.	BOE-25	Linux & Shell Programming	-	3	1	0	4
2.	BOE-26	Web Technology	-	3	1	0	4
3.	BOE-27	Digital Forensic & Cyber Laws	-	3	1	0	4
4.	BOE-28	Network Security	-	3	1	0	4

Audit Courses for BTech (IT)

S.N.	**Category**	**Paper Code**	**Subject**	**L**	**T**	**P**	**Credit**
1.	AC	BAS-05	Environment & Ecology	2	1	0	-
2.	AC	BEC-01	Fundamentals of Electronics Engineering	2	1	0	-
3.	AC	BCS-13	Internet & Java Programming	3	1	2	-
4.	AC	BCS-53	LAMP Technology	3	1	0	-
5.	AC	BCS-73	Neural Network & Fuzzy Systems	3	1	0	-
6.	AC	BEE-15	Introduction to Microprocessors	3	1	2	-
7.	AC	MAS-109	Foreign Language- French	2	1	0	-
8.	AC	MAS-109	Foreign Language- German	2	1	0	-
9.	AC	MAS-109	Foreign Language- Spanish	2	1	0	-

Humanities & Social Science Electives (HSSE)

S.N.	Category	Paper Code	Subject	L	T	P	Credit
1.	AC	BAS-10	Technical Writing	2	1	0	3
2.	AC	BAS-11	Human Values & Professional Ethics	2	1	0	3
3.	AC	BAS-12	Industrial Psychology	2	1	0	3
4.	AC	BCS-13	Industrial Sociology	2	1	0	3

Computer Fundamental (CF)courses for BBA

S.N.	Category	Paper Code	Subject	L	T	P	Credit
1.	CF	BIT-81	Fundamentals of Computer Applications	2	0	0	2
2.	CF	BIT-82	IT Tools for Business	2	0	2	3

Syllabus (B.Tech.-I)

BIT-01 **Fundamentals of Information Technology**

Course category : Engineering Fundamentals (EF)
Pre-requisite Subject : NIL
Contact hours/week : Lecture: 3, Tutorial: 1 , Practical: 0
Number of Credits : 4
Course Assessment methods : Continuous assessment through tutorials, attendance, home assignments, quizzes and Three Minor tests and One Major Theory Examination

Course Outcomes : The students are expected to be able to demonstrate the following knowledge, skills, and attitudes after completing this course

1. understand the basics of computers Hardware/Software
2. understand the importance of data compression and the algorithms for lossy and lossless data compression
3. understand the concept of operating system and fundamentals of computer networking

UNIT-I **9**

Introduction to Computer Hardware/Software: Processor, Motherboard, I/O Devices, peripherals, Memory Types & Hierarchy: Cache, Primary & Secondary memories with examples, Concept of Computer Languages: Low-Level, Assembly and High-Level, System Software: Assembler, Compiler, Interpreter, Loader/Linker

UNIT-II **9**

Data & Information, Digital representation of Information, Number Systems & Comparisons: Binary, Octal, Decimal, Hexadecimal, Text Representation: ASCII, EBCDIC, Unicode, Multimedia Data, Data Compression Types and Techniques: Lossy / Lossless, Huffman, Shannon-Fano, Dictionary Based Compression techniques

UNIT-III **9**

Operating System: Concept, Functions, Types, Single-user/Multi-user operating system, Architectural differences, Shell fundamentals, Exemplary commands: Internal & External, Basics of Primary and Secondary Memory Management

UNIT-IV **9**

Network Basics: Concept, Types, Transmission modes, Topologies, OSI & TCP/IP Models: Functions of different Layers, concept of MAC, IP (Private/Public) and TCP addresses, Basic Introduction to CSMA/CD, IP & TCP/UDP and HTTP Protocols, Current Internet Applications

Text Books & References

1. Mark Nelson and Jean-Loup Gailly"The Data Compression Book", M&T Books, A Division of MIS: Press,Inc.
2. K Sayood, "Introduction to Data Compression" 3/e, Elsevier 2006
3. Forouzan, Data Communication and Networking, TMH
4. Silberschatz, A., Galvin, P. and Gagne, G., Applied Operating Systems Concepts, John Wiley& Sons Inc.

BIT-02 **Software Tools-I**

Course category : Engineering Fundamentals (EF)

Pre-requisite Subject : NIL

Contact hours/week : Lecture: 0, Tutorial: 0 , Practical: 4

Number of Credits :2

Course Assessment methods : Continuous assessment through Viva-voce, Practical work/Record, attendance and Major Practical Examination

Course Outcomes : The students are expected to be able to demonstrate the following knowledge, skills and attitudes after completing this course

1. Understanding of Booting Process and installation of Operating system
2. Usage of Operating system commands
3. Understanding of Shell and its usage as a programming language
4. Understanding of Computer Networking concepts

Experiments

1. Understanding CMOS settings of operating system
2. Installation of Linux operating system using virtualization technique
3. Understanding and practice of various Linux commands
4. Creation/usage of various types of files supported by Linux
5. Practice of Computer networking commands
6. Programs using shell programming

BIT-03 **Programming Fundamentals**

Course category : Engineering Fundamentals (EF)

Pre-requisite Subject : NIL

Contact hours/week : Lecture: 3, Tutorial: 1, Practical: 2

Number of Credits : 5

Course Assessment methods : Continuous assessment through tutorials, attendance, home assignments, quizzes and Three Minor tests and One Major Theory Examination

Course Outcomes : The students are expected to be able to demonstrate the following knowledge, skills and attitudes after completing this course

1. Describing the basics of terminologies used in computer programming.
2. Practicing C language programming by writing, compiling and debugging the code.
3. Designing programs involving simple statements, conditional statements, iterative statements, array, strings, functions, recursion and structure.
4. Discussing the dynamic memory allocations and use of the pointers.
5. Applying basic operations on files through programs.
6. Studying and implementing the codes using macros, preprocessor directives and command line arguments

UNIT-I **9**

Basics of Computers and Programming: Functional diagram of computer; Language Processors; Approaches to problem solving, Concept of algorithm and flow charts. **Simple Statements:** Datatypes; Tokens and its types; Variable declaration and initialization; User defined type declaration: typedef, enum; Comments; Format specifiers; Standard I/O: taking input and displaying output; **Operators:** types, precedence and associativity; Expressions; Type conversion, C short-hands.

UNIT-II **9**

Conditional Statements: Simple if, if-else, nested if-else, else-if ladder, switch statements, nested switch, advantages of switch over nested if, restrictions on switch values. **Iterative Statements:** Concepts of entry and exit controlled loops; Uses of for, while and do while loops; Nested Loops; Printing various patterns using nested loops; Using break, continue and goto statements.

UNIT-III **9**

Arrays: Single-dimensional, multi-dimensional array and their applications; declaration and manipulation of arrays; strings and string handling functions. **Pointers:** Pointer and address arithmetic; dereferencing; pointers and arrays;

dynamic memory allocation and de-allocation. **Functions:** Function prototype; Arguments and its types: actual, formal and default arguments; Scope of a variable; Argument passing methods; Passing pointer as the function argument; Recursion: types, advantages and disadvantages; Storage class specifies; Character test functions.

UNIT-IV **9**

Structure: Declaring and defining structures; Array within structure; Array of structure; Defining and using some data structures: Stack, Queue, and Linked lists. **File Handling:** Types of files; Text files and different operations on text files, opening a file, closing a file;Data structure of a file; EOF; I/O operations on files; Random access to the files. **Standard C Preprocessors & C Library:** Pre-processor, Directives, Macro, Macro substitution; Conditional Compilation; Command Line Arguments; Standard C Library.

Text Books & References

1. Brian W. Kernighan and Dennis M. Ritchie, "The C programming language", Pearson
2. E. Balagurusamy, "Programming in ANSI C", McGraw Hill Education
3. Yashavant Kanetkar, "Let Us C", bpb publication
4. Jeri R. Hanly, Elliot B. Koffman, "Problem Solving and Program Design in C", Pearson
5. Herbert Schildt,"C: The Complete Reference",McGraw Hill Education

EXPERIMENTS

Implementing programs in following categories using programming language 'C':

1. Programs of simple statements, conditional statements and iterative statements with their applications.
2. Programs of single and multi dimensional arrays and their applications.
3. Programs of strings and their applications
4. Programs of pointer and their applications
5. Programs of function and their applications
6. Programs of structure and their applications
7. Codes of file handling and management
8. Codes with Preprocessor, Macro, Conditional Compilation and Command Line Arguments

BIT-11 **SWITCHING THEORY & LOGIC DESIGN** **4 Credits (3-1-0)**

Course Objectives:

The course objectives of the course are:

1. To introduce the concept of digital and binary systems
2. To be able for designing and analysing combinational logic circuits
3. To be able for designing and analysing sequential logic circuits
4. To understand basic software tools for the design and implementation of digital circuits and systems
5. To reinforce theory and techniques taught in the classroom through experiments and projects in the laboratory.

Learning Outcomes:

On completion of this course, students will be able to:

1. Design a finite state machine and sequential logic design.
2. Synthesize a logic design from a natural language description of a problem.
3. Realize a complete arithmetic and logic unit.
4. Generate a realization of combinational logic in a programmable gate array.
5. Simulate a complete design to evaluate functional correctness and timing.

UNIT- I: **9**

Binary Codes - Weighted and Non-Weighted - Binary Arithmetic Conversion Algorithms - Error Detecting and Error Correcting Codes - Canonical and Standard Boolean Expressions - Truth Tables.

UNIT- II: **9**

K-Map Reduction - Don't Care Conditions - Adders / Subtractors- Carry Look-Ahead Adder - Code Conversion Algorithms - Design of Code Converters - Equivalence Functions.

Binary/Decimal Parallel Adder/Subtractor for Signed Numbers - Magnitude Comparator - Decoders / Encoders - Multiplexers / Demultiplexers- Boolean Function Implementation using Multiplexers.

UNIT- III: **9**

Sequential Logic - Basic Latch - Flip-Flops (SR, D, JK, T and Master-Slave) - Triggering of Flip-Flops - Counters - Design Procedure - Ripple Counters - BCD and Binary - Synchronous Counters.

UNIT- IV: **9**

Registers - Shift Registers - Registers with Parallel Load - Memory Unit - Examples of RAM, ROM, PROM, EPROM - Reduction of State and Flow Tables - Race-Free State Assignment - Hazards.

Text Books & References:

1. Morris Mano, Digital Design, Prentice Hall of India
2. W. H. Gothmann, Digital Electronics -An Introduction to Theory and Practice, Prentice Hall of India

---

BIT-12 **DATA STRUCTURES** **5 Credits (3-1-2)**

Course Objectives:

The course objectives of the course are:

6. To understand the various techniques of searching and sorting
7. To design and implement arrays, stacks, queues, and linked lists
8. To design and implement the non-linear data structures such as trees and graphs

Learning Outcomes:

On completion of this course, students will be able to:

1. Write the algorithms and understand their complexities
2. Learn various linear data structures such as stack and queue
3. Learn various non-linear data structures such as such as tree and graph
4. Know applications of linear and non-linear data structures
5. Implement the different data structures statically (using array)
6. Implement the different data structures dynamically (using pointer or linked list)
7. Understand various searching and sorting techniques
8. Understand different hashing techniques

UNIT- I : Introduction **9**

Basics: Data and Information, Need of data structure, Algorithms and their complexities, Time complexity, Space Complexity, Time Space Trade-off, Big Oh notation. **Array:** Definition, Different representations – row major, column major, address calculation, Basic operations on matrix, Sparse matrix and its types, Basic array operations (Creation, Insertion into and Deletion from an array), Applications of array, Array representation of polynomials. **Linked List:** Definition and its types, Singly linked list, Circular linked list, Doubly linked list, Basic linked list operations (Creation, Insertion into and Deletion from a linked list), Applications of linked list, Linked list representation of polynomial.

UNIT II : Linear Data Structures **9**

Stack: Definition and its implementations (static using array and dynamic using linked list), PUSH and POP operations, Applications of stack, Infix, Prefix and Postfix Expressions and their inter-conversions, Evaluation

of Postfix expressions using stack, Recursion, Types of recursion, difference between recursion and iteration, Tower of Hanoi Problem. **Queue**: Definition and its types, Circular queue, Double ended queue, Implementation of queue (static using array and dynamic using linked list), Basic operation (Creation, Insertion into and Deletion from a queue), Applications of queue.

UNIT III : Nonlinear Data Structures **9**

Trees: Definition, Basic terminologies, difference between tree and forest, tree representation, Types of tree, Implementation of tree (static using array and dynamic using linked list), Basic operations (Creation, Insertion into and Deletion from a tree), Traversal techniques (Inorder, Preorder and Postorder), Applications of tree, Binary trees, B Tree, B+ Tree, Binary Search Tree, Height balanced binary tree – AVL Tree, Threaded binary tree. **Graph**: Definitions, Graph representations – adjacency matrix, adjacency list, Basic operations (Creation, Insertion into and Deletion from a graph), Traversal techniques (Depth first search - DFS, Breadth first search - BFS), Applications of graph. Minimum spanning tree – Prim's algorithm, Kruskal algorithm.

UNIT IV : Searching, Sorting and Hashing **9**

Searching Algorithms: Sequential or Linear search, Binary search. **Sorting Algorithms:** Bubble sort, Insertion sort, Merge sort, Quick sort, Heap sort (concept of max heap, and Min Heap), Radix sort. **Hashing Techniques:** Definition, Difference between Searching and Hashing, Hash functions, Collision.

List of Experiments:

Write C/C++ programs to illustrate the concept of the following:

1. Arrays
2. Linked List
3. Stack
4. Queue
5. Tree
6. Graph
7. Searching
8. Sorting
9. Hashing
10. Applications of various data structures

Text Books & References:

1. "Fundamentals of Data Structures of C", Ellis Horowitz, Sartaj Sahni, Susan Anderson-freed.
2. "Data Structures", S. Lipschutz.
3. "Introduction to Algorithms", T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein.
4. "Data Structures and Program Design in C", Robert L. Kruse, Bruce P. Leung.
5. "Data Structures in C", Aaron M. Tenenbaum.

BIT-13 OBJECT ORIENTED PROGRAMMING 5 Credits (3-1-2)

Course Objectives:

The course objectives of the course are:

1. To give introductory as well as advanced knowledge of object oriented programming
2. To provide various syntax and object oriented programming concepts
3. To improve problem solving skills with Python

Learning Outcomes:

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

1. Write basic and advance object oriented programs
2. Create and use classes and objects
3. Write code for Constructors and Destructors
4. Write code for Inheritance
5. Write code for Polymorphism
6. Write code for Operator Overloading
7. Write code for Exception handling
8. Write code for file handling and various file operations

UNIT I **9**

Tokens, Keywords, Identifiers and Constants, Basic Data Types, User-Defined and Derived Data Types, Type Casting, Implicit Conversion, Operators and Expressions, Operator Precedence, Simple statements, Conditional statements, Iterative statements, Array, Function, Pointer, Structure

UNIT II **9**

Basic Concepts of Object Oriented Programming, Object Oriented Programming Paradigm, Benefits of OOP, Object Oriented Languages, Class and Objects, Scope Resolution Operator, Access specifiers, Data members, Accessing class members, Data hiding, Member function, Inline function, Friend function, Passing objects as arguments, Returning objects from functions

UNIT III **9**

Constructors and its types, Destructor, Constructor overloading, Order of construction and destruction, Inheritance, Single, Multilevel, Multiple, Hierarchical, Hybrid Inheritance, Base class, Derived class, Virtual function, Polymorphism, Operator Overloading, Overloading Unary Operators, Overloading Binary Operators

UNIT IV **9**

Exception Handling, Throwing and Catching Mechanism, Templates, File handling, Types of files, End of File, Basic file operations: creating, opening, closing, reading, writing and appending a file, copying a file to another, Object oriented system development

List of Experiments:

Write programs to illustrate the following concepts:

1. Operators and expressions
2. Simple statements, Conditional statements and Iterative statements
3. Arrays
4. Functions
5. Pointers
6. Structures
7. Objects and Classes
8. Inline Function, Friend function and Virtual Functions
9. Scope Resolution Operator
10. Constructors and Destructors
11. Inheritance
12. Polymorphism
13. Operator Overloading
14. Exception Handling
15. File operations

Text Books & References:

1. P. Deitel and H. Deitel, “C++ How to Program”, Pearson.
2. E. Balagurusamy, “Object Oriented Programming with C++”, TMH Publication.

3. Yashavant Kanetkar, “Let us C++”, BPB Publications
4. Robert Lafore, “Object Oriented Programming in Turbo C++”, Galgotia Publication.
5. B. Trivedi, "Programming with ANSI C++", Oxford University Press.
6. Ira Pohl, "Object Oriented Programming using C++", Pearson Education, Second Edition Reprint
7. B. Stroustrup, "The C++ Programming language", Pearson Education.
8. Timthy Budd, “An Introduction to Object Oriented Programming with C++,” Addition-Wesley.
9. Kip R. Irvine, “C++ and Object-Oriented Programming,” Prentice Hall.

BIT-14 **Software Tools-II** **2 Credits (0-0-4)**

Course Objectives:

The objectives of this course are:

1. To give overview of advanced C programming (command line arguments & file handling)
2. To give knowledge about practical usage of LINUX system calls
3. To develop capability of system call programming for practical understanding of operating system’s internal functioning

Learning Outcomes:

The students are expected to be able to demonstrate the following knowledge, skills and attitudes after completing this course

1. Understanding of process handling in Operating system
2. Usage of kernel system calls
3. Understanding of multiprogramming function of operating system using system calls

Experiments

1. Programs related to advanced C programming (command line arguments & file handling)
2. Understanding and usage of various LINUX kernel system calls
3. System call programming (programs based on LINUX kernel system calls using C language)

BIT-11 SWITCHING THEORY & LOGIC DESIGN 4 Credits (3-1-0)

Course Objectives:

The course objectives of the course are:

1. To introduce the concept of digital and binary systems
2. To be able for designing and analysing combinational logic circuits
3. To be able for designing and analysing sequential logic circuits
4. To understand basic software tools for the design and implementation of digital circuits and systems
5. To reinforce theory and techniques taught in the classroom through experiments and projects in the laboratory.

Learning Outcomes:

On completion of this course, students will be able to:

1. Design a finite state machine and sequential logic design.
2. Synthesize a logic design from a natural language description of a problem.
3. Realize a complete arithmetic and logic unit.
4. Generate a realization of combinational logic in a programmable gate array.
5. Simulate a complete design to evaluate functional correctness and timing.

UNIT- I: 9

Binary Codes - Weighted and Non-Weighted - Binary Arithmetic Conversion Algorithms - Error Detecting and Error Correcting Codes - Canonical and Standard Boolean Expressions - Truth Tables.

UNIT- II: 9

K-Map Reduction - Don't Care Conditions - Adders / Subtractors- Carry Look-Ahead Adder - Code Conversion Algorithms - Design of Code Converters - Equivalence Functions.
Binary/Decimal Parallel Adder/Subtractor for Signed Numbers - Magnitude Comparator - Decoders / Encoders - Multiplexers / Demultiplexers- Boolean Function Implementation using Multiplexers.

UNIT- III: 9

Sequential Logic - Basic Latch - Flip-Flops (SR, D, JK, T and Master-Slave) - Triggering of Flip-Flops - Counters - Design Procedure - Ripple Counters - BCD and Binary - Synchronous Counters.

UNIT- IV: 9

Registers - Shift Registers - Registers with Parallel Load - Memory Unit - Examples of RAM, ROM, PROM, EPROM - Reduction of State and Flow Tables - Race-Free State Assignment - Hazards.

Text Books & References:

1. Morris Mano, Digital Design, Prentice Hall of India
2. W. H. Gothmann, Digital Electronics -An Introduction to Theory and Practice, Prentice Hall of India

BIT-12 DATA STRUCTURES 5 Credits (3-1-2)

Course Objectives:

The course objectives of the course are:

6. To understand the various techniques of searching and sorting
7. To design and implement arrays, stacks, queues, and linked lists
8. To design and implement the non-linear data structures such as trees and graphs

Learning Outcomes:

On completion of this course, students will be able to:

1. Write the algorithms and understand their complexities
2. Learn various linear data structures such as stack and queue
3. Learn various non-linear data structures such as such as tree and graph

4. Know applications of linear and non-linear data structures
5. Implement the different data structures statically (using array)
6. Implement the different data structures dynamically (using pointer or linked list)
7. Understand various searching and sorting techniques
8. Understand different hashing techniques

UNIT- I : Introduction **9**

Basics: Data and Information, Need of data structure, Algorithms and their complexities, Time complexity, Space Complexity, Time Space Trade-off, Big Oh notation. **Array:** Definition, Different representations – row major, column major, address calculation, Basic operations on matrix, Sparse matrix and its types, Basic array operations (Creation, Insertion into and Deletion from an array), Applications of array, Array representation of polynomials. **Linked List:** Definition and its types, Singly linked list, Circular linked list, Doubly linked list, Basic linked list operations (Creation, Insertion into and Deletion from a linked list), Applications of linked list, Linked list representation of polynomial.

UNIT II : Linear Data Structures **9**

Stack: Definition and its implementations (static using array and dynamic using linked list), PUSH and POP operations, Applications of stack, Infix, Prefix and Postfix Expressions and their inter-conversions, Evaluation of Postfix expressions using stack, Recursion, Types of recursion, difference between recursion and iteration, Tower of Hanoi Problem. **Queue**: Definition and its types, Circular queue, Double ended queue, Implementation of queue (static using array and dynamic using linked list), Basic operation (Creation, Insertion into and Deletion from a queue), Applications of queue.

UNIT III : Nonlinear Data Structures **9**

Trees: Definition, Basic terminologies, difference between tree and forest, tree representation, Types of tree, Implementation of tree (static using array and dynamic using linked list), Basic operations (Creation, Insertion into and Deletion from a tree), Traversal techniques (Inorder, Preorder and Postorder), Applications of tree, Binary trees, B Tree, B+ Tree, Binary Search Tree, Height balanced binary tree – AVL Tree, Threaded binary tree. **Graph**: Definitions, Graph representations – adjacency matrix, adjacency list, Basic operations (Creation, Insertion into and Deletion from a graph), Traversal techniques (Depth first search - DFS, Breadth first search - BFS), Applications of graph. Minimum spanning tree – Prim‘s algorithm, Kruskal algorithm.

UNIT IV : Searching, Sorting and Hashing **9**

Searching Algorithms: Sequential or Linear search, Binary search. **Sorting Algorithms:** Bubble sort, Insertion sort, Merge sort, Quick sort, Heap sort (concept of max heap, and Min Heap), Radix sort. **Hashing Techniques:** Definition, Difference between Searching and Hashing, Hash functions, Collision.

List of Experiments:

Write C/C++ programs to illustrate the concept of the following:

1. Arrays
2. Linked List
3. Stack
4. Queue
5. Tree
6. Graph
7. Searching
8. Sorting
9. Hashing
10. Applications of various data structures

Text Books & References:

1. “Fundamentals of Data Structures of C”, Ellis Horowitz, Sartaj Sahni, Susan Anderson-freed.
2. “Data Structures”, S. Lipschutz.

3. "Introduction to Algorithms", T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein.
4. "Data Structures and Program Design in C", Robert L. Kruse, Bruce P. Leung.
5. "Data Structures in C", Aaron M. Tenenbaum.

BIT-13 OBJECT ORIENTED PROGRAMMING 5 Credits (3-1-2)

Course Objectives:

The course objectives of the course are:

1. To give introductory as well as advanced knowledge of object oriented programming
2. To provide various syntax and object oriented programming concepts
3. To improve problem solving skills with C++

Learning Outcomes:

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

1. Write basic and advance object oriented programs
2. Create and use classes and objects
3. Write code for Constructors and Destructors
4. Write code for Inheritance
5. Write code for Function Overloading
6. Write code for Operator Overloading
7. Write code for Dynamic or Run-time Polymorphism (Overriding)
8. Write code for Exception handling
9. Write code for file handling and various file operations

UNIT I **9**

Tokens, Keywords, Identifiers and Constants, Basic Data Types, User-Defined and Derived Data Types, Type Casting, Implicit Conversion, Operators and Expressions, Operator Precedence, Simple statements, Conditional statements, Iterative statements, Array, Function, Pointer, Structure

UNIT II **9**

Basic Concepts of Object Oriented Programming, Object Oriented Programming Paradigm, Benefits of OOP, Object Oriented Languages, Class and Objects, Scope Resolution Operator, Access specifiers, Data members, Accessing class members, Data hiding, Member function, Inline function, Friend function, Passing objects as arguments, Returning objects from functions

UNIT III **9**

Constructors and its types, Destructor, Constructor overloading, Order of construction and destruction, Inheritance, Single, Multilevel, Multiple, Hierarchical, Hybrid Inheritance, Base class, Derived class, Virtual function, Polymorphism: Function Overloading, Operator Overloading, Overloading Unary Operators, Overloading Binary Operators, Overriding

UNIT IV **9**

Exception Handling, Throwing and Catching Mechanism, Templates, File handling, Types of files, End of File, Basic file operations: creating, opening, closing, reading, writing and appending a file, copying a file to another, Object oriented system development

List of Experiments:

Write programs to illustrate the following concepts:

1. Operators and expressions
2. Simple statements, Conditional statements and Iterative statements
3. Arrays
4. Functions
5. Pointers
6. Structures
7. Objects and Classes

8. Inline Function, Friend function and Virtual Functions
9. Scope Resolution Operator
10. Constructors and Destructors
11. Inheritance
12. Function Overloading
13. Operator Overloading
14. Dynamic or Run-time Polymorphism (Overriding)
15. Exception Handling
16. File operations

Text Books & References:

1. P. Deitel and H. Deitel, "C++ How to Program", Pearson.
2. E. Balagurusamy, "Object Oriented Programming with C++", TMH Publication.
3. Yashavant Kanetkar, "Let us C++", BPB Publications
4. Robert Lafore, "Object Oriented Programming in Turbo C++", Galgotia Publication.
5. B. Trivedi, "Programming with ANSI C++", Oxford University Press.
6. Ira Pohl, "Object Oriented Programming using C++", Pearson Education, Second Edition Reprint
7. B. Stroustrup, "The C++ Programming language", Pearson Education.
8. Timthy Budd, "An Introduction to Object Oriented Programming with C++," Addition-Wesley.
9. Kip R. Irvine, "C++ and Object-Oriented Programming," Prentice Hall.

BIT-14 SOFTWARE TOOLS-II 2 Credits (0-0-4)

Course Objectives:

The objectives of this course are:

1. To give overview of advanced C programming (command line arguments & file handling)
2. To give knowledge about practical usage of LINUX system calls
3. To develop capability of system call programming for practical understanding of operating system's internal functioning

Learning Outcomes:

The students are expected to be able to demonstrate the following knowledge, skills and attitudes after completing this course

1. Understanding of process handling in Operating system
2. Usage of kernel system calls
3. Understanding of multiprogramming function of operating system using system calls

List of Experiments:

1. Programs related to advanced C programming (command line arguments & file handling)
2. Understanding and usage of various LINUX kernel system calls
3. System call programming (programs based on LINUX kernel system calls using C language)

BIT-15 DESIGN & ANALYSIS OF ALGORITHM 5 Credits (3-1-2)

Course Objectives:

The course objectives of the course are:

1. To understand the importance of algorithm and how to analyze the complexity of algorithms.
2. To analyze the complexity of an algorithm in terms of time and space complexities
3. To design and implement various programming paradigms and its complexity
4. To program brute force, divide and conquer, greedy, and dynamic and backtracking techniques

Learning Outcomes:

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

1. Analyze the time and space complexity of a given algorithm

2. Apply the techniques of algorithm in solving real world problems
3. Systematic development of an algorithm for solving a problem

UNIT I **9**

Introduction: Algorithms, Analyzing Algorithms, Asymptotic Notation, Complexity of Algorithms, Growth of Functions, Performance Measurements, Solving Recurrence Equations Sorting and Order Statistics - Shell Sort, Heap Sort, Comparison of Sorting Algorithms, Sorting in Linear Time

UNIT II **9**

Divide and Conquer with examples such as Sorting- Quick Sort, Merge Sort, Matrix Multiplication, Convex hull and Searching.
Advanced Data Structures: Red-Black trees, B – trees, 2-3 Trees, Binomial Heaps, and Fibonacci Heaps

UNIT III **9**

Greedy methods with examples such as Optimal Reliability Allocation, Knapsack, Minimum Spanning Trees – Prim's and Kruskal's algorithms, Single Source Shortest Paths - Dijkstra's and Bellman Ford algorithms. Dynamic programming with examples such as Fibonacci Numbers, Multistage Graphs, Resource Allocation, Kanpsack, All Pair Shortest Paths – Warshal's and Floyd's algorithms

UNIT IV **9**

Backtracking Algorithms- Graph Coloring, n-Queen Problem, Hamiltonian Cycles and Sum of Subsets.
Selected Topics: String Matching, Theory of NP-completeness, Polynomial Time, Polynomial time Verification and Reducibility, NP – Hard - NP-Complete problems with examples. Approximation Algorithms Randomized Algorithms

List of Experiments:

1. To analyze time complexity of Insertion Sort, Merge Sort and Quick Sort.
2. To Implement Strassen's Matrix Multiplication.
3. To implement Merge Sort using Divide and Conquer approach.
4. To implement Quick Sort using Divide and Conquer approach.
5. To implement Knapsack Problem.
6. To implement Activity Selection Problem
7. To implement Dijkstra's Algorithm
8. To implement Bellman Ford's Prim's
9. To implement Kruskal's Algorithms.
10. To implement Largest Common Subsequence.
11. To implement Matrix Chain Multiplication.
12. To implement Multistage Graph Algorithms
13. To implement n-Queen Algorithms.
14. To implement Naïve String-Matching Algorithm.
15. To implement Rabin Karp String Matching Algorithm.

Text Books & References:

1. Ellis Horowitz, Sartaj Sahni and Sanguthevar Rajasekaran, Computer Algorithms/ C++, Second Edition, Universities Press, 2007.
2. T. H. Cormen, C. E. Leiserson, R.L.Rivest, and C. Stein, "Introduction to Algorithms", MIT Press, 3rd Edition, 2009.
3. Alfred V. Aho, John E. Hopcroft and Jeffrey D. Ullman, "The Design and Analysis of Computer Algorithms", Pearson Education, 1999.
4. S. Arora and B. Barak. Computational Complexity: A Modern Approach. Cambridge University Press, 2009.
5. S. Skiena. The Algorithm Design Manual. Springer, 2nd edition, 2008.
6. M. T. Goodrich and R. Tamassia. Data Structures and Algorithms in Java, Wiley, 5th edition, 2010.
7. J. Edmonds. How to Think About Algorithms. Cambridge University, Press, 2008.
8. M. R. Garey and D. S. Johnson. Computers and Intractability: A Guide to the Theory of NP-Incompleteness. V.H. Freeman, 1979.

BIT-16 COMPUTER ORGANIZATION & ARCHGITECTURE 5 Credits (3-1-2)

Course Objectives:

The course objectives of the course are:

1. To understand the basic structure and operation of digital computer.
2. To study the design of ALU and implementation of fixed point and floating-point arithmetic operations
3. To study the two types of control unit techniques and the concept of Pipelining
4. To study the hierarchical memory system including cache memories and virtual memory
5. To study the different ways of communicating with I/O devices and standard I/O interfaces

Learning Outcomes:

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

1. Discuss the basic concepts and structure of computers.
2. Understand concepts of register transfer logic and arithmetic operations.
3. Explain different types of addressing modes and memory organization.
4. Explain the function of each element of a memory hierarchy.
5. Identify and compare different methods for computer I/O.

UNIT I **9**

Register Transfer Language, Bus and Memory Transfers, Bus Architecture, Bus Arbitration, Arithmetic Logic, Shift Micro-operation, Arithmetic Logic Shift Unit, Design of Fast address, IEEE standard for Floating point numbers.

UNIT II **9**

Control Design: Hardwired & Micro Programmed Control Unit. **Processor Design:** Processor Organization: General register organization, Stack organization, Addressing mode, Instruction format, Data transfer & manipulations, Program Control, Reduced Instruction Set Computer, Pipelining.

UNIT III **9**

Arithmetic - Addition & subtraction of signed numbers - Multiplication - Integer division - Floating point operations, Decimal Arithmetic Unit, Decimal arithmetic operations.

UNIT IV **9**

Input-Output Organization: I/O Interface, Modes of transfer, Interrupts & Interrupt handling, Direct Memory access, Input-Output Processor, Serial Communication. **Memory Organization:** Memory Hierarchy, Main Memory (RAM and ROM Chips), Auxiliary memory, Cache memory, Virtual Memory

List of Experiments:

1. Implementing half adder, full adder using basic logic gates
2. Implementing Binary -to -Gray, Gray -to -Binary code conversions.
3. Implementing 3-8 line decoder.
4. Implementing 4x1 and 8x1 multiplexers.
5. Verify the excitation tables of various flip-flops.
6. Design of an 8-bit input/output system with four 8-bit internal registers.
7. Design of an 8-bit arithmetic logic unit.
8. Design the data path of a computer from its register transfer language description.
9. Implement a simple instruction set computer with a control unit and a data path.
10. Design the control unit of a computer using either hardwiring or microprogramming based on its register transfer language description.

Text Books & References:

1. Computer System Architecture, by M. Mano

2. Computer Organization, by Vravice, Zaky&Hamacher
3. Structured Computer Organization, by Andrew S. Tanenbaum
4. Computer Organization and Architecture: Designing for Performance by William Stallings

BIT-17 DATABASE MANAGEMENT SYSTEM 5 Credits (3-1-2)

Course Objectives:

The course objectives of the course are:

1. To list and define the fundamental concepts of database management system.
2. To execute manually a given (simple) database design and transaction over it.
3. To infer manually the type of a given (simple) database transaction.
4. To implement (simple) algorithms and data structures as database transaction.
5. To design (large) databases that are modular and have reusable components.
6. To explain how to apply concurrency control over concurrent database transactions.

Learning Outcomes:

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

1. Understand the basic concepts, techniques and terminology of the database management system.
2. Know and understand the basic syntax, semantics, and pragmatics of SQL & PL/SQL.
3. Analyse problems and apply DBMS concepts and techniques for developing the programs to solve them,
4. Evaluate alternative database designs to determine which are better according to selected criteria,
5. Know the concepts of Normalization and apply it to solve various problems related to database design
6. Know and understand the basic features of database transactions and concurrency control.
7. Reason about and manipulate concurrency control techniques.

UNIT I **9**

Introduction to DBMS: An overview of database management system, database system Vs file system, Database system concepts, Three-Level Architecture of DBMS, data models: Hierarchical Data Model, Network Data Model and Relational Data Model, schema and instances, data independence and data base language, Data definitions language, DML, Overall Database Structure. **Entity Relationship Model:** ER model concepts, notation for ER diagram, mapping constraints, keys, Concepts of Super Key, candidate key, primary key, Generalization, Specialization and aggregation, Translating E-R diagrams into tables and relational model, extended ER model, relationships of higher degree.

UNIT II **9**

Relational data Model and Language: Relational data model concepts, integrity constraints: entity integrity, referential integrity, Keys constraints, Domain constraints, relational algebra, relational calculus, tuple and domain calculus. **Introduction to SQL:** Characteristics of SQL, Advantage of SQL. SQL data types and literals. Types of SQL commands. SQL operators and their procedure. Tables, views and indexes. Queries and sub queries: Insert, update, delete, Joins, Unions, Intersection, Minus operations, SQL Functions: Number function, Character function, Aggregate functions, Conversion function, Date function.

UNIT III **9**

Data Base Design & Normalization: Functional dependencies and Multivalued dependency, Closure of a set of FDs, Minimal or Canonical cover of FDs, Normal form: first, second, third, BCNF, fourth and fifth normal forms along with examples, inclusion dependences, loss less join decompositions, dependency preserving decomposition, normalization using FD, MVD, and JDs, alternative approaches to database design.

UNIT IV **9**

Transaction Processing Concepts: Transaction system, Schedule, types of schedule, testing of serializability, Serializability of schedules, conflict & view serializable schedule, recoverability, Recovery from transaction failures, log-based recovery, checkpoints, deadlock handling, Deadlock detection, deadlock

prevention, Concurrency Control Techniques: Concurrency control, locking techniques for concurrency control, Time stamping protocols for concurrency control, validation-based protocol, multiple granularity, Multi version schemes, Recovery with concurrent transaction.

List of Experiments:

1. Creation and querying of database tables for the following cases.
    a. Write SQL queries using relational operations (=, <,>, etc)
    b. Write SQL queries using SQL operators.
    c. Write SQL query using character, number, date and group functions.
    d. Write SQL queries for relational algebra.
    e. Write SQL queries for extracting data from more than one table.
    f. Write SQL queries for sub queries and nested queries.
    g. Write programme by using PL/SQL.
    h. Concepts for ROLL BACK, COMMIT & CHECK POINTS
    i. Create VIEWS, CURSORS and TRGGERS & write ASSERTIONS.
    j. Create FORMS and REPORTS.
2. Development of applications involving vendor development systems, stores management system, finance management, Hotel management system, Hospital management system, Hostel management system etc.
3. Design of tables by normalization and dependency analysis.
5. Write application software with host language interface

Text Books & References:

1. Date C J, "An Introduction to Database System", Addision Wesley Publication.
2. Korth, Silbertz, Sudarshan, "Database Concepts", McGraw Hill Publication.
3. Elmasri, Navathe, "Fundamentals of Database Systems", Addision Wesley Publications
4. Leon & Leon, "Database Management System", Vikas Publishing House.
5. Bipin C. Desai, "An introduction to Database Systems", Galgotia Publication.
6. Majumdar & Bhattacharya, "Database Management System", TMH.
7. Ramakrishnan, Gehrke, "Database Management System", McGraw Hill.
8. Kroenke, "Database Processing: Fundamentals, Design and Implementation", Pearson Education.
9. Maheshwari Jain, "DBMS: Complete Practical Approach", Firewall Media, New Delhi.

BIT-18 Software Tools-III 2 Credits (0-0-4)

Course Objectives:

The objectives of this course are:

1. To give overview of all four LAMP components namely Linux, Apache Web Server, MySQL and PHP
2. To give knowledge about practical usage of LAMP technologies
3. To develop practical understanding of web programming for various real time applications

Learning Outcomes:

The students are expected to be able to demonstrate the following knowledge, skills and attitudes after completing this course:

1. Understanding the various functions of Linux Operating System
2. Usage of Apache Web Server, MySQL and PHP for creating different kinds of websites
3. Developing ability to create the desktop based and mobile based web applications

List of Experiments:

Students should install Linux and understand its various functions. Using Apache Web Server, MySQL and PHP, they can create the desktop based and mobile based web applications from following list of projects or any other creative projects with the concern of associated faculty in Lab.

1. University Management System
2. Attendance Management System
3. Examination Management System
4. Library Management System
5. Academic Registration Management System
6. Employee Management System
7. Student Management System
8. Grievance Management System
9. Hostel Management System
10. Mess Management System
11. Canteen Food Ordering and Management System
12. University Hospital Management System
13. Students' Departmental Society Portal
14. Students' Technical Event Management Portal
15. Students' Cultural Event Management Portal
16. Students' Sports Management Portal
17. Online Learning Management System
18. University Guest House Booking System
19. Online Faculty Staff Directory for Multi University
20. Security System using Biometric Authentication
21. Online Medicine Shopping
22. Online Food Ordering System
23. Online Auction System (for different valuables of the students who are leaving hostel)
24. Online Second-hand Book Buying and Selling Portal
25. Chatting and messaging system
26. Web-based chat application with webcam using PHP
27. Internet based Discussion Forum
28. University Alumni Portal
29. Online Book Recommendation System
30. Online Hotel Recommendation System
31. Advanced Intelligent Tourist Guide
32. Online Shopping
33. Online Banking System
34. Hotel Management System
35. Online Ticket Booking System
36. Online Hotel Booking System
37. Online Voting System
38. Online Mobile Recharge Portal
39. Online TV Recharge Portal
40. Creating Social Networking Site
41. Webpage ranking search engine

BIT-26 **OPERATING SYSTEM** **5 Credits (3-1-2)**

Course Objectives:

The course objectives of the course are:

1. To understand the structure and functions of OS.
2. To learn about Processes, Threads and Scheduling algorithms.
3. To understand the principles of concurrency and Deadlocks.
4. To learn various memory management schemes.
5. To study I/O management and File systems.

Learning Outcomes:

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

1. Describe how computing resources (such as CPU and memory) are managed by the operating system; describe the basic principles used in the design of modern operating systems.
2. Summarise the full range of considerations in the design of file systems, summarise techniques for achieving synchronisation in an operation system.
3. Explain the objective and functions of modern operating systems, explain memory hierarchy and cost-performance trade-offs, explain the operation, implementation and performance of modern operating systems, and the relative merits and suitability of each for complex user applications.
4. Compare and contrast the common algorithms used for both pre-emptive and non-pre-emptive scheduling of tasks in operating systems, such a priority, performance comparison, and fair-share schemes. Contrast kernel and user mode in an operating system.
5. Evaluate and report appropriate design choices when solving real-world problems.
6. Analyze the key trade-offs between multiple approaches to operating system design.

UNIT I **9**

Introduction: Basic architectural concepts, Operating System Services, interrupt handling, concepts of batch processing, multiprogramming, time-sharing, real-time operations; Resource Manager view, process view and hierarchical view of an OS. **Memory management**: Partitioning, paging, concepts of virtual memory demand paging, page replacement algorithms, segmentation, Segmentation and demand-paging, Cache memory management.

UNIT II **9**

Processor management: CPU scheduling – short-term, medium term and long term scheduling, non-preemptive and preemptive algorithms, performance analysis of multiprogramming, multiprocessing and interactive systems, Concurrent processes, precedence graphs, critical section problem, semaphores; Classical process, co-ordination problems, Producer consumer problem, Reader-writer problem, Dining philosophers problem, Barber's shop problem, Inter-process communication.

UNIT III **9**

Concurrent Programming: Critical region, conditional critical region, monitors, Deadlocks: prevention, avoidance, detection and recovery. Device Management: Scheduling algorithms – FCFS, shortest-seek-time-first, SCAN, CSCAN, LOOK, C-LOOK algorithms, spooling, spool management algorithm

UNIT IV **9**

Information Management: File concept, file support, directory structures, symbolic file directory, basic file directory, logical file system, physical file system, access methods, file protection, file allocation strategies. Protection: Goals, policies and mechanisms, domain of protection, access matrix and its implementation, access lists, capability lists, Lock/Key mechanisms, passwords, dynamic protection scheme, security concepts and public and private keys, RSA encryption and decryption algorithms. A case study: A UNIX OS file system, shell, filters, shell programming, programming with the standard I/O, UNIX system calls.

List of Experiments:

Write C programs for following:

1. CPU Scheduling Algorithms
2. File Allocation Strategies
3. Memory Management Techniques
4. File Organization Techniques
5. Deadlock Management Techniques
6. Page Replacement Algorithms
7. Process Synchronization

Text Books & References:

1. Silberschatz, Galvin And Gagne, Operating System Concepts (2012).

2. J. L. Peterson and A. Silberschatz: Operating Systems Concepts, Addison-Wesley, 1987.
3. M Deitel, Operating System
4. Andrew S. Tanenbaum,Modern Operating Systems (3rd Edition) (GOAL Series)
5. Charles Crowley, Operating Systems: A Design-Oriented Approach
6. Lubomir F. Bic and Alan C. Shaw, Operating Systems Principles
7. D Irtegov, Operating System Fundamentals (Programming Series)
8. Ramez Elmasri, A. Gil Carrick, and David Levine, Operating Systems: A Spiral Approach
9. Jean Bacon and Tim Harris,Operating Systems
10. William Stallings, Operating System

BIT-27 COMPUTER NETWORKS 5 Credits (3-1-2)

Course Objectives:

The course objectives of the course are:

1. To expose to the TCP/IP protocol suite.
2. To design, calculate, and apply subnet masks and addresses to fulfil networking requirements.
3. To identify security and privacy issues that relate to computer networks.
4. To solve mathematical problems of bandwidth, data rate, Hamming codes, cyclic redundancy check.

Learning Outcomes:

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

1. Provide insight about networks, topologies, and the key concepts
2. Know the basic concepts of network security and its various security issues related with each layer
3. Identify different types of communication mediums and techniques
4. Define and identify different types of multiplexing, data encoding, modulation and switching techniques
5. Illustrate different standards of Local Area Network in terms of technologies and hardware used
6. Illustrate network addressing and analysis techniques
7. Understand the Wide Area Network technologies
8. Understand the network routing concepts
9. Understand the internetworking concepts and architectures
10. Understand the TCP/IP protocols and design architectures

UNIT I **9**

Introductory Concepts - Network Hardware - Network software, Networks Topologies. Layering and Protocols, Switching Methods, LAN Inter Connection Devices - Physical Layer - Different types of Transmission Media, Errors in Transmission: attenuation, noise

UNIT II **9**

MAC Layer: Channel Allocation Problem – Aloha, CSMA, CSMA/CD, CSMA/CA Protocols. Examples: Ethernet, including Gigabit, IEEE standards, FDDI.

Data Link Layer: Framing, Error Detection (Parity, CRC), Sliding Window, Stop and Wait protocols

UNIT III **9**

Network Layer - Design issues, Routing Algorithms: Congestion Control Algorithms - Quality of Service, Distance Vector, Link State, Inter-domain Routing. Internet Protocol, IPv6, ARP, DHCP, ICMP, Subnetting, Classless Addressing, Network Address Translation

UNIT IV **9**

Transport Layer - Design issues, Elements of Transport Protocols - User Datagram Protocol -Transmission Control Protocol, Connection Establishment and Termination.

Session, Presentation, and Application Layers - Examples: DNS - Electronic mail - World Wide Web - Multimedia - Network Security

List of Experiments:

1. To create scenario and study the performance of CSMA/CD protocol through simulation.
2. To create scenario and study the performance of token bus and token ring protocols through simulation.
3. Implementation of Error detection and correction algorithms.
4. Implementation and study of 1-bit sliding window viz., stop and wait protocol.
5. Implementation and study of Go-BackN protocol.
6. Implementation and study of selective repeat protocol.
7. To get the MAC or Physical address of the system using Address Resolution Protocol.
8. Implementation of distance vector routing algorithm.
9. Implementation of link state routing algorithm.
10. To write a clientserver application for chat using TCP.
11. To write a C program to develop a DNS client server to resolve the given hostname.

Text Books & References:

1. S. Tanenbaum, "Computer Networks", Pearson Education, Fourth Edition, 2003
2. Behrouz A. Foruzan, "Data Communication and Networking", Tata McGraw Hill, 2004
3. LL Peterson, BS Davie, Computer Networks: A Systems Approach, 5th Ed., Morgan-Kauffman, 2011.
4. Comer, Computer Networks & Internet with Internet Applications by Pearson Education
5. JF Kurose, KW Ross, Computer Networking: A Top-Down Approach, 5th Ed., Addison-Wesley, 2009.
6. W Stallings, Cryptography and Network Security, Principles and Practice, 5th Ed., Prentice-Hall, 2010
7. W. Stallings, "Data and Computer Communication", Pearson Education, Fifth Edition.

BIT-28 SOFTWARE ENGINEERING 5 Credits (3-1-2)

Course Objectives:

The course objectives of the course are:

1. To understand requirement analysis, feasibility study, software development, testing, maintenance
2. To enhance the software project development and management skills.
3. To develop functioning software that is benchmark to the international standards.

Learning Outcomes:

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

1. List and define the fundamental concepts of Software Engineering and its applications.
2. Design and develop the DFD, E-R Diagram, Flow Chart etc for a project.
3. Design and develop the SRS for a project, ISO 9000 Models, SEI-CMM Model etc.
4. List and define the test cases, fundamental concepts of testing to be applied on various projects
5. Implement and analyse the various model of software development for a project.
6. Design and define the cost estimations, size estimations using various techniques.
7. List and define the fundamental concepts of Software maintenance
8. Explain the CASE tool, Reverse Engineering, Re-Engineering, Software risk analysis and management

UNIT I **9**

Introduction: Introduction to Software Engineering, Evolution and impact of Software engineering, Software Components, Software Characteristics, Software Crisis, Similarity and Differences from Conventional Engineering Processes, Feasibility study, Functional and Non-functional requirements, Requirements gathering, Requirements analysis, Software Development Life Cycle (SDLC) Models: Waterfall Model, Prototype Model, Spiral Model, Evolutionary Development Models, Iterative Enhancement Models.

UNIT II **9**

Software Requirement Specifications (SRS): Requirement Engineering Process: Elicitation, Analysis, Documentation, Data Flow Diagrams, Entity Relationship Diagrams, Decision Tables, SRS Document, IEEE Standards for SRS, Basic issues in software design: modularity, Top-Down and Bottom-Up Design, Cohesion and Coupling, Structure chart, Object-oriented software development, Software Quality Assurance (SQA):

Verification and Validation, SQA Plans, Software Quality Frameworks, ISO 9000 Models, SEI-CMM Model.

UNIT III **9**

Software Testing: Fundamental of testing, Testing Objectives, Unit Testing, Integration Testing, Acceptance Testing, Regression Testing, Testing for Functionality and Testing for Performance, Top Down and Bottom-Up Testing Strategies: Test Drivers and Test Stubs, Structural Testing (White Box Testing), Functional Testing (Black Box Testing), Test Data Suit Preparation, Alpha and Beta Testing of Products. Static Testing Strategies: Formal Technical Reviews (Peer Reviews), Walk Through, Code Inspection, Software Reliability Metrics.

UNIT IV **9**

Software Maintenance and Software Project Management: Need of Software Maintenance, Categories of Maintenance: Preventive, Corrective and Perfective Maintenance, Cost of Maintenance, Software Re-Engineering, Reverse Engineering. Software Configuration Management Activities, Change Control Process, Software Version Control, An Overview of CASE Tools, Estimation of Various Parameters such as Cost, Efforts, Schedule/Duration, Constructive Cost Models (COCOMO), Resource Allocation Models, Software Risk Analysis and Management.

List of Experiments:

Design and draw the SRS, ER diagram, DFD and develop the software project for the followings:

1. Hospital Management
2. Hotel management
3. University Management
4. Library Management
5. Time Table Management
6. Student Result Management
7. Attendance Management
8. Hostel Management
9. Mess Management
10. Banking Management
11. Inventory Management

Text Books & References:

1. RS Pressman, Software Engineering: A Practitioners Approach, McGraw Hill.
2. Pankaj Jalote, Software Engineering, Wiley
3. Rajib Mall, Fundamentals of Software Engineering, PHI Publication.
4. KK Aggarwal and Yogesh Singh, Software Engineering, New Age International Publishers.
5. Ghezzi, M. Jarayeri, D. Manodrioli, Fundamentals of Software Engineering, PHI Publication.
6. Ian Sommerville, Software Engineering, Addison Wesley.
7. Kassem Saleh, "Software Engineering", Cengage Learning.
8. P fleeger, Software Engineering, Macmillan Publication

BIT-29 **AUTOMATA THEORY** 4 **Credits (3-1-0)**

Course Objectives:

The course objectives of the course are:

1. To learn fundamentals of Regular and Context Free Grammars and Languages.
2. To understand the relation between Regular Language and Finite Automata and machines.
3. To learn how to design Automata's and machines as Acceptors, Verifiers and Translators.
4. To understand the relation between Contexts free Languages, PDA and TM.
5. To learn how to design PDA as acceptor and TM as Calculators.
6. To learn how to co-relate Automata's with Programs and Functions.

Learning Outcomes:

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

1. Understand, design, construct, analyze and interpret Regular languages, Expression and Grammars.
2. Design different types of Finite Automata and Machines as Acceptor, Verifier and Translator.
3. Understand, design, analyze and interpret Context Free languages, Expression and Grammars.
4. Understand, design, analyze and interpret Context Free languages, Expression and Grammars
5. Design different types of Push down Automata as Simple Parser.
6. Design different types of Turing Machines (Acceptor, Verifier, Translator & Basic computing machine).
7. Compare, understand and analyze different languages, grammars, Automata and Machines and appreciate their power and convert Automata to Programs and Functions

UNIT I **9**

Alphabets, Strings and Languages, Automata and Grammars, Deterministic Finite Automata (DFA)-Formal Definition, Simplified Notation: State Transition Graph, Transition Table, Language of DFA, Nondeterministic Finite Automata (NFA), NFA with Epsilon Transition, Equivalence of NFA and DFA, Minimization of Finite Automata, Myhill-Nerode Theorem

UNIT II **9**

Chomsky Classification of Grammars, Regular Expression, Definition, Operators of Regular Expression and their Precedence, Algebraic Laws for Regular Expressions, Kleen's Theorem, Regular Expression to FA, DFA to Regular Expression, Arden Theorem, Non Regular Languages, Pumping Lemma for Regular Languages. Application of Pumping Lemma, Closure Properties of Regular Languages, Decision Properties of Regular Languages, FA with Output: Moore and Mealy Machine, Equivalence of Moore and Mealy Machine, Applications and Limitation of FA.

UNIT III **9**

Context Free Grammar (CFG) and Languages: Definition, Examples, Parsing, Derivation, Derivation Trees, Ambiguity in Grammar, Inherent Ambiguity, Ambiguous to Unambiguous CFG, Useless Symbols, Simplification of CFGs, Normal Forms for CFGs: CNF and GNF, Closure Proper ties of CFLs, Decision Properties of CFLs: Emptiness, Finiteness and Membership, Pumping Lemma for CFLs. Context Sensitive Grammar and Language. **Push Down Automata (PDA):** Description and Definition, Instantaneous Description, Language of PDA, Acceptance by Final State, Acceptance by Empty Stack, Deterministic PDA, Equivalence of PDA and CFG, CFG to PDA and PDA to CFG.

UNIT IV **9**

Turing Machines (TM): Basic Model, Definition and Representation, Instantaneous Description, Language Acceptance by TM, Variants of Turing Machine, TM as Computer of Integer Functions, Universal TM, Church's Thesis, Recursive and Recursively Enumerable Languages, Halting Problem, Introduction to Undesirability, Undecidable Problems about TMs. Post Correspondence Problem (PCP), Modified PCP, Multi-Tape Turing Machine.

Text Books & References:

1. Introduction to Automata Theory, Languages, and Computation, 3nd Edition, John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, Pearson Education
2. Micheal Sipser, Introduction to the Theory of Computation, Thomson Learning
3. Theory of Computer Science – Automata languages and computation, Mishra and Chandrashekaran, 2nd edition, PHI.
4. H R. Lewis and Christos H. Papadimitriou, Elements of the theory of Computation, PHI Ltd
5. Introduction to Formal languages Automata Theory and Computation Kamala Krithivasan, Rama R, Pearson.