Raja N. L. Khan Women's College (Autonomous)

Syllabus and Scheme of Examination

for

B. C. A. (Honours)

Under

Choice Based Credit System

Raja Narendra Lal Khan Women's College (Autonomous)

December 2021

Course Structure of Bachelor of Computer Applications (Honours)

Course Type	Credit	Marks
I. Core Course Theory and Practical (14 Papers)	14 ´ 6 Credits = 84 Credits	14 ´ (Theory Exam + Practical Exam + Internal Evaluation) = 14 ´ (40 + 20 + 15) = 14 ´ 75 Marks = 1050 Marks
III. Discipline Specific Elective Theory and Practical (4 Papers)	4 ´ 6 Credits = 24 Credits	4 ´ 75 = 300 Marks
IV. Generic Elective Theory and Practical (4 Papers)	4 ´ 6 Credits = 24 Credits	4 ´ 75 = 300 Marks
V. Ability Enhancement Compulsory Theory and Practical (2 Papers of 2 credits and 4 credits respectively) 1. English Communication, 2. Environmental Science	2 + 4 Credits = 6 Credits	50 + 100 = 150 Marks
VI. Ability Enhancement Elective (Skill Based Theory and Lab) (2 Papers of 2 credit each)	2 ´ 2 Credits = 4 Credits	2 ´ 50 = 100 Marks
Total	Total Credit: 142	Total Marks: 1900

Marks Distribution Details- For Each Paper:

Theory/Practical/Internal Evaluation		Marks
Theory	Core /DSE / GE	40
	SEC / AEC	20
Practical / Tutorial	Core /DSE / GE	20 (Practical Experiment – 15, Viva – 3, Laboratory Note Book – 2). For Tutorial: Assignments – 10, Presentations – 5, Viva – 5.
	SEC / AEC	20 (Practical Experiment – 15, Viva – 3, Laboratory Note Book – 2). For Tutorial: Assignments – 15, Presentations – 3, Viva – 2.
Internal Evaluation	Core /DSE / GE	15 (Attendance – 5 Marks, Assignments/Presentations/Class Tests – 10 Marks)
	SEC / AEC	10 (Attendance – 5 Marks, Assignments/Presentations/Class Tests – 5 Marks) for 50 marks paper.

Lecture / Practical / Tutorial Contact Hours Distribution Details:

Theory/Practical/Tutorial		Number of Lecture Hours per Week
Theory	Core / DSE / GE	4
	AEC	2
	SEC	1
Practical / Tutorial	Core / DSE / GE	4
	AEC	1
	SEC	2

Syllabus Structure of BCA (Honours)

SEM.	Course Code	**CORE COURSE** (14) * *Number within parenthesis indicates total number of courses*	**Ability Enhancement Compulsory Course** (AECC) (2)	**Skill Enhancement Elective Course** (SEC) (2)	**Elective: Discipline Specific** (DSE) (4)	**Elective: Generic** (GE) (4)
I	**CC-1**	Programming Fundamentals using C and C++ Theory + Practical (4 + 2) * *Number within parenthesis indicates credit point of the course*	English Communications – Theory + Practical (1+1)	None	None	GE – 1 – Theory + Practical (4+2)
	CC-2	Computer Fundamental and Digital Electronics Theory + Practical (4 + 2)				
II	**CC-3**	Programming in JAVA Theory + Practical (4 + 2)	Environmental Science – Theory + Practical (1+1)	None	None	GE-2 – Theory + Practical (4+2)
	CC-4	Computer Organization and Architecture Theory (4 + 2) PC Assembly and Software Installation Practical (2)				
III	**CC-5**	Data Structures Theory + Practical (4 + 2)	None	SEC-1 – Theory + Practical (1+1)	None	GE-3 – Theory + Practical (4+2)
	CC-6	Operating Systems Theory + Practical (4 + 2)				
	CC-7	Computer Networks Theory + Practical (4 + 2)				

IV	**CC-8**	Design and Analysis of Algorithms Theory + Practical (4 + 2)	None	SEC-2 – Theory + Practical (1+1)	None	GE-4 – Theory + Practical (4+2)
	CC-9	Software Engineering – Theory + Practical (4 + 2)				
	CC-10	Database Management System Theory + Practical (4 + 2)				
V	**CC-11**	Web Technologies Theory + Practical (4 + 2)	None	None	DSE -1 – Theory + Practical (4 + 2)	None
	CC-12	Introduction to Data Science Theory + Practical (4 + 2)			DSE-2 – Theory + Practical (4 + 2)	
VI	**CC-13**	Cyber Security and Cyber Laws Theory + Practical (4 + 2)	None	None	DSE-3 – Theory + Practical (4 + 2)	None
	CC-14	Computer Graphics and Multimedia Theory + Practical (4 + 2)			DSE-4 – Theory + Practical (4 + 2)	
CREDIT		84	4	4	24	24
TOTAL CREDIT		140				

Discipline Specific Elective Papers: (4 Papers to be selected) – DSE 1, DSE 2, DSE 3, DSE 4:

1. Information Security (4 Credit) + Lab (2 Credit)
2. Digital Image Processing (4) + Lab (2)
3. Machine Learning (4) + Lab (2)
7. E-Commerce (4) + Lab (2)
8. Optimization Techniques (4) + Lab (2)
9. Cloud Computing (4) + Lab (2)
10. Big Data Analytics (4) + Lab (2)
11. Microprocessors (4) + Lab (2)
12. Graph Theory (4) + Lab (2)
13. Project Work / Dissertation (6 Credit)

Skill Enhancement Courses (2 Papers to be selected) – SEC 1, SEC 2:

1. Android Programming Lab (2 Credit)
2. Programming in MATLAB Lab (2)
3. HTML Programming Lab (2)

4. Programming in Python Lab(2)

6. PHP Programming Lab (2)

7. R Programming Lab (2)

8. UNIX / LINUX Programming Lab (2)

Generic Elective from Other Discipline (Four Papers to be opted: Two from one discipline, Two more from another discipline) – GE 1, GE 2, GE 3, GE 4

Any two disciplines to be chosen from the following four disciplines:

1. Mathematics

2. Economics

3. Geography

4. Psychology

5. English

Semester-wise Course Structure with Number of Contact Hours

Sem	Course Type, Name	Paper Code	Paper Name	Teaching Hours Per Week			Credit	Marks			
				L*	T	P		Exam	Internal	Attendance	Total
I	Core - 1	C1T	Programming Fundamentals using C & C++	3	1	0	4	40	10	5	75
		C1P	Programming Fundamentals using C & C++ Lab.	0	0	4	2	20			
	Core - 2	C2T	Computer Fundamental & Digital Electronics	3	1	0	4	40	10	5	75
		C2P	Digital Electronics Lab.	0	0	4	2	20			
	Generic-1	GE1T	Generic Paper Theory	3	1	0	6	60	10	5	75
	Ability Enhancement Compulsory Course-1	AEC1	Communicative English								50
	Total						20				275

* L: Lecture, T: Tutorial, P: Practical

Sem	Course Type, Name	Paper Code	Paper Name	Teaching Hours Per Week			Credit	Marks			
				L*	T	P		Exam	Internal	Attendance	Total
II	Core - 3	C3T	Programming in JAVA	3	1	0	4	40	10	5	75
		C3P	Programming in JAVA Lab.	0	0	4	2	20			
	Core - 4	C4T	Computer Architecture and Organization	3	1	0	4	40	10	5	75
		C4P	PC Assembly and software Installation Lab.	0	0	4	2	20			
	Generic - 2	GE2T	Generic Paper Theory	3	1	0	6	60	10	5	75
	Ability Enhancement Compulsory Course - 2	AEC2	Environment Science				4	50	40	10	100
	Total						20				275

Sem	Course Type, Name	Paper Code	Paper Name	Teaching Hours Per Week			Credit	Marks			
				L*	T	P		Exam	Internal	Attendance	Total
III	Core - 5	C5T	Data Structures	3	1	0	4	40	10	5	75
		C5P	Data Structures Lab.	0	0	4	2	20			
	Core - 6	C6T	Operating Systems	3	1	0	4	40	10	5	75
		C6P	Operating Systems Lab.				4				
	Core - 7	C7T	Computer Networks	3	1	0	4	40	10	5	75
		C7P	Computer Networks Lab.	0	0	4	2	20	5		
	Generic - 3	GE3T	Generic Paper Theory	3	1	0	6	60	10	5	75
	Skill Enhancement Course - 1	SEC1T	SEC Paper Theory	1	0	0	2	20	5	5	50
		SEC1P	SEC Paper Lab.	0	0	1		20			
	Total						26				275

Sem	Course Type, Name	Paper Code	**Paper Name**	Teaching Hours Per Week			Credit	Marks			
				L*	T	P		Exam	Internal	Attendance	Total
IV	Core - 8	C8T	Design and Analysis of Algorithms	3	1	0	4	40	10	5	75
		C8P	Design and Analysis of Algorithms Lab.	0	0	4	2	20			
	Core - 9	C9T	Software Engineering	3	1	0	4	40	10	5	75
		C9P	Software Engineering Lab.	0	0	4	2	20			
	Core - 10	C10T	Database Management System	3	1	0	4	40	10	5	75
		C10P	Database Management System Lab.	0	0	4	2	20			
	Generic - 4	GE4T	Generic Paper Theory	3	1	0	6	60	10	5	75
	Skill EnhancementCourse - 2	SEC2T	SEC Paper Theory	1	0	0	2	20	5	5	50
		SEC2P	SEC Paper Laboratory	0	0	1		20			
	Total						26				275

Sem	Course Type, Name	Paper Code	**Paper Name**	Teaching Hours Per Week			Credit	Marks			
				L	T	P		Exam	Internal	Attendance	Total
V	Core - 11	C11T	Web Technologies	3	1	0	4	40	10	5	75
		C11P	Web Technologies Laboratory	0	0	4	2	20			
	Core - 12	C12T	Introduction to Data Science	3	1	0	4	40	10	5	75
		C12P	Introduction to Data Science Practical				4	20			
	Discipline Specific Elective - 1	DSE1 T	DSE1 Theory	3	1	0	4	40	10	5	75
		DSE1 P	DSE1 Practical	0	0	4	2	20			
	Discipline Specific Elective - 2	DSE2 T	DSE2 Theory	3	1	0	4	40	10	5	75
		DSE2 P	DSE2 Practical	0	0	4	2	20			
	Total						24				300

Sem	Course Type, Name	Paper Code	**Paper Name**	Teaching Hours Per Week			Credit	Marks			
				L*	T	P		Exam	Internal	Attendance	Total
VI	Core - 13	C13T	Cyber Security and Cyber Laws	3	1	0	4	40	10	5	75
		C13P	Cyber Security and Cyber Laws Practical	0	0	4	2	20			
	Core - 14	C14T	Computer Graphics and Multimedia	3	1	0	4	40	10		75
		C14P	Computer Graphics and Multimedia Practical				2				
	Discipline Specific Elective - 3	DSE3 T	DSE3 Theory	3	1	0	4/6	40	10	5	
		DSE3 P	DSE3 Practical	0	0	4/0	2/0	20			
	Discipline Specific Elective - 4	DSE4 T	DSE4 Theory	3	1	0	4/6	40/60	10	5	75
		DSE4 P	DSE4 Laboratory	0	0	4/0	2/0	20/0			
	Total						24				300

Discipline Specific Elective Papers: (4 Papers to be selected) – DSE 1, DSE 2, DSE 3, DSE 4:

1. Information Security (4 Credit) + Lab (2 Credit)
2. Digital Image Processing (4) + Lab (2)
3. Machine Learning (4) + Lab (2)
4. Numerical Methods (4) + Lab (2)
5. Optimization Techniques (4) + Lab (2)
6. Cloud Computing (4) + Lab (2)
7. Big Data Analytics (4) + Lab (2)
8. Microprocessors (4) + Lab (2)
9. Graph Theory (4) + Lab (2)
10. Numerical Methods (4) + Lab (2)

10. Seminar (6 Credit)
11. Project Work / Dissertation (6 Credit)

Skill Enhancement Courses (2 Papers to be selected) – SEC 1, SEC 2:

1. Android Programming (2 Credit)
2. Programming in MATLAB (2)
3. HTML Programming (2)
4. Programming in Python (2)
6. PHP Programming (2)
7. R Programming (2)
8. UNIX / LINUX Programming (2)

Programme Objectives (PO)

PO 1: To provide students with a solid foundation in Computer Application fundamentals necessary to analyze the requirements of the software, design and create innovative software products and solutions for real life problems.

PO 2: To provide exposure to emerging technologies to work as teams on multidisciplinary

PO 3: To prepare the students for a successful professional career as developer, scientist, teacher, administrator or an entrepreneur and work with values & social concern bridging the digital divide and meeting the requirements of Indian and multinational companies.

C1T: Programming Fundamentals using C and C++ Theory: 60 Lectures

	Course Outcome
1.	Know the procedural and object-oriented programming (OOP) paradigms
2.	Develop the skill to understand programming logic for solving a computing problem
3.	Understand syntax of C and C++ programming language
4.	Apply procedural programming skill to solve a given problem using C
5.	Develop skill to identify classes and its members from a given problem statement
6.	Apply the key features of OOP to solve a given problem using C++

Unit 1: **Introduction to C and C++** **(3 Lectures)**

History of C and C++, Overview of Procedural Programming and Object-Orientation Programming, Using main() function, Compiling and Executing Simple Programs in C++.

Unit 2: **Data Types, Variables, Constants, Operators and Basic I/O** **(5 Lectures)**

Declaring, Defining and Initializing Variables, Scope of Variables, Using Named Constants, Keywords, Data Types, Casting of Data Types, Operators (Arithmetic, Logical and Bitwise), Using Comments in programs, Character I/O (getc, getchar, putc, putchar etc), Formatted and Console I/O (printf(), scanf(), cin, cout), Using Basic Header Files (stdio.h, iostream.h, conio.h etc).

Unit 3: **Expressions, Conditional Statements and Iterative Statements** **(5 Lectures)**

Simple Expressions in C++ (including Unary Operator Expressions, Binary Operator Expressions), Understanding Operators Precedence in Expressions, Conditional Statements (if construct, switch-case construct), Understanding syntax and utility of Iterative Statements (while, do-while, and for loops), Use of break and continue in Loops, Using Nested Statements (Conditional as well as Iterative)

Unit 4: **Functions and Arrays** **(10 Lectures)**

Utility of functions, Call by Value, Call by Reference, Functions returning value, Void functions, Inline Functions, Return data type of functions, Functions parameters, Differentiating between Declaration and Definition of Functions, Command Line Arguments/Parameters in Functions, Functions with variable number of Arguments.

Creating and Using One Dimensional Arrays (Declaring and Defining an Array, initializing an Array, accessing individual elements in an Array, manipulating array elements using loops), Use Various types of arrays (integer, float and character arrays / Strings) Two-dimensional Arrays (Declaring, Defining and Initializing Two Dimensional Array, Working with Rows and Columns), Introduction to Multi-dimensional arrays

Unit 5: **Derived Data Types (Structures and Unions)** **(3 Lectures)**

Understanding utility of structures and unions, Declaring, initializing and using simple structures and unions, manipulating individual members of structures and unions, Array of Structures, Individual data members as structures, Passing and returning structures from functions, Structure with union as members, Union with structures as members.

Unit 6: **Pointers and References in C++** **(7 Lectures)**

Understanding a Pointer Variable, Simple use of Pointers (Declaring and Dereferencing Pointers to simple variables), Pointers to Pointers, Pointers to structures, Problems with

Pointers, passing pointers as function arguments, returning a pointer from a function, using arrays as pointers, Passing arrays to functions. Pointers vs. References, Declaring and initializing references, Using references as function arguments and function return values

Unit 7: **Memory Allocation in C++** **(3 Lectures)**

Differentiating between static and dynamic memory allocation, use of malloc, calloc and free functions, use of new and delete operators, storage of variables in static and dynamic memory allocation

Unit 8: **File I/O, Preprocessor Directives** **(4 Lectures)**

Opening and closing a file (use of fstream header file, ifstream, ofstream and fstream classes), Reading and writing Text Files, Using put(), get(), read() and write() functions, Random access in files, Understanding the Preprocessor Directives (#include, #define, #error, #if, #else, #elif, #endif, #ifdef, #ifndef and #undef), Macros

Unit 9: **Using Classes in C++** **(7 Lectures)**

Principles of Object-Oriented Programming, Defining & Using Classes, Class Constructors, Constructor Overloading, Function overloading in classes, Class Variables &Functions, Objects as parameters, Specifying the Protected and Private Access, Copy Constructors,Overview of Template classes and their use.

Unit 10: **Overview of Function Overloading and Operator Overloading** **(5 Lectures)**

Need of Overloading functions and operators, Overloading functions by number and type of arguments, Looking at an operator as a function call, Overloading Operators (including assignment operators, unary operators)

Unit 11: **Inheritance, Polymorphism and Exception Handling** **(8 Lectures)**

Introduction to Inheritance (Multi-Level Inheritance, Multiple Inheritance), Polymorphism (Virtual Functions, Pure Virtual Functions), Basics Exceptional Handling (using catch and throw, multiple catch statements), Catching all exceptions, Restricting exceptions, Rethrowing exceptions.

Recommended Books

1. Dey, Bandyopadhyay, *C Programming Essentials*, Pearson Education.

2. Balagurusamy, *Programming in ANSI C*, Tata McGraw Hill.

3. Kerninghan, Ritchie, *The C Programming Language*, Pearson.

4. Venugopal, *Masterning C*, Pearson.

5. Kanetkar, *Let Us C*, BPB Publications.

6. Ghosh, *All of C*, Prentice Hall India.

7. Thareja, *Programming in C*, Oxford University Press.

8. Herbert Schildt, *C++: The Complete Reference*, 4th Edition, McGraw Hill.

9. Stroustrup, *The C++ Programming Language*, 4th Edition, Pearson.

10. Lafore, *Object-Oriented Programming in C++*, 4th Edition, Pearson.

11. Balagurusamy, *Object Oriented Programming with C++*, TMH.

12. Shukla, *Object Oriented Programming in C++*, Wiley.

14. Meyers, *Effective C++*, 3rd Edition, Pearson.

C1P: Programming Fundamentals using C/C++ Lab — Practical: 60 Lectures

	Course Outcome
1.	Ability to write and execute C programs for solving a given problem
2.	Ability to write and execute C++ programs for solving a given problem
3.	Comprehend the compilation error messages and debugging
4.	Skill to trace execution of a program to fix errors in programing logic

1. Write a program (WAP) to print the sum and product of digits of an integer.

2. WAP to reverse a number.

3. WAP to compute the sum of the first n terms of the following series
S = 1+1/2+1/3+1/4+……

4. WAP to compute the sum of the first n terms of the following series
S =1-2+3-4+5…….

5. Write a function that checks whether a given string is Palindrome or not. Use this function to find whether the string entered by user is Palindrome or not.

6. Write a function to find whether a given no. is prime or not. Use the same to generate the prime numbers less than 100.

*7.*WAP to compute the factors of a given number.

8. Write a macro that swaps two numbers. WAP to use it.

9. WAP to print a triangle of stars as follows (take number of lines from user):

```
*
***
*****
*******
```

10. WAP to perform following actions on an array entered by the user:
i) Print the even-valued elements
ii) Print the odd-valued elements
iii) Calculate and print the sum and average of the elements of array

iv) Print the maximum and minimum element of array
v) Remove the duplicates from the array
vi) Print the array in reverse order

The program should present a menu to the user and ask for one of the options. The menu should also include options to re-enter array and to quit the program.

11. WAP that prints a table indicating the number of occurrences of each alphabet in the text entered as command line arguments.

12. Write a program that swaps two numbers using pointers.

13. Write a program in which a function is passed address of two variables and then alter its contents.

14. Write a program which takes the radius of a circle as input from the user, passes it to another function that computes the area and the circumference of the circle and displays the value of area and circumference from the main() function.

15. Write a program to find sum of n elements entered by the user. To write this program, allocate memory dynamically using malloc() / calloc() functions or new operator.

16. Write a menu driven program to perform following operations on strings:

a) Show address of each character in string
b) Concatenate two strings without using strcat function.
c) Concatenate two strings using strcat function.
d) Compare two strings
e) Calculate length of the string (use pointers)
f) Convert all lowercase characters to uppercase
g) Convert all uppercase characters to lowercase
h) Calculate number of vowels
i) Reverse the string

17. Given two ordered arrays of integers, write a program to merge the two-arrays to get an ordered array.

18. WAP to display Fibonacci series (i)using recursion, (ii) using iteration

19. WAP to calculate Factorial of a number (i)using recursion, (ii) using iteration

20. WAP to calculate GCD of two numbers (i) with recursion (ii) without recursion.

21. Create Matrix class using templates. Write a menu-driven program to perform following Matrix
operations (2-D array implementation):
a) Sum b) Difference c) Product d) Transpose

22. Create the Person class. Create some objects of this class (by taking information from the user). Inherit the class Person to create two classes Teacher and Student class. Maintain the respective information in the classes and create, display and delete objects of these two classes (Use Runtime Polymorphism).

23. Create a class Triangle. Include overloaded functions for calculating area. Overload assignment operator and equality operator.

24. Create a class Box containing length, breath and height. Include following methods in it:
a) Calculate surface Area
b) Calculate Volume
c) Increment, Overload ++ operator (both prefix & postfix)
d) Decrement, Overload -- operator (both prefix & postfix)
e) Overload operator == (to check equality of two boxes), as a friend function
f) Overload Assignment operator
g) Check if it is a Cube or cuboid

Write a program which takes input from the user for length, breath and height to test the above

class.

25. Create a structure Student containing fields for Roll No., Name, Class, Year and Total Marks. Create 10 students and store them in a file.

26. Write a program to retrieve the student information from file created in previous question and print it in following format: *Roll No. Name Marks*

27. Copy the contents of one text file to another file, after removing all white spaces.

28. Write a function that reverses the elements of an array in place. The function must accept only one pointer value and return void.

29. Write a program that will read 10 integers from user and store them in an array. Implement array using pointers. The program will print the array elements in ascending and descending order.

C2T: Computer Fundamental and Digital Electronics Theory: 60 Lectures

C2T:	Course Outcome
1.	Illustrate different types of software and the concept of I/O devices and their evolution
2.	Illustrate binary arithmetic, code conversion; and solve Boolean logic minimization
3.	Improve the combinational and sequential circuit design and minimization techniques
4.	Design the fundamental combinational and sequential logic circuits; and counters and registers
5.	Illustrate memory organization and CPU
6.	Discuss about Emerging Technologies such as cloud computing, big data, data mining etc.

Unit 1: Introduction - Introduction to computer system, uses, types. (2 Lectures)

Unit 2: Human Computer Interface - Types of software, Operating system as user interface, utility programs, algorithm, flowchart. (8 Lectures)

Unit 3: Devices - Input and output devices (with connections and practical demo), keyboard, mouse, joystick, scanner, OCR, OMR, bar code reader, web camera, monitor, printer, plotter. (4 Lectures)

Unit 4: Memory- Primary, secondary, auxiliary memory, RAM, ROM, cache memory, hard disks, optical disks. (6 Lectures)

Unit 5: Computer Organization and Architecture - C.P.U., registers, system bus, main memory unit, cache memory, Inside a computer, SMPS, Motherboard, Ports and Interfaces, expansion cards, ribbon cables, memory chips, processors. (8 Lectures)

Unit 6: Overview of Emerging Technologies: (3 Lectures)
Bluetooth, cloud computing, big data, data mining, mobile computing and embedded systems.

Unit 7: Use of Computers in Education and Research - Data analysis, Heterogeneous storage, e-Library, Google Scholar, Domain specific packages such as SPSS, Mathematica etc. (2 Lectures)

Unit 8: Introduction of System programming- Assembler, Linker, Loader, Compiler, Interpreter, software tools. (5 Lectures)

Unit 9: Introduction to digital electronics- Logic gates, Boolean algebra, combinational circuits, circuit simplification, flip-flops and sequential circuits, decoders, multiplexers, registers, counters and memory units. (12 Lectures)

Unit 10: Data Representation and Basic Computer Arithmetic- Number systems, complements, fixed and floating point representation, character representation, addition, subtraction, magnitude comparison, multiplication and division algorithms for integers. (10 Lectures)

Recommended Books

1. Rajaraman, Adabala, *Fundamentals of* Computers, PHI.
2. Sinha, Sinha, *Computer Fundamentals: Concepts, Systems & Applications*, BPB.
3. Mano, *Digital Logic and Computer Design*, Mano, Pearson.
4. Leach, Malvino, Saha, *Digital Principles and Applications*, TMH.
5. Goel, *Computer Fundamentals*, Pearson.

C2P: Digital electronics LAB Practical: 60 Lectures

C2T:	Course Outcome
1.	Extend the knowledge to implement the basic digital circuits.
2.	Experiment with the ICs to solve problems related to Digital logic circuits.
3.	Design and test the combinational circuits, and code conversion methods.
4.	Compare various synchronous and asynchronous sequential circuits.

1. Realization of different basic gates by NAND and NOR gates.
2. Realization of any Boolean function.
3. Implement a decoder, encoder, multiplexer, demultiplexer.
4. Implement half adder, full adder.
5. Implement half subtractor, full subtractor.
6. Implement a 2-bit magnitude comparator.
7. Implement S-R flip flop, D flip flop, J-K flip flop, Master-Slave J-K flip flop.
8. Implement counters.

C3T : Programming in JAVA Theory: 60 Lectures

	Course Outcome
1.	Interpret the principal of Object Oriented Programming OOP using programming syntaxes of JAVA programming language by analyzing the problems.
2.	Identify the requirements to the solution of complex engineering problems by proper analysis of classes with their relationships and interpretation of data/objects.
3.	Construct computer programs to implement the major OOP concepts related to Class & Object, Polymorphism, Inheritance, Interface, Exception Handling, Multi-processing (Thread), etc using JAVA coding ethics by making use of modern tools like Notepad++, Netbeans or Eclipse IDE.
4.	Develop Graphical User Interfaces using Applet, Swing, Layout manager, Jbutton class with action listeners, etc.
5.	Build small OOP based applications working individually or in a team with proper documentations following the professional OOP based engineering solution techniques.
6.	Determine the need for different OOP components from the implementation point of view to contribute to lifelong learning.

1.Introduction to Java (4 Lectures)

Java Architecture and Features, Understanding the semantic and syntax differences between C++

and Java, Compiling and Executing a Java Program, Variables, Constants, Keywords Data Types, Operators (Arithmetic, Logical and Bit wise) and Expressions, Comments, Doing Basic Program Output, Decision Making Constructs (conditional statements and loops) and Nesting, Java Methods (Defining, Scope, Passing and Returning Arguments, Type Conversion and Type and Checking, Built-in Java Class Methods),

2. **Arrays, Strings and I/O** **(8 Lectures)**

Creating & Using Arrays (One Dimension and Multi-dimensional), Referencing Arrays Dynamically, Java Strings: The Java String class, Creating & Using String Objects, Manipulating Strings, String Immutability & Equality, Passing Strings To & From Methods, String Buffer Classes. Simple I/O using System.out and the Scanner class, Byte and Character streams, Reading/Writing from console and files.

3. **Object-Oriented Programming Overview** **(4 Lectures)**

Principles of Object-Oriented Programming, Defining & Using Classes, Controlling Access to Class Members, Class Constructors, Method Overloading, Class Variables & Methods, Objects as parameters, final classes, Object class, Garbage Collection.

4. **Inheritance, Interfaces, Packages, Enumerations, Autoboxing and Metadata (14 lectures)**

Inheritance: (Single Level and Multilevel, Method Overriding, Dynamic Method Dispatch, Abstract Classes), Interfaces and Packages, Extending interfaces and packages, Package and Class Visibility, Using Standard Java Packages (util, lang, io, net), Wrapper Classes, Autoboxing/Unboxing, Enumerations and Metadata.

5. Exception Handling, Threading, Networking and Database Connectivity (15 Lectures)

Exception types, uncaught exceptions, throw, built-in exceptions, Creating your own exceptions; Multi-threading: The Thread class and Runnable interface, creating single and multiple threads, Thread prioritization, synchronization and communication, suspending/resuming threads. Using java.net package, Overview of TCP/IP and Datagram programming. Accessing and manipulating databases using JDBC.

6. Applets and Event Handling **(15 Lectures)**

Java Applets: Introduction to Applets, Writing Java Applets, Working with Graphics, Incorporating Images & Sounds. Event Handling Mechanisms, Listener Interfaces, Adapter and Inner Classes. The design and Implementation of GUIs using the AWT controls, Swing components of Java Foundation Classes such as labels, buttons, text fields, layout managers, menus, events and listeners; Graphic objects for drawing figures such as lines, rectangles, ovals, using different fonts. Overview of servlets.

Recommended Books

1. Ken Arnold, James Gosling, David Homes, "The Java Programming Language", 4th Edition, 2005.

2. Cay S. Horstmann, Gary Cornell, "Core Java Volume 1 Fundamentals, Pearson.

3. Cay S. Horstmann, Gary Cornell, "Core Java 2 Volume 2 - Advanced Features", Pearson.

4. James Gosling, Bill Joy, Guy L Steele Jr, Gilad Bracha, Alex Buckley, "The Java Language Specification, Java SE 8 Edition (Java Series)", Addison-Wesley, 2014.

5. Joshua Bloch, "Effective Java" 2nd Edition,Publisher: Addison-Wesley, 2008.

6. Bruce Eckel, "Thinking in Java", 3rd Edition, PHI, 2002.

7. E. Balagurusamy, "Programming with Java", 4th Edition, McGraw Hill.2009.

8. Paul Deitel, Harvey Deitel, "Java: How to Program", 10th Edition, Prentice Hall, 2011.

9. "Head First Java", Orielly Media Inc. 2nd Edition, 2005.

C3P: Programming in Java Lab **Practical: 60 Lectures**

	Course Outcome
1.	Interpret the principles of OOP using programming syntaxes of JAVA programming language by analyzing the problems.
2.	Identify the requirements to the solution of complex engineering problems by proper analysis of classes with their relationships and interpretation of data/objects.
3.	Construct computer programs to implement the major OOP concepts related to Class & Object, Polymorphism, Inheritance, Interface, Exception Handling, Multi-processing (Thread), etc using JAVA coding ethics by making use of modern tools like Notepad++, Netbeans or Eclipse IDE.
4.	Develop GraphicalUser Interfaces using Applet, Swing, Layout manager, Jbutton class with action listeners, etc.
5.	Build small OOP based applications working individually or in a team with proper documentations following the professional OOP based engineering solution techniques.
6.	Determine the need for different OOP components hands-on to produce huge distributed data driven software from the implementation point of view to contribute to lifelong learning.

1. To find the sum of any number of integers entered as command line arguments
2. To find the factorial of a given number
3. To learn use of single dimensional array by defining the array dynamically.
4. To learn use of.length in case of a two dimensional array
5. To convert a decimal to binary number
6. To check if a number is prime or not, by taking the number as input from the keyboard
7. To find the sum of any number of integers interactively, i.e., entering every number from the keyboard, whereas the total number of integers is given as a command line argument
8. Write a program that show working of different functions of String and StringBufferclasss like setCharAt(, setLength(), append(), insert(), concat()and equals().
9. Write a program to create a ―distance‖ class with methods where distance is computed in terms of feet and inches, how to create objects of a class and to see the use of this pointer
10. Modify the ―distance‖ class by creating constructor for assigning values (feet and inches) to the distance object. Create another object and assign second object as reference variable to another object reference variable. Further create a third object which is a clone of the first object.
11. Write a program to show that during function overloading, if no matching argument is found, then java will apply automatic type conversions (from lower to higher data type)
12. Write a program to show the difference between public and private access specifiers. The program should also show that primitive data types are passed by value and objects are passed by reference and to learn use of final keyword
13. Write a program to show the use of static functions and to pass variable length arguments in a function.
14. Write a program to demonstrate the concept of boxing and unboxing.
15. Create a multi-file program where in one file a string message is taken as input from the user and the function to display the message on the screen is given in another file (make use of Scanner package in this program).
16. Write a program to create a multilevel package and also creates a reusable class to generate Fibonacci series, where the function to generate Fibonacci series is given in a different file

belonging to the same package.

17. Write a program that creates illustrates different levels of protection in classes/subclasses belonging to same package or different packages
18. Write a program —*DivideByZero* that takes two numbers a and b as input, computes a/b, and invokes Arithmetic Exception to generate a message when the denominator is zero.
19. Write a program to show the use of nested try statements that emphasizes the sequence of checking for catch handler statements.
20. Write a program to create your own exception types to handle situation specific to your application (*Hint*: Define a subclass of Exception which itself is a subclass of Throwable).
21. Write a program to demonstrate priorities among multiple threads.
22. Write a program to demonstrate multithread communication by implementing synchronization among threads (Hint: you can implement a simple producer and consumer problem).
23. Write a program to create URL object, create a URLConnection using the openConnection() method and then use it examine the different components of the URLand content.
24. Write a program to implement a simple datagram client and server in which a message that is typed into the server window is sent to the client side where it is displayed.
25. Write a program that creates a Banner and then creates a thread to scrolls the message in the banner from left to right across the applet‘s window.
26. Write a program to get the URL/location of code (i.e. java code) and document(i.e. html file).
27. Write a program to demonstrate different mouse handling events like mouseClicked(), mouseEntered(), mouseExited(), mousePressed, mouseReleased() and mouseDragged().
28. Write a program to demonstrate different keyboard handling events.
29. Write a program to generate a window without an applet window using main() function.
30. Write a program to demonstrate the use of push buttons.

C4T: Computer Architecture and Organisation: Theory: 60 Lectures

	Course Outcome
1.	Demonstrate sufficient knowledge and understanding of data representation, and experiment with basic arithmetic operations.
2.	Analyze and model various functional units of CPU such as ALU, control unit and register file.
3.	Organize the memory hierarchy and design a memory of any type.
4.	Explain the instruction set architecture, instruction formats and instruction cycle.
5.	Outline various modes of I/O operations and summarize working principles of I/O interface circuits.
6.	Explain the pipelining technique and its related issues.

Unit 1: **Basic Computer Organization and Design** **(13 lectures)**

Computer registers, bus system, instruction set, timing and control, instruction cycle, memory reference, input-output and interrupt, Interconnection Structures, Bus Interconnection design of basic computer.

Unit 2: **Central Processing Unit** **(20 lectures)**

Register organization, arithmetic and logical micro-operations, stack organization, micro programmed control. Instruction formats, addressing modes, instruction codes, machine language, assembly language, input output programming, RISC, CISC architectures, pipelining and parallel architecture.

Unit 3: **Memory Organization**

(15 lectures)

Cache memory, Associative memory, mapping.

Unit 4: **Input-Output Organization** **(12 lectures)**

Input / Output: External Devices, I/O Modules, Programmed I/O, Interrupt-Driven I/O, Direct

Memory Access, I/O Channels.

Recommended Books:

1. M. Mano, *Computer System Architecture*, Pearson Education.
2. W. Stallings, *Computer Organization and Architecture Designing for Performance*, PHI.
3. Patterson, Hennessy, *Computer Organization and Design*, MK Publication.
4. M. Mano , *Digital Design*, Pearson Education.
5. Carl Hamacher, *Computer Organization*, TMH.
6. Pal Chaudhury, *Computer Organization and Design*, PHI.
7. Hayes, *Computer Architecture and Organization*, TMH.
8. Sajjan G Shiva, *Computer Organization Design and Architecture*, CRC Press.

C4P: PC Assembly and software Installation **Practical: 40 Lectures**

	Course Outcome
1.	Familiarization with different parts inside a computer system cabinet
2.	Identify different types of cables and their connections to different components
3.	Ability to identify different jumpers and their roles
4.	Learn to install software in Windows and Linux systems
5.	Learn to install virtual machine and installing iso image files

Hardware Assembly, motherboard, CPU, RAM slots, HDD, interconnection cables, jumpers in motherboard, SMPS, connecting I/O devices through USB and installation of device drivers, Application software installation in Windows and Linux, Installation of Windows and Ubuntu O.S., Oracle VM Virtual Box installation and install *iso* image of an O.S. in Virtual Box.

C5T: Data Structures **Theory: 60 Lectures**

	Course Outcome
1.	Define different operations on data structure such as insertion, deletion, merging using arrays
2.	Demonstrate implementation of stacks and queues: insertion, deletion of elements
3.	Construction and implementation of linked lists: inserting, deleting, and inverting a linked list. Analyze implementation of stacks & queues using linked lists,
4.	Design recursive algorithms for different some recursive and non-recursive problems.
5.	Understand the binary tree data structure, its memory representations
6.	Identify the unique binary tree from given any two traversals of a binary tree
7.	Ability to understand different searching algorithms and their applicability
8.	Ability to understand working principle of different sorting techniques
9.	construct a binary search tree (BST), insertion and deletion in BST. Understand the need of balanced search tree
10.	Construction of different hash functions

1. Arrays **(5 Lectures)**

Single and Multi-dimensional Arrays, Sparse Matrices (Array and Linked Representation)

2. **Stacks** **(5 Lectures)**
Implementing single / multiple stack/s in an Array; Prefix, Infix and Postfix expressions, Utility and conversion of these expressions from one to another; Applications of stack; Limitations of Array representation of stack

3. **Linked Lists** **(10 Lectures)**
Singly, Doubly and Circular Lists (Array and Linked representation); Normal and Circular representation of Stack in Lists; Self Organizing Lists; Skip Lists

4. **Queues** **(5 Lectures)**
Array and Linked representation of Queue, Dequeue, Priority Queues

5. **Recursion** **(5 lectures)**
Developing Recursive Definition of Simple Problems and their implementation; Advantages and Limitations of Recursion; Understanding what goes behind Recursion (Internal Stack Implementation)

6. **Trees** **(20 Lectures)**
Introduction to Tree as a data structure; Binary Trees (Insertion, Deletion , Recursive and Iterative Traversals on Binary Search Trees); Threaded Binary Trees (Insertion, Deletion, Traversals); Height-Balanced Trees (Various operations on AVL Trees).

7. **Searching and Sorting** **(5 Lectures)**
Linear Search, Binary Search, Comparison of Linear and Binary Search, Selection Sort, Insertion Sort, Insertion Sort, Shell Sort, Comparison of Sorting Techniques

8. **Hashing** **(5 Lectures)**
Introduction to Hashing, Deleting from Hash Table, Efficiency of Rehash Methods, Hash Table Reordering, Resolving collusion by Open Addressing, Coalesced Hashing, Separate Chaining, Dynamic and Extendible Hashing, Choosing a Hash Function, Perfect Hashing Function

Recommended Books:

1. Horowitz, Sahni, Mehta, *Fundamentals of Data Structures in C++*, University Press, 2011.
2. Mark Allen Weiss, *Data Structures and Algorithms Analysis in Java*, Pearson Education, 3rd Edition, 2013.
3. Tenenbaum, Augenstein, Langsam, *Data Structures Using C and C++*, 2nd Edition, PHI, 2009.
4. Cormen, Leiserson, Rivest, Stein, *Introduction to Algorithms*, PHI, 3rd Edition 2009
5. Kruse, *Data Structures and Program Design in C++*, Pearson, 1999.
6. Goodrich, Tamassia, *Data Structures and Algorithms Analysis in Java*, 4th Edition, Wiley, 2013.

C5P: Data Structures Lab **Practical: 60 Lectures**

	Course Outcome
1.	Ability to implement stack and queue with array data structure.
2.	Ability to design stack from queue and vice versa
3.	Implement multiple stacks in a single array
4.	Implement a linked list using self-referential structure and perform basic operations such as insertion, deletion, traversal, reversal etc.
5.	Implement a binary search tree
6.	Design and implement of different searching and sorting algorithms.

1. Write a program to search an element from a list. Give user the option to perform Linear or Binary search. Use Template functions.
2. WAP using templates to sort a list of elements. Give user the option to perform sorting using Insertion sort, Bubble sort or Selection sort.
3. Implement Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list and concatenate two linked lists (include a function and also overload operator +).
4. Implement Doubly Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.
5. Implement Circular Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.
6. Perform Stack operations using Linked List implementation.
7. Perform Stack operations using Array implementation. Use Templates.
8. Perform Queues operations using Circular Array implementation. Use Templates.
9. Create and perform different operations on Double-ended Queues using Linked List implementation.
10. WAP to scan a polynomial using linked list and add two polynomials.
11. WAP to calculate factorial and to compute the factors of a given no.
    (i) using recursion,
    (ii) using iteration
12. WAP to display Fibonacci series (i) using recursion, (ii) using iteration
13. WAP to calculate GCD of 2 number (i) with recursion (ii) without recursion
14. WAP to create a Binary Search Tree and include following operations in tree:
    (a) Insertion (Recursive and Iterative Implementation)
    (b) Deletion by copying
    (c) Deletion by Merging
    (d) Search a no. in BST
    (e) Display its preorder, postorder and inorder traversals Recursively
    (f) Display its preorder, postorder and inorder traversals Iteratively
    (g) Display its level-by-level traversals
    (h) Count the non-leaf nodes and leaf nodes
    (i) Display height of tree
    (j) Create a mirror image of tree
    (k) Check whether two BSTs are equal or not
15. WAP to convert the Sparse Matrix into non-zero form and vice-versa.
16. WAP to reverse the order of the elements in the stack using additional stack.
17. WAP to reverse the order of the elements in the stack using additional Queue.
18. WAP to implement Diagonal Matrix using one-dimensional array.
19. WAP to implement Lower Triangular Matrix using one-dimensional array.
20. WAP to implement Upper Triangular Matrix using one-dimensional array.
21. WAP to implement Symmetric Matrix using one-dimensional array.
22. WAP to create a Threaded Binary Tree as per inorder traversal, and implement operations like finding the successor / predecessor of an element, insert an element, inorder traversal.
23. WAP to implement various operations on AVL Tree.

C6T: Operating Systems

Theory: 60 Lectures

	Course Outcome
1.	Illustrate the resource-management by the Operating System (OS) and describe the basic principles used in the design of modern OS
2.	Apply various CPU scheduling algorithms for any given problem and outline the needs and applications of process synchronization
3.	Identify the issues in deadlock in terms of avoiding, preventing and recovering the same

4.	Design solutions using semaphore variables to solve different mutual exclusion problems
5.	Elaborate the different schemes used in memory management including paging and segmentation
6.	Analyze various file and disk management strategies
7.	Justify the issues in I/O management and security

1. Introduction (10 Lectures)

Basic OS functions, resource abstraction, types of operating systems–multiprogramming systems, batch systems, time sharing systems; operating systems for personal computers & workstations, process control & real time systems.

2. Operating System Organization (6 Lectures)

Processor and user modes, kernels, system calls and system programs.

3. Process Management (20 Lectures)

System view of the process and resources, process abstraction, process hierarchy, threads, threading issues, thread libraries; Process Scheduling, non-pre-emptive and pre-emptive scheduling algorithms; concurrent and processes, critical section, semaphores, methods for inter- process communication; deadlocks.

4. Memory Management (10 Lectures)

Physical and virtual address space; memory allocation strategies –fixed and variable partitions, paging, segmentation, virtual memory

5. File and I/O Management (10 Lectures)

Directory structure, file operations, file allocation methods, device management.

6. Protection and Security (4 Lectures)

Policy mechanism, Authentication, Internal access Authorization.

Recommended Books:

1. Silberschatz, Galvin, Gagne, Operating Systems Concepts, 8th Edition, John Wiley Publications 2008.
2. Tanenbaum, Modern Operating Systems, 3rd Edition, Pearson Education 2007.
3. G. Nutt, Operating Systems: A Modern Perspective, 2nd Edition Pearson Education 1997.
4. W. Stallings, Operating Systems, Internals & Design Principles , 5th Edition, PHI. 2008.
5. M. Milenkovic, Operating Systems- Concepts and design, Tata McGraw Hill 1992.

C6P: Operating Systems Lab **Practical: 60 Lectures**

	Course Outcome
1.	Execute the Linux commands to perform the basic operations related to process and system.
2.	Know different system calls in Linux and their usage
3.	Demonstrate the execution of the programs like creating new process, creating orphan process and zombie process based on child-parent relationship
4.	Apply the knowledge of semaphore and thread
5.	Implement different CPU scheduling algorithms
6.	Implement some basic memory allocation strategies

1. WRITE A C/C++ PROGRAM (WAP) (using *fork()* and/or *exec()* commands) where parent and child execute:
   a) same program, same code.
   b) same program, different code.
   c) before terminating, the parent waits for the child to finish its task.
2. WAP to report behaviour of Linux kernel including kernel version, CPU type and model.

(CPU information)

3. WAP to report behaviour of Linux kernel including information on configured memory, amount of free and used memory. (memory information)
4. WAP to print file details including owner access permissions, file access time, where file name is given as argument.
5. WAP to copy files using system calls.
6. WAP to implement FCFS scheduling algorithm.
7. WAP to implement Round Robin scheduling algorithm.
8. WAP to implement SJF scheduling algorithm.
9. WAP to implement non-preemptive priority based scheduling algorithm.
10. WAP to implement preemptive priority based scheduling algorithm.
11. WAP to implement SRJF scheduling algorithm.
12. WAP to calculate sum of n numbers using *thread* library.
13. WAP to implement first-fit, best-fit and worst-fit allocation strategies.

C7T: Computer Networks — Theory: 60 Lectures

	Course Outcome
1.	Explain data communication system, components and the purpose of layered architecture.
2.	Illustrate the functionalities of each layer of OSI and TCP/IP reference model including their associated protocols.
3.	Understand the IP addressing schemes, subnet mask and the address used for broadcast
4.	Understand the routing protocols
5.	Apply the different error detection and correction schemes with examples
6.	Comprehend the different commonly used application layer protocols

1. Introduction to Computer Networks (8 Lectures)

Network definition; network topologies; network classifications; network protocol; layered network architecture; overview of OSI reference model; overview of TCP/IP protocol suite.

2. Data Communication Fundamentals and Techniques (10 Lectures)

Analog and digital signal; data-rate limits; digital to digital line encoding schemes; pulse code modulation; parallel and serial transmission; digital to analog modulation-; multiplexing techniques- FDM, TDM; transmission media.

Networks Switching Techniques and Access mechanisms (10 Lectures)

Circuit switching; packet switching- connectionless datagram switching, connection-oriented virtual circuit switching; dial-up modems; digital subscriber line; cable TV for data transfer.

3. Data Link Layer Functions and Protocol (10 Lectures)

Error detection and error correction techniques; data-link control- framing and flow control; error recovery protocols- stop and wait ARQ, go-back-n ARQ; Point to Point Protocol on Internet.

4. Multiple Access Protocol and Networks (5 Lectures)

CSMA/CD protocols; Ethernet LANS; connecting LAN and back-bone networks- repeaters, hubs, switches, bridges, router and gateways;

5. Networks Layer Functions and Protocols (6 Lectures)

Routing; routing algorithms; network layer protocol of Internet- IP protocol, Internet control protocols.

6. Transport Layer Functions and Protocols (6 Lectures)

Transport services- error and flow control, Connection establishment and release- 3-way handshake;

7. Overview of Application layer protocol (5 Lectures)

Overview of DNS protocol; overview of WWW &HTTP protocol.

Recommended Books

1. Kurose, Ross, *Computer Networking*: A Top Down Approach, 7th Edition, Pearson Education.
2. Forouzan, *Data Communications and Networking*, 4th Edition, TMH, 2007.
3. Tanenbaum, *Computer Networks*, 4th Edition, PHI , 2002.
4. Peterson, *Computer Network: A Systems Approach*, Morgan Kauffman Publication.

C7P: Computer Networks Lab **Practical: 60 Lectures**

	Course Outcome
1.	Implement program to read an IP address and determine its class
2.	Understand different library functions used for creation of socket, establishment of connections between client and server using TCP and UDP
3.	Implement client-server communication using TCP
4.	Implement client-server communication using UDP
5.	Implement some error detection algorithms
6.	Simulate some routing protocols

1. Simulate Cyclic Redundancy Check (CRC) error detection algorithm for noisy channel.
2. Simulate and implement stop and wait protocol for noisy channel.
3. Simulate and implement distance vector routing algorithm
4. Simulate and implement Dijkstra algorithm for shortest path routing.
5. WAP for Creation of sockets
6. Write a program to communicate between TCP client & server
7. Write a program to communicate between UDP client & socket
8. Write a socket Program for Echo/Ping/Talk commands

SEC-1: To select from SEC1 Electives

C8T: Design and Analysis of Algorithms **Theory: 60 Lectures**

	Course Outcome
1.	Understand asymptotic notations, determine time complexity of different algorithms from their recurrence relation.
2.	Comprehend different algorithm design paradigms
3.	Analyze use of divide-and-conquer strategy in merge sort, quick sort
4.	Appreciate the use of greedy technique to obtain optimum global solution for some problems
5.	Apply backtracking method to solve n-queens problems
6.	Understand the applicability of dynamic programming for some problems
7.	Comprehend the graph data structure, its traversal, and solution of minimum spanning tree

1. Introduction (10 Lectures)

Basic Design and Analysis techniques of Algorithms, Correctness of Algorithm. Time Complexity, Master's Theorem in solving recurrence relation.

2. Algorithm Design Techniques (15 Lectures)

Iterative techniques, Divide and Conquer, Dynamic Programming, Greedy Algorithms.

3. Sorting and Searching Techniques (20 Lectures)

Elementary sorting techniques–Bubble Sort, Insertion Sort, Merge Sort, Advanced Sorting

techniques - Heap Sort, Quick Sort, Sorting in Linear Time - Bucket Sort, Radix Sort and Count Sort, Searching Techniques, Medians & Order Statistics, complexity analysis;

4. **Red-Black Tree**. (5 Lectures)

5. **Graph Algorithms:** Breadth First Search, Depth First Search and its Applications, Minimum Spanning Trees. (5 Lectures)

6. String Processing: (5 Lectures)
String Matching, KMP Technique

Recommended Books:

1. Cormen, Leiserson, Rivest, Stein, *Introduction to Algorithms*, PHI, 3rd Edition 2009.

2. Horowitz, Sahni, Rajasekaran, *Computer Algorithm*, 2nd Edition, Orient Blackswan Publication.

3. Sarabasse, Gelder, *Computer Algorithm – Introduction to Design and Analysis*, 3rd Edition, Pearson Education.

4. Kleinberg, Tardos, *Algorithm Design*, Pearson Education.

5. Basu, *Design Methods and Analysis of Algorithms*, PHI.

C8P: Design and Analysis of Algorithms Lab **Practical: 60 Lectures**

	Course Outcome
1.	Understand divide and conquer approach and to implement binary search, merge sort, quick sort
2.	Analysis of working principle of heap sort and radix sort by implementing them
3.	Implement graph traversal algorithms and analyse its output
4.	Apply backtracking method to solve n-queens problems.
5.	Design and Develop program to solve minimum spanning tree problem

1. i. Implement Insertion Sort (The program should report the number of comparisons)
   ii. Implement Merge Sort(The program should report the number of comparisons)
2. Implement Heap Sort(The program should report the number of comparisons)
3. Implement Randomized Quick sort (The program should report the number of comparisons)
4. Implement Radix Sort
5. Create a Red-Black Tree and perform following operations on it:
   i. Insert a node
   ii. Delete a node
   iii. Search for a number & also report the color of the node containing this number.
6. Write a program to determine the LCS of two given sequences
7. Implement Breadth-First Search in a graph
8. Implement Depth-First Search in a graph
9. Write a program to determine the minimum spanning tree of a graph

For the algorithms at S.No 1 to 3 test run the algorithm on 100 different inputs of sizes varying from 30 to 1000. Count the number of comparisons and draw the graph. Compare it with a graph of nlogn.

C9T: Software Engineering **Theory: 60 lectures**

	Course Outcome

1.	Explain the principles of software engineering in the context of social, ethical, legal, economic and environmental concerns by building applicable solutions.
2.	Identify and classify the customer requirements to the solution of complex engineering problems by proper analysis and interpretation of data and processes.
3.	Estimate software matrices like size, effort and cost, software reliability and quality, etc and apply project management techniques to maximize the productivity.
4.	Design various components of software using DFD, ERD, Modularization, Use-case diagram, Class diagram, Sequence diagram, etc. following the professional software design guidelines.
5.	Develop and Test software products following standard coding and testing guidelines.
6.	Asses the utility of various components of software development process and to combine them to produce different types of software to adapt in the software industries in future.

Introduction (12 Lectures)

The Evolving Role of Software, Software Characteristics, Changing Nature of Software, Software Engineering as a Layered Technology, Software Process Framework, Framework and Umbrella Activities, Process, Software Development Life Cycle. Waterfall model of SDLC.

Requirement Analysis (12 Lectures)

Software Requirement Analysis, Initiating Requirement Engineering Process, Requirement Analysis and Modeling Techniques, Flow Oriented Modeling, Need for SRS, Characteristics and Components of SRS.

Software Project Management (8 Lectures)

Estimation in Project Planning Process, Project cost estimation metrics, COCOMO model, Project Scheduling. Project risk.

Quality Management (8 Lectures)

Quality Concepts, Software Quality Assurance, Software Reviews, Metrics for Process and Projects.

Design Engineering (10 Lectures)

Design Concepts, Architectural Design Elements, Software Architecture, Data Design at the Architectural Level and Component Level, Mapping of Data Flow into Software Architecture, Modeling Component Level Design.

Testing Strategies & Tactics (12 Lectures)

Software Testing Fundamentals, Strategic Approach to Software Testing, Test Strategies for Conventional Software, Validation Testing, System testing, Black-Box Testing, White-Box Testing and their type, Basis Path Testing.

Recommended Books:

1 .R.S. Pressman, Software Engineering: A Practitioner‘s Approach (7th Edition), McGraw-Hill, 2009.
2 .P. Jalote, An Integrated Approach to Software Engineering (2ndEdition), Narosa Publishing House, 2003.

3. I. Sommerville, Software Engineering (8th edition), Addison Wesley, 2006.

4. D. Bell, Software Engineering for Students (4th Edition), Addison-Wesley, 2005.

5. R. Mall, Fundamentals of Software Engineering (2nd Edition), Prentice-Hall of India, 2004.

C9P: Software Engineering Lab **Practical: 60 Lecture**

	Course Outcome

1.	Identify and classify the customer requirements for the solution of complex engineering problems by proper analysis and interpretation of data and processes supported by standard documentation.
2.	Analyzethe software processes by mapping requirements in to Use case diagrams/ Data Flow Diagrams and Entity Relationship Diagrams for given case studies
3.	Experiment with modern tools like Rational rose, Smartdraw, Erdraw, etc. to design dynamic behaviour of software with modular programming, class diagrams, sequence diagrams, etc.
4.	Estimate software metric like size, effort and cost , software reliability and quality, etc and plan development schedule using PERT and GNATT charts
5.	Design the Test cases and the Test suits for the given case studies using Black box and White box techniques
6.	Determine and evaluate the various components of software development process practically and to combine them to produce different types of software to adapt in the software industries in future

S. No	**Practical Title**
1.	□ Problem Statement, □ Process Model
2.	Requirement Analysis: Creating a Data Flow Data Dictionary, Use Cases Project
	Management:
3	□ Computing FP □ Effort • Schedule, Risk Table, Timeline chart
4.	Design Engineering: □ Architectural Design • Data Design, Component Level Design
5.	Testing: □ Basis Path Testing

Sample Projects:

1. Criminal Record Management: Implement a criminal record management system for jailers, police officers and CBI officers

2. DTC Route Information: Online information about the bus routes and their frequency and fares

3. Car Pooling: To maintain a web based intranet application that enables the corporate employees within an organization to avail the facility of carpooling effectively.

4. Patient Appointment and Prescription Management System

5. Organized Retail Shopping Management Software

6. Online Hotel Reservation Service System

7. Examination and Result computation system

8. Automatic Internal Assessment System

9. Parking Allocation System

10. Wholesale Management System

C10T : Database Management System Theory: 60 Lectures

	Course Outcome
1.	Define and understand the fundamentals of Data base management System and traditional file system.
2.	Understand and explain the concepts of relational database management system.
3.	Make use of the tools to implement Entity Relationship diagrams.
4.	Utilize and take part in the normalization of the real world database to remove redundancies and able to apply the conversion of one Normal Form to Higher Normal Form.
5.	Elaborate the importance and rule on database management system concepts to minimize conflict in concurrent transactions.
6.	Discuss the importance of Database management system for storage of data in various formats and able to judge the environmental, societal and market issues specific to software development.

1. Introduction (6 Lectures)

Characteristics of database approach, data models, database system architecture and data independence.

2. Entity Relationship(ER) Modeling (8 Lectures)

Entity types, relationships, constraints.

3. Relation data model (20 Lectures)

Relational model concepts, relational constraints, relational algebra, SQLqueries

4. Database design (15 Lectures)

Mapping ER/EER model to relational database, functional dependencies, Lossless decomposition, Normalforms(upto BCNF).

5. Transaction Processing (3 Lectures)

ACID properties, concurrency control

6. File Structure and Indexing (8 Lectures)

Operations on files, File of Unordered and ordered records, overview of File organizations, Indexing structures for files(Primary index, secondary index, clustering index), Multilevel

indexing using B and B **+trees.**

Books Recommended:

1. Elmasri, Navathe, *Fundamentals of Database Systems*, 6th Edition, Pearson Education, 2010.

2. Silberschatz, Korth, Sudarshan, *Database System Concepts*, 6th Edition, TMH.

3. Ramakrishanan, Gehrke, *Database Management Systems*, 3rd Edition, TMH. 2002.

C10P : Database Management System Lab Practical: 60 Lectures

	Course Outcome
1.	Outline the underlying concepts of database technologies.
2.	Define and demonstrate DBMS architecture, schema, instance, DDL, DML.
3.	Experiment with SQL to construct and apply to execute database query using SQL DML/DDL commands.
4.	List and test the integrity constraints on a database using a RDBMS and discover relationships.
5.	Explain Programming in PL/SQL with stored procedures, cursors, packages.
6.	Compose and improve/solve the need of DBMS tool for the use of modern software development.

Structured Query Language

1. Creating Database
Creating a Table
Specifying Relational Data Types
Specifying Constraints
Creating Indexes

2. Table and Record Handling

INSERT statement
Using SELECT and INSERT together
DELETE, UPDATE, TRUNCATE statements
DROP, ALTER statements

3. Retrieving Data from Database

The SELECT statement
Using the WHERE clause
Using Logical Operators in the WHERE clause
Using IN, BETWEEN, LIKE , ORDER BY, GROUP BY and HAVING Clause
Using Aggregate Functions
Combining Tables Using JOINS
Subqueries

4. Database Management

Creating Views
Creating Column Aliases
Creating Database Users
Using GRANT and REVOKE

C11T: Web Technologies Theory: 60 Lectures

	Course Outcome
1.	To examine and evaluate the need for secured web application development with client-side, server-side scripting languages.
2.	To construct web programs using the web languages--HTML, XML, JavaScript, Applet, Perl, etc.
3.	To design and develop small interactive websites using modern tools following the professional web based engineering solutions, ethics and management techniques.
4.	To determine and combine the advanced technologies like network security, multimedia applications, search engine, web crawler, etc with the websites.

Unit 1: Introduction to WWW- Protocols and programs, secure connections, application and development tools, the web browser, What is server, choices, setting up UNIX and Linux web servers, Logging users, dynamic IP Web Design: Web site design principles, planning the site and navigation, (8 Lectures)

Unit 2: Introduction to HTML- The development process, Html tags and simple HTML forms, web site structure Introduction to XHTML: XML, Move to XHTML, Meta tags, Character entities, frames and frame sets, inside browser. HTML Tags: Table, list, link etc (12 Lectures)

Unit 3: Cascading Style sheets- Need for CSS, introduction to CSS, basic syntax and structure, using CSS, background images, colours and properties, manipulating texts, using fonts, borders and boxes, margins, padding lists, positioning using CSS. (8 Lectures)

Unit 4: Javascript- Client side scripting, What is Javascript, How to develop Javascript, simple Javascript, variables, functions, conditions, loops and repetition. JavaScript alert, prompt and confirm. Objects in JavaScript, Access/Manipulate web browser elements using DOM Structure, forms and validations, JavaScript events, Basics of jQuery, jQuery syntaxes, jQuery selectors, events, effects, Access / Manipulate web browser elements using jQuery. (10 Lectures)

Unit 5: Introduction to PHP and its syntax, combining PHP and HTML, understanding PHP code blocks like Arrays, Strings, Functions, looping and branching, file handling, processing forms on server side, cookies and sessions. Introduction to PHP MyAdmin, connection to MySQL server from PHP, execution of MySQL queries from PHP, receiving data from database server and processing it on webserver using PHP. (12 Lectures)

Recommended Books:

1. Ivan Bayross, *Web Enabled Commercial Application Development Using HTML, DHTML, Javascript, Perl CGI*, BPB Publications, 2009.
2. Hans Bergsten, *Java Server Pages*, O'Reilly , Third Edition, 2003.
3. Knuckles, *Web Applications: Concepts and Real World Design*, Wiley-India.
4. P.J. Deitel & H.M. Deitel, *Internet and World Wide Web How to program*, Pearson.
5. Julie C. Meloni, *HTML, CSS and JavaScript All in One*, Sams Teach Yourself, 2nd Edition.
6. Paul Wellens, *Practical Web Development*, PACKT Publication.

C11P : Web Technologies Lab **Practical: 60 Lectures**

	Course Outcome
1.	Learn finer details of website development process
2.	Design simple as well as moderate websites
3.	Apply knowledge of HTML, CSS, Javascript and PHP in designing dynamic website

Practical on HTML

1. Design web pages for your college containing a description of the courses, departments, faculties, library etc, use href, list tags.
2. Create your class time table using table tag.
3. Create user Student feedback form (use textbox, text area , checkbox, radio, button, select box etc.)
4. Create a web page using frame. Divide the page into two parts with Navigation links on left hand side of page (width=20%) and content page on right hand side of page (width = 80%). On clicking the navigation Links corresponding content must be shown on the right hand side.
5. Write html code to develop a webpage having two frames that divide the webpage into two equal rows and then divide the row into equal columns fill each frame with a different background color.
6. Create your resume using HTML tags also experiment with colors, text , link , size and also other tags you studied.

Practical on CSS

7. Design a web page of your home town with an attractive background color, text, color, an Image, font etc. (use internal CSS).

8. Use Inline CSS to format your resume that you created.
9. Use External CSS to format your class timetable as you created.
10. Use External, Internal, and Inline CSS to format college web page that you created.

Practical on JavaScript

11. Develop a JavaScript to display today's date.
12. Develop simple calculator for addition, subtraction, multiplication and division operation using JavaScript
13. Create HTML Page with JavaScript which takes Integer number as input and tells whether the number is ODD or EVEN.
14. Create HTML Page that contains form with fields Name, Email, Mobile No, Gender , Favourite Colour and a button now write a JavaScript code to combine and display the information in textbox when the button is clicked.

Practicals on PHP

15. Server side scripts and validation arrays for a simple log-in page of website.

SEC-2: To select from list of SEC2 electives

C12T: Introduction to Data Science **Theory: 60 Lectures**

	Course Outcome
1.	Revisit the R programming
2.	Appreciate the need of data science in today's world and it's complex nature
3.	Ability to clean the data from the raw data
4.	Plot the datasets
5.	Use the dplyr, tidyr and other packages

Data Scientist's Tool Box: Turning data into actionable knowledge, introduction to the tools that will be used in building data analysis software: version control, markdown, git, GitHub. (8 Lectures)

R Programming Basics: Overview of R, R data types and objects, reading and writing data, Control structures, functions, scoping rules, dates and times, Loop functions, debugging tools, Simulation, code profiling. (15 Lectures)

Getting and Cleaning Data: Obtaining data from the web. Use dplyr package to select specific rows/columns of a data set, and to join two datasets. Basics of data cleaning and making data using tidyr package. (12 Lectures)

Exploratory Data Analysis: Essential exploratory techniques for summarizing data, applied before formal modeling commences, eliminating or sharpening potential hypotheses about the world that can be addressed by the data, common multivariate statistical techniques used to visualize high-dimensional data. (12 Lectures)

Reproducible Research: Concepts and tools behind reporting modern data analyses in a reproducible manner, To write a document using R markdown, integrate live R code into a literate statistical program, compile R markdown documents using knitr and related tools, and organize a data analysis so that it is reproducible and accessible to others. (13 Lectures)

Recommended books

1. Motwani, Data Analytics using R, Wiley Asia
2. Seema Acharya, Data Analytics Using R, TMH
3. Rajkamal, Saxena, Big Data and Analytics, Wiley Asia

C12P : Introduction to Data Science Lab Practical: 60 Lectures

	Course Outcome

1.	Ability to write R programs for solving data processing computing problems
2.	Apply the dplyr and tidyr packages to extract desired data from dataframe or dataset
3.	Plot one or more functions in a window with appropriate labels, legneds

List of Practical:

1. Write a program that prints '*Hello World*' to the screen.
2. Write a program that asks the user for a number n and prints the sum of the numbers 1to n
3. Write a program that prints a multiplication table for numbers up to 12.
4. Write a function that returns the largest element in a list.
5. Write a function that computes the running total of a list.
6. Write a function that tests whether a string is a palindrome.
7. Implement linear search.
8. Implement binary search.
9. Implement matrices addition, subtraction and Multiplication
10. Fifteen students were enrolled in a course. There ages were: 20 20 20 20 20 21 21 21 22 22 22 22 23 23 23
    i. Find the median age of all students under 22 years
    ii. Find the median age of all students
    iii. Find the mean age of all students
    iv. Find the modal age for all students
    v. Two more students enter the class. The age of both students is 23. What is now mean, mode and median?
11. Use dplyr package to select some rows and columns of a dataset.
12. Use dplyr package to join two tables.
13. Display system date and time and time zone.
14. Plot a function
15. Plot multiple functions on same window with appropriate labels.

DSE-1: To select from list of DSE2 Electives.

DSE-2: To select from list of DSE2 Electives.

C13T : Cyber Security and Cyber Laws Theory: 60 Lectures

	Course Outcome
1.	To understands the conceptual and technical foundation cyber security.
2.	To exhibit knowledge to secure corrupted systems, protect personal data, and secure computer networks in an Organization
3.	To identify and analyze statutory, regulatory, constitutional, and organizational laws that affects the information technology professional.
4.	To apply case law and common law to current legal dilemmas in the technology field.
5.	To apply diverse viewpoints to ethical dilemmas in the information technology field and recommend appropriate actions,
6.	To understand principles of web security and to guarantee a secure network by monitoring and analyzing the nature of attacks through cyber/computer forensics software/tools.

1. Introduction (5 Lectures)

Security, Attacks, Computer Criminals, Security Services, Security Mechanisms.

2. Cryptography (20 Lectures)

Substitution ciphers, Transpositions Cipher, Confusion, diffusion, Symmetric, Asymmetric Encryption. DES Modes of DES, Uses of Encryption, Hash function, key exchange, Digital Signatures, Digital Certificates.

3. Program Security (5 Lectures)

Secure programs, Non malicious Program errors, Malicious codes virus, Trap doors, Salami attacks, Covert channels, Control against program

4. Threats. (6 Lectures)

Protection in OS: Memory and Address Protection, Access control, File Protection, User Authentication.

5. Database Security (6 Lectures)

Requirements, Reliability, Integrity, Sensitive data, Inference, Multilevel Security.

6. Security in Networks (8 Lectures)

Threats in Networks, Security Controls, firewalls, Intrusion detection systems, Secure e-mails

7. Administrating Security (10 Lectures)

Security Planning, Risk Analysis, Organisational Security Policy, Physical Security. Ethical issues in Security: Protecting Programs and data. Information and law.

Recommended Books:

1. C. P. Pfleeger, S. L. Pfleeger; *Security in Computing*, PHI
2. W. Stallings, *Network Security Essentials: Applications and Standards*, Pearson
3. Forouzan, Mukhopadhyay, *Cryptography And Network Security*, TMH
4. Stallings, *Cryptography and Network Security: Principles and Practice*, Pearson

C13P : Cyber Security and Cyber Laws Lab Practical: 60 Lectures

	Course Outcome
1.	Design a simple cipher offering confidentiality
2.	Design a simple cipher that provides authentication and/or integrity and/or non-repudiation
3.	Demonstrate use of different network security tools
4.	Protect word document, worksheet etc from unauthorized users

1. Demonstrate the use of Network tools: ping, ipconfig, ifconfig, tracert, arp, netstat, whois
2. Use of Password cracking tools : John the Ripper, Ophcrack. Verify the strength of passwords using these tools.
3. Perform encryption and decryption of Caesar cipher. Write a script for performing these operations.
4. Perform encryption and decryption of a Rail fence cipher. Write a script for performing these operations.
5. Use nmap/zenmap to analyse a remote machine.
6. Use Burp proxy to capture and modify the message.
7. Demonstrate sending of a protected word document.
8. Demonstrate sending of a digitally signed document.
9. Demonstrate sending of a protected worksheet.

10. Demonstrate use of steganography tools.
11. Demonstrate use of gpg utility for signing and encrypting purposes.

C14T: Computer Graphics Theory: 60 Lectures

	Course Outcome
1.	Outline computer graphics system, display devices and various application areas of graphics.
2.	Develop scan conversion algorithms for line, circle and ellipse with examples.
3.	Demonstrate and illustrate 2D and 3D transformation operations such as translation, rotation, scaling, etc.
4.	Analyze and model any kind of 3D objects using viewing, clipping and projection techniques.
5.	Apply various curve and surface representation methods such as BSpline, Bezier, etc.
6.	Demonstrate and discuss various hidden surface removal algorithms, and lighting and shading models.

1. Introduction (5 Lectures)
Basic elements of Computer graphics, Applications of Computer Graphics.

2. Graphics Hardware (8 Lectures)

Architecture of Raster and Random scan display devices, input/output devices.

3. Fundamental Techniques in Graphics (22 Lectures)

Raster scan line, circle and ellipse drawing, thick primitives, Polygon filling, line and polygon clipping algorithms, 2D and 3D Geometric Transformations, 2D and 3D Viewing Transformations (Projections- Parallel and Perspective), Vanishing points.

4. Geometric Modeling (10 Lectures)
Representing curves & Surfaces.

5. Visible Surface determination (8 Lectures)

Hidden surface elimination.

6. Surface rendering (7 Lectures)

Illumination and shading models. Basic color models and Computer Animation.

Books Recommended:

1. J.D. Foley, A.Van Dan, Feiner, Hughes Computer Graphics Principles & Practice, Addison Wesley
2. D. Hearn, Baker: Computer Graphics, PHI
3. D.F. Rogers Procedural Elements for Computer Graphics, McGraw Hill 1997.
4. D.F. Rogers, Adams Mathematical Elements for Computer Graphics, TMH
5. Xiang, Plastock, Schaum's Outline Computer Graphics, TMH

C14P: Computer Graphics Lab **Practical: 60 Lectures**

	Course Outcome
1.	Implement drawing lines, circles, certain curves
2.	Transform a given object by rotation, scaling
3.	Clip certain portion of objects
4.	Design a drawing by using simple objects and color different components

1. Write a program to implement Bresenham‘s line drawing algorithm.

2. Write a program to implement mid-point circle drawing algorithm.

3. Write a program to clip a line using Cohen and Sutherland line clipping algorithm.

4. Write a program to clip a polygon using Sutherland Hodgeman algorithm.
5. Write a program to apply various 2D transformations on a 2D object (use homogenous coordinates).

6. Write a program to apply various 3D transformations on a 3D object and then apply parallel and perspective projection on it.

7. Write a program to draw Hermite/Bezier curve.

DSE-3: To select from list of DSE3 Electives.

DSE-4: To select from list of DSE4 Electives.

Discipline Specific Elective Papers: (4 Papers to be selected) – DSE 1, DSE 2, DSE 3, DSE 4:

DSE1 Elective List

1. Artificial Intelligence (4 Credit) + Lab (2 Credit)
2. Microprocessor (4 Credit) + Lab (2 Credit)
3. Graph Theory (4 Credit) + Lab (2 Credit)

DSE2 Elective List

4. Numerical Methods (4 Credit) + Lab (2 Credit)
5. Computational Biology (4 Credit) + Lab (2 Credit)
6. Finite Automata (6+0 Credit)

DSE3 Elective List

7. Seminar (0+6 Credit)
8. Digital Image Processing (4 Credit) + Lab (2)
9. Machine Learning (4 Credit) + Lab (2)

DSE4 Elective List

10. Systems Programming (4 Credit) + Lab (2 Credit)

11. Cloud Computing (4 Credit) + Lab (2)

12. Project Work / Dissertation (0+6 Credit)

DSE1T : Artificial Intelligence **Theory: 60 Lectures**

	Course Outcome
1.	Explain the modern tools of Artificial Intelligence (AI) and knowledge representation models.
2.	Determine the domain-specific problems solving methods using AI based algorithms and techniques.
3.	Evaluate knowledge based representation and learning methods for broad areas of state-of-the-art technological growth.
4.	Develop and access some current applications of AI in the fields of Expert Systems, Robotics, Machine Learning and others.
5.	Design the efficient solutions using different AI based schemes, and relate with technological advancement for future learning.

Course Contents:

Unit-1. Introduction (8 Lectures)

Introduction to Artificial Intelligence, Background and Applications, Turing Test and Rational Agent approaches to AI, Introduction to Intelligent Agents, their structure, behavior and environment.

Unit-2. Problem Solving and Searching Techniques: (24 Lectures)

Problem Characteristics, Production Systems, Control Strategies, Breadth First Search, Depth First Search, Hill climbing and its Variations, Heuristics Search Techniques: Best First Search, A* algorithm, Constraint Satisfaction Problem, Means-End Analysis, Introduction to Game Playing, Min-Max and Alpha-Beta pruning algorithms.

Unit-3. Knowledge Representation: (12 Lectures)

Introduction to First Order Predicate Logic, Programming in Logic (PROLOG)

Unit-4. Dealing with Uncertainty and Inconsistencies: (16 Lectures)

Truth Maintenance System, Default Reasoning, Probabilistic Reasoning, Bayesian Probabilistic Inference, Possible World Representations.

Reference Books:

1. Russell & Norvig, Artificial Intelligence -A Modern Approach, Prentice Hall, 3rd edition, 2009.

2. Rich & Knight, Artificial Intelligence – Tata McGraw Hill, 2nd edition, 1991.

3. Ela Kumar, Artificial Intelligence, Wiley.

DSE1P Artificial Intelligence Lab **Practical: 60 Lectures**

	Course Outcome
1.	Learn Prolog programming language
2.	Apply knowledge of Prolog to solve some common computation problems
3.	Apply knowledge of Prolog to accomplish some list operations

List of Practical:

1. Write a prolog program to calculate the sum of two numbers.
2. Write a prolog program to find the maximum of two numbers.
3. Write a prolog program to calculate the factorial of a given number.
4. Write a prolog program to calculate the nth Fibonacci number.
5. Write a prolog program, insert_nth(item, n, into_list, result) that asserts that result is the list into_list with item inserted as the n'th element into every list at all levels.
6. Write a Prolog program to remove the Nth item from a list.
7. Write a Prolog program, remove-nth(Before, After) that asserts the After list is the Before list with the removal of every n'th item from every list at all levels.
8. Write a Prolog program to implement append for two lists.
9. Write a Prolog program to implement palindrome(List).
10. Write a Prolog program to implement max(X,Y,Max) so that Max is the greater of two numbers X and Y.
11. Write a Prolog program to implement reverse(List,ReversedList) that reverses lists.

DSE-1T: Microprocessor **Theory: 60 Lectures**

	Course Outcome
1.	Demonstrate software architectures of 8085 microprocessor
2.	Explain and illustrate the instruction set of 8085 microprocessor and how do they affect flags
3.	Understand the time and space requirement by 8085 instructions
4.	Design 8085 assembly level programs (ALP) to solve given tasks
5.	Mask or unmask interrupts by ALP
6.	Design circuit diagram for interfacing I/O devices through 8255

Unit1: 8085 Pin Diagram and Architecture: (8 Lectures)
8085 Microprocessor pin diagram and working of each pin, 8085 software architecture, Flag register.

Unit 2: 8085 instruction set: (25 Lectures)
Data transfer instructions, arithmetic instructions, logic instructions, rotate instructions, branching instructions, stack related instructions, subroutine call and return. 8085 ALPs.

Unit 3: 8085 and memory interfacing: (9 Lectures)
Interfacing circuit, linear decoding, partial decoding, fold back memory space, memory map, memory-mapped I/O, I/O-mapped I/O.

Unit 4: 8085 interrupts: (8 Lectures)
Vectored interrupt, non-vectored interrupt, Obtaining interrupt service routine addresses, interrupt related instructions, 8085 ALPs to enable/disable and mask/unmask interrupts.

Unit 5: 8255 PPI: (10 Lectures)
Ports, Control word, I/O mode, BSR mode, Interfacing I/O devices in memory-mapped I/O or I/O-mapped I/O through 8255.

Reference Books:

1. R. Gaonkar, 8085 Microprocessor, Penram International
2. Mandal, 8085 Microprocessor, TMH
3. Senthil Kumar, M Saravanan & S Jeevananthan, Microprocessors and Microcontrollers, OUP

DSE-1P: Microprocessor Practical **Practical: 60 Lectures**

	Course Outcome
1.	Write 8085 ALP to initialize 8085 registers, memory as per requirement
2.	Write 8085 ALP to set/reset flags as per requirement
3.	Use flags to implement loops using conditional jump instructions
4.	Write ALP to accomplish a given task
5.	Design a delay subroutine as per requirement

1. Familiarization with 8085 register level architecture and trainer kit and / or simulator.
2. ALP for initializing programmable registers.
3. ALP for initializing consecutive memory locations
4. ALP to do 8-bit and 16-bit addition, subtraction
5. ALP for copying a block of memory where source and destination are disjoint or overlapped
6. ALP for addition of BCD numbers,
7. ALP for binary to ASCII conversion
8. ALP to use a subroutine to find minimum of two numbers and use it to find minimum from a group of consecutive memory locations.
9. ALP to implement searching
10. ALP to implement sorting.

DSE1T Graph Theory **Theory: 60 Lectures**

	Course Outcome
1.	Familiar with different basic terminologies of graph
2.	Identify the paths, walks, circuits in a given graph
3.	Apply shortest path algorithms to find shortest path between a pair of vertices
4.	Determine minimum spanning tree from a given graph
5.	Determine chromatic number of a given graph using graph coloring
6.	Recognize a planar/non-planar graph
7.	Understand network flow, matching

Unit 1. Introduction: Graphs – Introduction, graph representation, Isomorphism, Sub graphs, Walks, Paths, Circuits, Euler graphs, Hamiltonian paths, Trees – binary tree, properties of trees, distance and centers in tree. (12 Lectures)

Unit 2. Graph Traversal: BFS, DFS. (6 Lectures)

Unit 3. Shortest Path and Spanning Tree: Single Source and All-pair shortest path, Shortest path, Spanning trees- Fundamental circuits, Spanning trees in a weighted graph. (12 Lectures)

Unit 4. Connectivity & Planarity: Cut sets – Properties of cut set, Fundamental circuits and cut sets, Connectivity and separability – Network flows. Planar graphs – Different representation of a planar graph.

(14 Lectures)

Unit 5. Graph Coloring, Matching: Graph Coloring: Four color problem , Applications of Graph Coloring. Chromatic Numbers, Chromatic polynomial and its determination. Independence and Clique Numbers. Definitions and Examples of Perfect Matching, Maximal and Maximum Matching, Hall's Marriage Theorem. (16 Lectures)

Reference Books:

1. Narasingh Deo, *Graph Theory*, PHI.

2. Douglas B. West, *Introduction to Graph Theory*, Prentice Hall India Ltd.

3. Robin J. Wilson, *Introduction to Graph Theory*, Longman Group Ltd.

DSE1P: Graph Theory Practical Practical: 60 Lectures

	Course Outcome
1.	Implement a graph using adjacency matrix and adjacency list
2.	Implement BFS, DFS graph traversal algorithms
3.	Implement single-source shortest path algorithms
4.	Find minimum spanning tree from a given graph by implementing Prim's or Kruskal's algorithm

1. Program to represent graph using adjacency matrix
2. Program to implement graph using adjacency list
3. Programs to implement BFS, DFS
4. Program to find single-source shortest path
5. Program to find all pair shortest path
6. Programs to obtain minimum spanning tree

DSE-2T: Numerical Methods **Theory: 60 Lectures**

	Course Outcome
1.	Recalling the basic mathematical tools such as, derivative, real integration, solution of equations, existence of solution of system of linear equations and differential equation.
2.	Describe the concept of error, operators and interpolation. Numerical approach of solving missing term, finding of polynomials, integrated value, solution of algebraic equations, system of linear equations and differential equation
3.	Use interpolation, integration for data analysis. Apply different numerical techniques to solve algebraic equations, system of linear equations in iterative way.
4.	Analyze different real time problems and categorize them during the process of solving, by numerical technique mentioned.

Unit 1. **Introduction**: Introduction to various kinds of errors. (3 Lectures)
Unit 2. **Interpolation**: Newton forward & backward, Lagrange. (12 Lectures)
Unit 3. **Numerical Integration**: Trapezoidal Rule, Simpson's 1 /3 Rule, Simpson's 3 /8 Rule. (12 Lectures)
Unit 4. **Numerical solution of a system of linear equation**: Gauss elimination, Gasuss Jacobi, Gauss Seidel. (10 Lectures)
Unit 5. **Solutions of Algebraic Equation**: Bisection, Regula-falsi, Newton Raphson. (9 Lectures)
Unit 6. **Solutions of Ordinary Differential Equation**: Taylor Series, Euler's method, Runge Kutta. (14 Lectures)

Recommended Books:

1. Dutta, Jana, Introductory Numerical Analysis, Shreedhar Prakashani
2. S. A. Mollah, Numerical Analysis and Computational Procedures, Allied Books
3. M. Pal, Numerical Analysis Scientists Engineers Programs, Alpha Science
4. S. S. Satry, Introductory Methods of Numerical Analysis, PHI

DSE-2P: Numerical Methods Practical **Practical: 60 Lecture**

	Course Outcome
1.	Ability to find solution of an algebraic polynomial through programming
2.	Find solution of system of linear equations by implementing suitable numerical algorithm
3.	Solve a given integration problem by implementing appropriate numerical integration and find error present in the solution obtained

4.	Obtain differential equation solution using methods like Runge Kutta method

Implement following using a appropriate programming language:

1. Interpolation: Newton forward & backward.
2. Numerical Integration: Trapezoidal Rule, Simson's 1/3 Rule, Simson's 3 /8 Rule.
3. Numerical solution of a system of linear equation: Gauss elimination.
4. Algebraic Equation: Bisection, Regula-falsi, Newton Raphson
5. Ordinary Differential Equation: Taylor Series, Euler's method, Runge Kutta.

DSE2T: Computational Biology **Theory: 60 Lectures**

	Course Outcome
1.	Differentiate biological algorithms and computer algorithms
2.	Capacity to analyze DNA sequence and subsequence
3.	Appreciate the application of gene microarray technology

Unit 1. Biological Algorithms versus Computer Algorithms, Algorithmic Notations. Introduction to Algorithm Design Techniques. (5 Lectures)

Unit 2. Introductory Molecular Biology, DNA Analysis, Regulatory Motifs in DNA Sequences, Finding Motifs, (12 Lectures)

Unit 3. Greedy Approach to Motif finding, Longest Common Subsequences, Global and Local Sequence Alignments, Multiple Alignment, Gene Prediction. (10 Lectures)

Unit 4. Constructing Algorithms in sub-quadratic time, Shortest Superstring Problem, Sequencing by Hybridization, Protein Sequencing and Hybridization, Spectrum Graphs, Spectral Convolution, Repeat Finding. (15 Lectures)

Unit 5. Transcriptome and Evolution: Regulation - Transcription regulation, microarray technology, expression clustering, (10 Lectures)

Unit 6. DNA binding sites, location analysis, regulatory motif prediction, Ribozymes, RNA World, RNA secondary structure, non-coding RNAs. (8 Lectures)

Recommended Books:

1. Anna Tramontano and Hershel Safer, Introduction to Bioinformatics, CRC Press

2. Harisha, Fundamental of Bioinformatics, Wiley

3. Lesk, Introduction to Bioinformatics, OUP

4. Bosu, Thukral, Bioinformatics: Experiments, Tools, Databases, and Algorithms, OUP

DSE2P: Computational Biology Lab **Practical: 60 Lectures**

	Course Outcome
1.	Familiarization with different tools for bioinformatics
2.	Know the different databases and knowhow for using those dataset
3.	Implement DNA sequencing and alignment
4.	Analyze protein motifs

1. Sequence Databases: EMBOSS, NCBI ToolKit, Expassy tools
2. Search tools against Databases:
    i. BLAST
    ii. FASTA

3. Pair wise alignment:
    a. Dot Plot
    b. Global and Local alignment methods
4. Multiple sequence alignment:
    a. Clustal
    b. Dialign
    c. Multalign
5. Primary and secondary structure prediction methods
    a. GOR Method
    b. PSI-pred
    c. Chou-Fasman method
6. Sequence patterns and profiles:
    a. generation of sequence profiles
        i. PSI-BLAST
    b. derivation of and searching sequence patterns:
        i. MEME/MAST
        ii. PHI-BLAST
7. Protein motif and domain analysis:
    a. MEME/MAST
    b. eMotif

DSE2 Finite Automata Credits 06 (6+0) **Theory: 60 Lectures**

	Course Outcome
1.	Explain to understand the concept of machines: finite automata, pushdown automata, linear bounded automata, and Turing machines.
2.	Determine The formal languages and grammars: regular grammar and regular languages, context-free languages and context-free grammar; and introduction to context-sensitive language and context-free grammar, and unrestricted grammar and languages.
3.	Evaluate The relation between these formal languages, grammars, and machines.
4.	Formulate The complexity or difficulty level of problems when solved using these machines.
5.	Develop and access The concept of algorithm.
6.	Design to compare the complexity of problems.

Unit 1. Introduction: Alphabet, strings and languages. Definable languages. (4 Lectures)

Unit 2. Regular languages ad Finite Automata: deterministic and non-deterministic finite automaton (DFA and NFA). Equivalence of DFA and NFA. Minimization of DFA. Some closure properties of regular languages. Regular expressions, construction of NFA and DFA from a regular expression. Pumping lemma and existence of non-regular languages. (15 Lectures)

Unit 3. Context-free languages (CFL) and Pushdown Automata: context-free grammar (CFG), push-down automaton (PDA), equivalence of CFG and PDA. Closure properties of CFLs. Chomsky normal form (CNF). Context-sensitive grammar and language. (15 Lectures)

Unit 4. Turing Machine: Turing machine recognizable or recursively enumerable languages, recursive languages. Variants of Turing machine model and their equivalence. Decision problems of formal languages. Language not recognized by Turing machine and undecidable problems of formal language. (16 Lectures)

Unit 5. Introduction to Complexity Classes: P, NP, NPC, NP-Hard. (10 Lectures)

Reference Books:

1. Michael Sipser, *Introduction to Theory of Computation*, Cengage Learning.

2. Hopcroft, Motwani & Ullman, *Introduction to Automata Theory, Languages, and Computation*, Pearson Education.

3. Mishra & Chandrasekaran, *Theory of Computer Science, Automata, languages and Computation*, PHI.

4. Lewis and Papadimitrou, *Elements of the theory of Computation*, PHI.

5. Linz Peter, *An Introduction to Formal Languages and Automata*, Narosa.

DSE3 Seminar Credits 06 (0+6)

Each student will be given a list of two seminar topics related to computer science and applications and they are to submit a seminar report and give presentation on the topics. First seminar should be presented during middle of semester, and second seminar be presented during end of semester.

DSE3T: Digital Image Processing **Theory: 60 Lectures**

	Course Outcome
1.	Review the fundamental concepts of a digital image processing system.
2.	Analyze images in the frequency domain using various transforms.
3.	Evaluate the techniques for image enhancement and image restoration.
4.	Categorize various compression techniques.
5.	Interpret Image compression standards.
6.	Interpret image segmentation and representation techniques.

1. **Introduction**: Light, Brightness adaption and discrimination, Pixels, coordinate conventions, Imaging Geometry, Perspective Projection, Spatial Domain Filtering, sampling and quantization. (10 Lectures)

2. **Spatial Domain Filtering**: Intensity transformations, contrast stretching, histogram equalization, Correlation and convolution, Smoothing filters, sharpening filters, gradient and Laplacian. (10 Lectures)

4. **Image Restoration**: Basic Framework, Interactive Restoration, Image deformation and geometric transformations, image morphing, Restoration techniques, Noise characterization, Noise restoration filters, Adaptive filters. (12 Lectures)

5. **Image Compression**: Encoder-Decoder model, Types of redundancies, Lossy and Lossless compression, Entropy of an information source, Shannon's 1st Theorem, Huffman Coding, Arithmetic Coding, LZW coding. (10 Lectures)

6. **Image Segmentation**: Boundary detection based techniques, Point, line detection, Edge detection, Edge linking, (8 Lectures)

Recommended Books:

1. Chanda, Majumder, Digital Image Processing, PHI

2. Malay Pakhira, Digital Image Processing and Pattern Recognition, PHI

3. Gonzalez, Woods, Digital Image Processing, Pearson

DSE3P: Digital Image Processing Lab **Practical: 60 Lectures**

	Course Outcome
1.	Familiarity with MATLAB or open source image processing tool such as Octave
2.	Apply some basic operations an image
3.	Write and execute programs applying arithmetic and/or logic operations on image
4.	Able to restore a noisy image up to a certain extent

List of Practical:

1. Write program to read and display digital image using MATLAB or SCILAB
    a. Become familiar with SCILAB/MATLAB Basic commands
    b. Read and display image in SCILAB/MATLAB
    c. Resize given image
    d. Convert given color image into gray-scale image
    e. Convert given color/gray-scale image into black & white image
    f. Draw image profile
    g. Separate color image in three R G & B planes
    h. Create color image using R, G and B three separate planes
2. To write and execute image processing programs using point processing method
    a. Obtain Negative image
    b. Obtain Flip image
    c. Thresholding
3. To write and execute programs for image arithmetic operations
    a. Addition of two images
    b. Subtract one image from other image
    c. Calculate mean value of image
    d. Different Brightness by changing mean value
4. To write and execute programs for image logical operations
    a. AND operation between two images
    b. OR operation between two images
    c. Calculate intersection of two images
    d. Water Marking using EX-OR operation
    e. NOT operation (Negative image)
5. To write a program for histogram calculation and equalization
6. To write and execute program for geometric transformation of image

a. Translation

b. Scaling

c. Rotation

d. Shrinking

e. Zooming

7. To understand various image noise models and to write programs for

a. Image restoration

b. Remove Salt and Pepper Noise

c. Minimize Gaussian noise

d. Median filter

8. Write and execute programs to remove noise using spatial filters

a. Understand 1-D and 2-D convolution process

b. Use 3x3 Mask for low pass filter and high pass filter

9. Write and execute programs for image frequency domain filtering

a. Apply FFT on given image

b. Perform low pass and high pass filtering in frequency domain

c. Apply IFFT to reconstruct image

DSE3T: Machine Learning **Theory: 60 Lectures**

	Course Outcome
1.	To Understand a wide variety of learning algorithms.
2.	To understand how to apply a variety of learning algorithms to data using various tools of Machine Learning.
3.	To identify the strengths and weaknesses of many popular machine learning approaches.
4.	To analyze the performance of learning algorithms and model selection
5.	To identify mathematical relationships within and across Machine Learning algorithms and the paradigms of supervised and unsupervised learning.
6.	To apply machine learning techniques in solving complex real world problems.

Unit 1. Basics: Introduction to Machine Learning - Different Forms of Learning,

Basics of Probability Theory. (8 Lectures)

Unit 2. Regression Analysis: Linear Regression, Ridge Regression, Bayesian Regression. (8 Lectures)

Unit 3. Classification Methods: Instance-Based Classification, Linear Discriminant Analysis, Logistic Regression, Large Margin Classification, Kernel Methods, Support Vector Machines, Multi-class Classification, Classification and Regression Trees. (10 Lectures)

Unit 4. Neural Networks: Non-linear Hypotheses, Neurons and the Brain, Model Representation, Multi-layer Networks, Back-propagation, Multi-class Discrimination, Training Procedures, Localized Network Structure, Deep Learning. (10 Lectures)

Unit 5. Graphical Models: Hidden Markov Models, Bayesian Networks. (6 Lectures)

Unit 6. Clustering: Partitional Clustering - K-Means, K-Medoids, Hierarchical Clustering - Agglomerative, Divisive, Distance Measures, Density Based Clustering. (8 Lectures)

Unit 7. Dimensionality Reduction: Principal Component Analysis, Independent Component Analysis. (5 Lectures)

Unit 8. Reinforcement Learning: Q-Learning, Temporal Difference Learning. (5 Lectures)

Reference Books:

1. C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer.

2. R. O. Duda, P. E. Hart, and D.G. Stork, *Pattern Classification*, Wiley.

3. T. M. Mitchell, *Machine Learning*, McGraw-Hill.

DSE3P: Machine Learning Lab **Practical: 60 Lectures**

	Course Outcome
1.	Familiarity with different tools for machine learning
2.	Learn to find standard datasets
3.	Implement linear regression
4.	Implement different clustering algorithms
5.	Implement different classification algorithms

The programs can be implemented in either Python or MATLAB.

Data sets can be taken from standard repositories (*https://archive.ics.uci.edu/ml/datasets.html*).

1. Implement and demonstrate the FIND-S algorithm for finding the most specific hypothesis based on a given set of training data samples. Read the training data from a .CSV file.

2. For a given set of training data examples stored in a .CSV file, implement and demonstrate the Candidate-Elimination algorithm to output a description of the set of all hypotheses consistent with the training examples.

3. Write a program to demonstrate the working of the decision tree based ID3 algorithm. Apply this knowledge to classify a new sample.

4. Build an Artificial Neural Network by implementing the Back propagation algorithm and test the same using appropriate data sets.

5. Write a program to implement the naive Bayesian classifier for a sample training data set stored as a .CSV file.

6. Assuming a set of documents that need to be classified, use the naive Bayesian Classifier model to perform this task.

7. Apply EM algorithm to cluster a set of data stored in a .CSV file. Use the same data set for clustering using k-Means algorithm.

8. Implement the Regression algorithm in order to fit data points. Select appropriate data set for your experiment and draw graphs.

DSE4T: Systems Programming **Theory: 60 Lectures**

	Course Outcome
1.	Understand the role of systems software in functioning of a system
2.	Comprehend the function of assembler, loader and linker

3.	Appreciate the different phases of compilation and their tasks
4.	Apply the knowledge of parsing to successfully (or unsuccessfully) parse a given string

Course Contents:

Unit 1. Introduction: Overview of compilation, Phases of a compile. (6 Lectures)

Unit 2. Assemblers & Loaders, Linkers: One pass and two pass assembler, design of an assembler, Absolute loader, relocation and linking concepts, relocating loader and Dynamic Linking. (12 Lectures)

Unit 3. Lexical Analysis: Role of a Lexical analyzer, Specification and recognition of tokens, Symbol table, lex. (6 Lectures)

Unit 4. Parsing: Top-down parsing, LL(1) parsing, Bottom up parsing- LR parser, yacc. (12 Lectures)

Unit 5. Intermediate representations: Three address code generation, syntax directed translation, translation of types, control statements. (10 Lectures)

Unit 6. Storage organization: Activation records stack allocation. (8 Lectures)

Unit 7. Code Generation: Object code generation (6 Lectures)

Recommended Books:

1. Santanu Chattopadhyaya, Systems Programming, PHI, 2011.

2. Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman, Compilers: Principles, Techniques, and Tools, 2 nd edition, Prentice Hall, 2006.

3. D. M. Dhamdhere, Systems Programming, Tata McGraw Hill, 2011.

4. Leland Beck, D. Manjula, System Software: An Introduction to System Programming, 3rd edition, Pearson Education, 2008.

5. Grune D, Van Reeuwijk . K, Bal H. E, Jacobs C J H, Langendoen K, Modern Compiler Design, 2nd edition, Springer, 2012

DSE4P: Systems Programming Lab **Practical: 30 Lectures**

	Course Outcome
1.	Implement a simple assembler
2.	Implement a simple lexical analyzer by following Lex.
3.	Implement a simple parser, by following YACC

List of Practicals:

1. To implement an assembler for a hypothetical language.

2. To get familiar with lex: write a program to recognize numbers, identifiers.

3. To get familiar with yacc: write a desk calculator.

DSE4T Cloud Computing **Theory: 60 Lectures**

	Course Outcome
1.	To interpret the main concepts, key technologies, strengths, and limitations of cloud computing and the possible applications for stateof-the-art cloud computing,
2.	To illustrate various problems and evaluate related cloud computing solutions

3.	To apply the architecture and infrastructure of cloud computing, including SaaS, PaaS, IaaS, public cloud, private cloud and hybrid cloud to different problems
4.	To analyze cloud provider for a defined environment and to a specific platform in a cost effective way.
5.	To analyze case studies to derive the best practice model to apply when developing and deploying cloud based applications
6.	To understand recent trends and applications in the Cloud computing.

Unit 1. ***Introduction***: Introduction to Cloud Computing, Origin of Cloud Computing. Fundamental concepts of Distributed Systems, Cluster Computing, Grid Computing, and Mobile Computing.

(10 Lectures)

Unit 2. ***Cloud Models***: Basics of Cloud Computing Concepts, Characteristics of Cloud Computing, Need for Cloud, Cloud Deployment models: private, public, hybrid and community cloud. (8 Lectures)

Unit 3. ***Cloud Services***: Resource-as-a-Service (RaaS), Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS), Examples of each service. RaaS: Usage of Physical resources like servers, networks, data center etc,

IaaS: Virtualization, PaaS: Integrated lifecycle platform: Google App Engine, Microsoft Azure, Anchored life cycle platform: Salesforce platform,

SaaS: Characterizing SaaS, Salesforce's software environment. (20 Lectures)

Unit 4. ***Resource Scheduling for Cloud Computing***: - Introduction, Virtual Machine provisioning and Migration Services, Scheduling techniques of Virtual machines for resource reservation, Cloud Service Scheduling hierarchy, Economic models for Resource-allocation scheduling , Heuristic Models for task – execution scheduling : Static Strategies , Dynamic Strategies , Heuristic Schedulers. (12 Lectures)

Reference Books:

3. Kumar Saurabh, *Cloud Computing – Insight into New Era Infrastructure*, Wiley India.
4. John Rhoton, *Cloud Computing Explained*, Recursive Press.
5. Barry Sosinsky, *Cloud Computing Bible*, Wiley.
6. Rajkumar Buyya, James Broberg, *Cloud Computing: Principles and Paradigms*, Wiley.
7. Judith Hurwiz, *Cloud Computing for Dummies*, Wiley Publishing.

DSE4P: Cloud Computing Lab Practical: 60 Lectures

	Course Outcome
1.	Familiarity with Amazon AWS, Microsoft Azure, Google Cloud Platform
2.	Able to create virtual machine

1. Create virtual machines that access different programs on same platform.
2. Create virtual machines that access different programs on different platforms.
3. Working on tools used in cloud computing online-
   a) Storage
   b) Sharing of data
   c) Manage your calendar, to-do lists,

d) A document editing tool

4. Exploring Google cloud

5. Exploring Microsoft cloud

6. Exploring Amazon cloud

DSE4 Project Work / Dissertation **Credits 06 (0+6)**

The students will be allowed to work on any project based on the concepts studied in core / elective or skill based elective courses. Maximum of four students can participate in a group for implemeting a project. Students should submit a project report and give a presentation demonstrating their project work at the end of semster.

Skill Enhancement Courses (2 Papers to be selected) – SEC 1, SEC 2.

SEC1 Elective List:

1. Android Programming (2 Credit)
2. Programming in Python (2)
3. HTML Programming (2)

SEC2 Elective List:

4. UNIX / LINUX Programming (2)
5. R Programming (2)
6. Programming in MALAB (2)

SEC1T: Android Programming **Theory: 30 Lectures**

	Course Outcome
1.	Revisit Java programming
2.	Learn Android syntax
3.	Apply Android to design a GUI
4.	Establish connection with backend database

Unit 1. Introduction: History of Android, Introduction to Android Operating Systems, Android Development Tools, Android Architecture. (4 Lectures)

Unit 2. Overview of object oriented programming using Java: OOPs Concepts: Inheritance, Polymorphism, Interfaces, Abstract class, Threads, Overloading and Overriding, Java Virtual Machine. (8 Lectures)

Unit 3. Development Tools: Installing and using Eclipse with ADT plug-in, Installing Virtual machine for Android sandwich/Jelly bean (Emulator), configuring the installed tools, creating a android project – Hello Word, run on emulator, Deploy it on USB-connected Android device. (8 Lectures)

Unit 4. User Interface Architecture: Application context, intents, Activity life cycle, multiple screen sizes. (3 Lectures)

Unit 5. User Interface Design: Form widgets, Text Fields, Layouts, Button control, toggle buttons, Spinners (Combo boxes), Images, Menu, Dialog. (4 Lectures)

Unit 6. Database: Understanding of SQLite database, connecting with the database. (3 Lectures)

Book Recommended:

1. Android application development for java programmers. By James C. Sheusi. Cengage Learning, 2013.

Online Reading / Supporting Material:

1. http://www.developer.android.com

SEC1P: Software Lab Based on Android Programming **Practical: 30 Lectures**

	Course Outcome
1.	Create a simple GUI for log-in page
2.	Create a GUI with menu bar, text box, radio button etc
3.	Develop an App and make it available in Google play store
4.	Establish connection with backend database

1. Create ―Hello World application. That will display ―Hello World in the middle of the screen in the emulator. Also display ―Hello World‖ in the middle of the screen in the Android Phone.

2. Create an application with login module. (Check username and password).

3. Create spinner with strings taken from resource folder (res >> value folder) and on changing the spinner value, Image will change.

4. Create a menu with 5 options and and selected option should appear in text box.

5. Create a list of all courses in your college and on selecting a particular course teacher-in-charge of that course should appear at the bottom of the screen.

6. Create an application with three option buttons, on selecting a button colour of the screen will change.

7. Create and Login application as above. On successful login, pop up the message.

8. Create an application to Create, Insert, update, Delete and retrieve operation on the database.

SEC1T: Programming in Python **Theory: 30 Lectures**

	Course Outcome
1.	Learn syntaxes of Python
2.	Write simple programs in Python for solving simple computation problems
3.	Learn the rich set of libraries
4.	Process strings and lists

Unit 1. Overview of Python Programming: Structure of a Python Program, Elements of Python

Introduction to Python: Python Interpreter, Using Python as calculator, Python shell, Indentation. Atoms, Identifiers and keywords, Literals, Strings, Operators (Arithmetic operator, Relational operator, Logical or Boolean operator, Assignment, Operator, Ternary operator, Bit wise operator, Increment or Decrement operator) (10 Lectures)

Unit 2. Creating Python Programs: Input and Output Statements, Control statements (Branching, Looping, Conditional Statement, Exit function, Difference between break, continue and pass.),

Iteration: Conditional execution, Alternative execution, Nested conditionals, Multiple assignment, The while statement, Tables, Two-dimensional tables. (8 Lectures)

Unit 3. Function: Defining Functions, default arguments, The return statement, Recursion, Stack diagrams for recursive functions.

Errors and Exceptions. (6 Lectures)

Unit 4. Strings and Lists: String as a compound data type, Length, Traversal and the for loop, String slices, String comparison, A find function, Looping and counting, List values, Accessing elements, List length, List membership, Lists and for loops, List operations, List deletion. Cloning lists, Nested lists

(6 Lectures)

Recommended Books:

1. T. Budd, Exploring Python, TMH, 1st Ed, 2011

2. Core Python Programming by Chun, Pearson 2007.

SEC1P: Software Lab Based on Programming in Python **Practical: 30 Lecture**

	Course Outcome
1.	Write simple programs for solving given computation problems
2.	Design user defined functions
3.	Implement some data structures using Python
4.	Develop some string and list processing functions

1. Using for loop, print a table of Celsius/Fahrenheit equivalences. Let c be the Celsius temperatures ranging from 0 to 100, for each value of c, print the corresponding Fahrenheit temperature.

2. Using while loop, produce a table of sins, cosines and tangents. Make a variable x in range from 0 to 10 in steps of 0.2. For each value of x, print the value of sin(x), cos(x) and tan(x).

3. Write a program that reads an integer value and prints —leap year or —not a leap year.

4. Write a program that takes a positive integer n and then produces n lines of output shown as follows.

For example enter a size: 5

*

**

***

****

*****

5. Write a function that takes an integer ‘n‘ as input and calculates the value of 1 + 1/1! + 1/2!

+ 1/3! + ... + 1/n!

6. Write a function that takes an integer input and calculates the factorial of that number.

7. Write a function that takes a string input and checks if it‘s a palindrome or not.

8. Write a list function to convert a string into a list, as in list (‘abc‘) gives [a, b, c].

9. Write a program to generate Fibonacci series.

10. Write a program to check whether the input number is even or odd.

11. Write a program to compare three numbers and print the largest one.

12. Write a program to print factors of a given number.

13. Write a method to calculate GCD of two numbers.

14. Write a program to implement linear search on lists.

15. Write a program to sort a list.

SEC1T: HTML Programming **Theory: 30 Lectures**

	Course Outcome
1.	Learn different tabs of HTML
2.	Design a static web page
3.	Design a dynamic webpage

Unit 1. Basics: Head, Body, Colors, Attributes, Lists, ordered and unordered. (6 Lectures)

Unit 2. Tables: Creating a Table, Table Headers, Captions, Spanning Multiple Columns, Styling Table. (4 Lectures)

Unit 3. Links: Hyperlinks, Relative Links, Absolute Links, Link Attributes, Using the ID Attribute to Link Within a Document. (4 Lectures)

Unit 4. Images: Putting an Image on a Page, Using Images as Links, Putting an Image in the Background. (6 Lectures)

Unit 5. Forms: Basic Input and Attributes, Other Kinds of Inputs, Styling forms with CSS. (10 Lectures)

SEC1P: Software Lab Based on HTML **Practical: 30 Lectures**

	Course Outcome
1.	Create a simple web page
2.	Ability to create an web page accessed in daily life such as department of an Institution

Q.1 Create an HTML document with the following formatting options:

a. Bold

b. Italics

c. Underline

d. Headings (Using H1 to H6 heading styles)

e. Font (Type, Size and Color)

f. Background (Colored background/Image in background)

g. Paragraph

h. Line Break

i. Horizontal Rule

j. Pre tag

Q.2 Create an HTML document which consists of:

a. Ordered List

b. Unordered List

c. Nested List

d. Image

Q.3 Create an HTML document which implements Internal linking as well as External linking.

Q.4 Create a table using HTML which consists of columns for Roll No., Student‘s name and grade.

Q.5 Create a table using HTML where cells are not uniform in size, and some cells are merged and contains an image in a cell like given below.

Q.6 Create a form using HTML which has the following types of controls :

a. Text Box

b. Option/radio buttons

c. Check boxes

d. Reset and Submit buttons

Q.7 Create HTML documents (having multiple frames) in the following three formats:

Frame I

Frame II

Frame I

Frame II Frame III

SEC2T: Linux / Unix Programming **Theory: 30 Lectures**

	Course Outcome

1.	Understand Linux OS architecture
2.	Ability to install a Linux OS
3.	Apply system administration related tasks such as user creation
4.	Develop and execute shell scripts

Unit 1. Introduction: (10 Lectures)

What is linux / unix Operating Systems

- Difference between linux/unix and other operating systems
- Features and Architecture
- Various Distributions available in the market
- Installation, Booting and shutdown process
- System processes (an overview)
- External and internal commands
- Creation of partitions in OS
- Processes and its creation phases – Fork, Exec, wait

Unit 2. User Management and the File System: (8 Lectures)

- Types of Users, Creating users, Granting rights
- User management commands

File quota and various file systems available

File System Management and Layout, File permissions

Login process, Managing Disk Quotas

Links (hard links, symbolic links

Unit 3. Shell introduction and Shell Scripting: (12 Lectures)

- What is shell and various type of shell, Various editors present in linux
- Different modes of operation in vi editor
- What is shell script, Writing and executing the shell script
- Shell variable (user defined and system variables)
- System calls, Using system calls
- Pipes and Filters
- Decision making in Shell Scripts (If else, switch), Loops in shell Functions
- Utility programs (cut, paste, join, tr , uniq utilities)
- Pattern matching utility (grep)

Recommended Books:

1. Sumitabha, Das, Unix Concepts And Applications, Tata McGraw-Hill Education, 2006

2. Nemeth Synder & Hein, Linux Administration Handbook, Pearson Education, 2nd Edition 2010

3. W. Richard Stevens, Bill Fenner, Andrew M. Rudoff, Unix Network Programming, The sockets Networking API, Vol. 1, 3rd Edition,2014

SEC2P: Software Lab Based on Linux **Practical: 30 Lectures**

	Course Outcome
1.	Implement shell scripts for basic computational and system level tasks
2.	Implement shell scripts for system administration level tasks
3.	Understand the Linux system booting and shut down process

1. Write a shell script to check if the number entered at the command line is prime or not.
2. Write a shell script to modify cal command to display calendars of the specified months.
3. Write a shell script to modify cal command to display calendars of the specified range of months.
4. Write a shell script to accept a login name. If not a valid login name display message – ― *Entered login name is invalid*.
5. Write a shell script to display date in the mm/dd/yy format.
6. Write a shell script to display on the screen sorted output of ―who command along with the total number of users .
7. Write a shell script to display the multiplication table any number,
8. Write a shell script to compare two files and if found equal asks the user to delete the duplicate file.
9. Write a shell script to find the sum of digits of a given number.
10. Write a shell script to merge the contents of three files, sort the contents and then display them page by page.
11. Write a shell script to find the LCD (least common divisor) of two numbers.
12. Write a shell script to perform the tasks of basic calculator.
13. Write a shell script to find the power of a given number.
14. Write a shell script to find the binomial coefficient C(n , x).
15. Write a shell script to find the permutation P(n,x).
16. Write a shell script to find the greatest number among the three numbers.
17. Write a shell script to find the factorial of a given number.
18. Write a shell script to check whether the number is Armstrong or not.
19. Write a shell script to check whether the file have all the permissions or not.

SEC2T: R-Programming **Theory: 30 Lectures**

	Course Outcome
1.	Learn syntax of R programming
2.	Implement R programs for simple computational tasks
3.	Develop R programming for data processing

Unit 1. Introduction: Overview and History of R, Getting Help, Data Types, Subsetting, Vectorized Operations, Reading and Writing Data. (12 Lectures)

Unit 2. Control Structures, Functions, lapply, tapply, split, mapply, apply, Coding Standards. Scoping Rules, Debugging Tools, Simulation, R Profiler (18 Lectures)

Recommended Books:

1. William N. Venables and David M. Smith, An Introduction to R. 2 nd Edition. Network Theory Limited. 2009

2. Norman Matloff, The Art of R Programming - A Tour of Statistical Software Design, No Starch Press.2011

SEC2P: Software Lab Based on R Programming **Practical: 30 Lectures**

	Course Outcome
1.	Learn R programming syntax
2.	Implement R programs for solving simple computational tasks
3.	Implement R programs for data processing tasks

1. Write a program that prints ‗Hello World' to the screen.
2. Write a program that asks the user for a number n and prints the sum of the numbers 1 to n
3. Write a program that prints a multiplication table for numbers up to 12.
4. Write a function that returns the largest element in a list.
5. Write a function that computes the running total of a list.
6. Write a function that tests whether a string is a palindrome.
7. Implement the following sorting algorithms: Selection sort, Insertion sort, Bubble Sort
8. Implement linear search.
9. Implement binary search.
10. Implement matrices addition, subtraction and Multiplication

SEC-2T: Programming in MATLAB **Theory: 30 Lectures**

	Course Outcome
1.	Learn Matlab commands
2.	Write Matlab programs for solving different kinds of problems
3.	Define Matlab user-defined functions and use it in Matlab scripts
4.	Plot graphs with labels and legends
5.	Design a GUI

Unit 1. Introduction to Programming: Components of a computer, working with numbers, Machine code, Software hierarchy

Programming Environment: MATLAB Windows, A First Program, Expressions, Constants, Variables and assignment statement, Arrays. (8 Lectures)

Unit 2. Graph Plots: Basic plotting, Built in functions,

Procedures and Functions: Arguments and return values, M-files, Formatted console input-output, String handling. (10 Lectures)

Unit 3. Control Statements: Conditional statements: If, Else, Else-if, Repetition statements: While, for loop.

Manipulating Text: Writing to a text file, Reading from a text file, Randomising and sorting a list, searching a list. (6 Lectures)

Unit 4. GUI Interface: Attaching buttons to actions, Getting Input, Setting Output. (6 Lectures)

Recommended Books:

1. MATLAB: An Introduction with Applications, by Amos Gilat, 2nd edition, Wiley, 2004,

2. C.B. Moler, Numerical Computing with MATLAB, SIAM, 2004.

SEC1P: Software Lab Based on MatLab **Practical: 30 Lectures**

	Course Outcome
1.	Learn to use Matlab commands
2.	Implement Matlab programs for solving different kinds of problems
3.	Define Matlab user-defined functions and use it in Matlab scripts
4.	Plot graphs with labels and legends
5.	Design a simple GUI

1. Write a program to assign the following expressions to a variable A and then to print out the value of A.

a) (3+4)/(5+6)

b) $2\pi^2$

c) $\sqrt{2}$

d) $(0.0000123 + 5.67 \times 10^{-3}) \times 0.4567 \times 10^{-4}$

2. Celsius temperatures can be converted to Fahrenheit by multiplying by 9, dividing by 5, and adding 32. Assign a variable called C the value 37, and implement this formula to assign a variable F the Fahrenheit equivalent of 37 Celsius.

3. Set up a vector called N with five elements having the values: 1, 2, 3, 4, 5. Using N, create assignment statements for a vector X which will result in X having these values:

a. 2, 4, 6, 8, 10

b. 1/2, 1, 3/2, 2, 5/2

c. 1, 1/2, 1/3, 1/4, 1/5

d. 1, 1/4, 1/9, 1/16, 1/25

4. A supermarket conveyor belt holds an array of groceries. The price of each product (in pounds) is [0.6, 1.2 ,0.5, 1.3] ; while the numbers of each product are [3, 2 ,1 ,5]. Use MATLAB to calculate the total bill.

5. The sortrows(x) function will sort a vector or matrix X into increasing row order. Use this

function to sort a list of names into alphabetical order.

6. The identity matrix is a square matrix that has ones on the diagonal and zeros elsewhere. You can generate one with the eye() function in MATLAB. Use MATLAB to find a matrix B, such that when multiplied by matrix A=[1 2; -1 0] the identity matrix I=[1 0; 0 1] is generated. That is A*B=I.

7. Create an array of N numbers. Now find a single MATLAB statement that picks out from that array the 1,4,9,16,...,$\sqrt{N^{th}}$ entries, i.e. those numbers which have indices that are square numbers.

8. Draw a graph that joins the points (0,1), (4,3), (2,0) and (5,-2).

9. Write a function called FtoC (ftoc.m) to convert Fahrenheit temperatures into Celsius. Make sure the program has a title comment and a help page. Test from the command window with:

   a. FtoC(96)

   b. lookfor Fahrenheit

   c. help FtoC

10. Write a program to input 2 strings from the user and to print out

   a. the concatenation of the two strings with a space between them,

   b. a line of asterisks the same length as the concatenated strings, and

   c. the reversed concatenation.