NEP 2020 Syllabus

B.Sc. in Computer Science (Major-Minor)

Program name	Eligibility Criteria of the programme, if any	Semester	Course name	Course code	credits	Credit distribution of the course			Pre-requisite	**Theory marks**		Practical Marks	Total Marks	Remarks
						L	T	P		Internal	External			
FYUGP in Computer Science Major-Minor	H.S (Science) with Mathematics as a subject securing the minimum pass mark	I	Computer Fundamentals and Programming	COM010104	4	3	0	1	No	30	45	25	100	Compulsory
		II	Computer Organization	COM020104	4	4	0	0	No	40	60	0	100	Compulsory
		III	Object Oriented Programming using C++	COM030104	4	3	0	1	No	30	45	25	100	Compulsory
		IV	Data Structure	COM040104	4	3	0	1	No	30	45	25	100	Elective II
			Database Management System	COM040204	4	3	0	1	No	30	45	25	100	Compulsory • Content same as CIT061104
			Mathematical Foundation of Computer Science	COM040304	4	4	0	0	No	40	60	0	100	Elective III
			Operating System	COM040404	4	3	0	1	No	30	45	25	100	Elective I • Content same as CIT040404
		V	Computer Networks	COM050104	4	3	0	1	No	30	45	25	100	Elective III • Content same as CIT050404

			Java Programming	COM050204	4	3	0	1	No	30	45	25	100	Elective II • Content of COM050204 is same as CIT050304
			Python Programming	COM050304	4	3	0	1	No	30	45	25	100	
			Software Engineering	COM050404	4	4	0	0	No	40	60	0	100	Elective I • Content same as CIT050104
			Web Technologies	COM050504	4	3	0	1	No	30	45	25	100	Compulsory • Content same as CIT050204
		VI	Automata Theory and Languages	COM060104	4	4	0	0	No	40	60	0	100	Compulsory
			Cloud Computing	COM060204	4	4	0	0	No	40	60	0	100	Elective III
			Compiler Design	COM060304	4	4	0	0	No	40	60	0	100	
			Artificial Intelligence	COM060404	4	3	0	1	No	30	45	25	100	Elective II • Content of COM060404 is same as CIT060404
			Computer Graphics	COM060504	4	3	0	1	No	30	45	25	100	• Content of COM060504 is same as CIT060104
			Data Mining and Warehousing	COM060604	4	3	0	1	No	30	45	25	100	• Content of COM060604 is same as CIT060604

			Design and Analysis of Algorithms	COM060704	4	4	0	0	No	40	60	0	100	Elective I • Content of COM060804 is same as CIT060904
			Graph Theory	COM060804	4	4	0	0	No	40	60	0	100	

COM010104: COMPUTER FUNDAMENTALS AND PROGRAMMING

1. **Learning Outcome:**
   - At the end of the course, students will be able to:
   - Understand the basics of Computer and programming
   - Adopt algorithmic approach to solve problems using pseudocode and flowcharts
   - Understand and write programs in C to implement conditions, loops, functions and other
   - programming constructs
   - Work on arrays, strings and basic file operations in C

2. **COURSE OUTCOMES:**

   At the end of the course, students will be able to:

   CO1: Identify different I/O devices, memory, number systems and data representations.
   CO2: Associate with data types, operators, inbuilt functions, principle of control structures, arrays and string in C programming.
   CO3: Explain functions, file processing techniques in C programming.
   CO4: Distinguish user defined data structures from primitive data structures.
   CO5: Solve programming problems that are related to functions, pointers and dynamic memory allocation.

3. **Prerequisite:** NIL
4. **Semester: 1**
5. **Course Type: Compulsory**
6. **Course Level: 100-199**
7. **Theory Credit:** 3
8. **Practical Credit:** 1
9. **Number of required hours:**
   a) Theory: 45 hrs
   b) Practical: 30 hrs
   c) Non Contact: 5 hrs

10. **List of reference books:**

a) B.S. Gottfried, “Schaum's Outline of Theory and Problems of Programming with C”, Mcgraw-Hill, 2007.
b) B. Kernighan, D. Ritchie, “The C Programming Language”, Second Edition, Prentice Hall, 1988
c) E. Balaguruswami, “Programming in ANSI C”, 2nd Ed., Tata McGraw Hill, 2004.
d) V. Rajaraman, “Fundamentals of Computer”, 4th Ed., PHI, 2006
e) R. Thareja, “Computer Fundamentals & Programming in C”, Oxford University Press, 2013.

11. Detailed Syllabus:

Unit 1: Computer Fundamentals (9 Lectures)

Introduction to computer hardware, software– application and system software. Operating systems. Major components of a Digital Computer – ALU and CU, Memory – primary and secondary memory. Storage devices – magnetic storage devices, optical storage devices, Input devices– mouse, keyboard, touch-screen, scanner etc., output devices – CRT/LCD/LED monitors, printers etc. Number systems – binary, octal, hexadecimal, BCD. Conversion between two number systems. Signed magnitude, 1's complement and 2's complement representation. Character encodings – ASCII, EBCDIC, Unicode. Basic overview of networks and the Internet, WWW.

Unit 2: Programming Basics (4 Lectures)

Introduction to programming languages. Low-level and high-level language and their characteristics. Compiler vs. interpreter. IDE. Bugs and its types. Algorithms, pseudocodes and flowcharts. Overview of the C programming language. Structure of a C program.

Unit 3: Datatypes and Operators (5 Lectures)

Basic data types in C - integers, floats, doubles, characters, and void. Size and range of values of data types. Variables. Declaring variables. Operators and expressions, Input and output statements – getchar(), getc(), getch(), putchar(), putc(), puts(), scanf(), printf(), format specifiers. Typecasting. Operators in C – binary and unary operators. Arithmetic, assignment, logical, comparison, bitwise and conditional operators. Order of precedence of operators. Associativity of operators. Expressions and statements in C. L-value and R-value. Basic syntax and semantics for expressions and statements.

Unit 4: Control Structures and Functions (8 Lectures)

Control structures in C. Decision making with if, if-else, switch statements. Nested conditions. Looping with while, do-while, and for statement. Break and continue statements. Nested loops. Introduction to functions. Function prototypes and arguments. Defining and calling functions in C. Return values and types. Formal and actual parameter. Call by value, Call by reference. Introduction to recursion. Writing recursive functions in C. Importance of main() function, return type of main() function.

Unit 5: Arrays and Strings (5 Lectures)

Introduction to arrays. Declaration and initialization of arrays. Accessing array elements. Multidimensional arrays. Introduction to strings. Declaration and initialization of strings. String input and output in C. String manipulation functions in C – strlen(), strcpy(), strcat(), strcmp().

Unit 6: Pointers and Memory Allocation (6 Lectures)

Introduction to Pointers. Pointer declaration and initialization. Pointers and addresses. Pointers and arrays. Pointers and functions. Review of call by reference. Pointer arithmetic. Passing an array using pointer in function call. Introduction to dynamic memory allocation. Allocation and deallocation of memory using malloc(), calloc(), and free() functions.

Unit 7: Structure and Union **(4 Lectures)**

Introduction to structures. Declaration and initialization of structures. Accessing structure members. Nested structures and arrays of structures. Unions in C. Declaration and initialization of unions. Accessing union members. Differences between structures and unions. Typedef.

Unit 8: File Handling and Preprocessor Directives **(4 Lectures)**

Introduction to file handling in C. Opening and closing files – fopen(), fclose(). Modes of opening a file. Binary files and text files. Reading and writing files – fgetc(), fgets(), fread(), fputc(), fputs(), fwrite(). File pointers. Error handling in file operations. Preprocessor directives in C - #define, #include, #ifdef, #ifndef, and #endif directives. Using preprocessor directives to define constants and macros. Header files.

List of Practical

(This is a suggestive list only. Questions need not be restricted to this list. The practical are advised to be performed in Linux environment.)

1. Write a program in C to print "Hello World"
2. Write a program to take input of two numbers and print their sum, product, difference.
3. Write a program to find the smallest or greatest of three numbers given as input.
4. Write a program to print the sum and product of digits of an integer.
5. Write a program to print a triangle of stars as follows (take number of lines from user):

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

6. Write a program to reverse a number.
7. Write a program to compute the sum of the first n terms of the following series
   S = 1+1/2+1/3+1/4+……
8. Write a program to compute the sum of the first n terms of the following series
   S =1-2+3-4+5……
9. 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.
10. Write a function to find whether a given no. is prime or not. Use the same to generate the prime numbers less than 100.
11. Write a program to compute the factors of a given number.
12. Write a program to display Fibonacci series (i) using recursion, (ii) using iteration
13. Write a program to calculate Factorial of a number (i) using recursion, (ii) using iteration
14. Write a program in which a function is passed address of two variables and then alter its contents.

15. 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.
16. Write a program to create an array with inputs from the user and print the same.
17. Write a program to perform following actions on an array entered by the user:
    a) Print the even-valued elements
    b) Print the odd-valued elements
    c) Calculate and print the sum and average of the elements of array
    d) Print the maximum and minimum element of array
    e) Remove the duplicates from the array
    f) 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.

18. Write a program to take a matrix from the user and print the transpose of the same.
19. Write a program to take two matrices from the user and find the sum and product of both.
20. Write a program to perform following operations on strings:
    a) Convert all lowercase characters to uppercase
    b) Convert all uppercase characters to lowercase
    c) Calculate number of vowels in the string
    d) Reverse the string
    e) Concatenate two strings without using strcat() function.
    f) Concatenate two strings using strcat() function.
    g) Compare two strings using strcmp()
    h) Copy one string to another using strcpy()
21. Write a program that swaps two numbers using pointers.
22. 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.
23. Write a function to accept two arrays as argument and returns their sum as an array.
24. Write a program to use a macro to swap two numbers.
25. Write a program to implement struct in C. Create a structure of Student with RNo, Name and other credentials with proper datatype and print the same.
26. Write a program to implement union in C. Create a structure of Person with Pid, Name and other credentials with proper datatype and print the same.
27. Write a C program that opens a file for reading and displays the contents of the file in binary mode and text mode.
28. Write a C program that opens a file for reading and displays the contents of the file character by character and line by line on the screen.
29. Write a C program to open a file and count the number of characters and lines in the file.
30. Write a C program that opens a file in append mode and allows the user to add text to the end of the file.

COM020104: COMPUTER ORGANIZATION

1. **Learning Outcome:**
   - Student will able to learn about the structure, function and characteristics of computer systems.
   - Student will understand the design of the various functional units and components of computers.
   - Student will identify the elements of modern instructions sets and their impact on processor design.
   - Student will able to learn about the function of each element of a memory hierarchy.
   - Student will able to learn about identify and compare different methods for computer I/O.
   - Student will able to learn about basics of assembly language.

2. **COURSE OUTCOMES:**

At the end of the course, students will be able to:

CO1: Describe the basic functional block of a computer.
CO2: Discuss different ways to represent data in memory and arithmetic operations performed on the data.
CO3: Explain the basic instruction set architectures of computer.
CO4: Analyze the hierarchy of memory organization and Input-Output systems used in computer.
CO5: Develop different assembly language programs for 8085 microprocessor.

3. **Prerequisite: NIL**
4. **Semester: 2**
5. **Course Type: Compulsory**
6. **Course Level: 100-199**
7. **Theory Credit:** 4
8. **Practical Credit:** 0
9. **Number of required hours:**
   a) Theory: 60 hrs
   b) Practical: 0 hrs
   c) Non Contact: 5 hrs

10. **List of reference books:**

f) M.Morris Mano, *Computer System Architecture*, PHI publication.
g) Hamachar, Vranesic and Zaky, *Computer Architecture*.
h) William Stallings, *Computer Organization and Architecture*; Pearson.
i) Ramesh Gaonkar, *Microprocessor Architecture, Programming, and Applications with the 8085*, 5th Edition.

11. Detailed Syllabus:

UNIT 1: Introduction (4 Lectures)

Definitions of Computer Organization and Architecture, History of computer architecture, Basic functional blocks of a computer: CPU, memory, Input-output subsystems, Control unit, Types of register- general purpose registers, special purpose registers, index registers.

UNIT 2: Data Representation (8 Lectures)

Number system, Complements, Representation of signed numbers, Subtraction of unsigned numbers, Fixed-Point representation- Integer representation, Arithmetic addition, Arithmetic subtraction, Overflow, Decimal Fixed-Point representation, Floating-Point representation, Other Binary Codes- Gray Code etc.

UNIT 3: Register Transfer and Micro-operation (8 Lectures)

Introduction to Register Transfer Language, Register transfer, Bus and Memory transfers, Arithmetic micro-operation- Binary adder, Binary adder-subtractor, Binary incrementer, Arithmetic circuit, Logic micro-operation, Shift micro-operation, Arithmetic logic shift unit.

UNIT4: Processing Unit (10 Lectures)

Instruction codes, Computer registers, General register organization, Register stack, Memory stack, Computer instructions, Data path in a CPU, Operations of a control unit, Hardwired control unit, Micro-programmed control unit, Instruction cycle, Operands, Addressing modes, Instruction format- Three-address instructions, Two-address instructions, One-address instructions, Zero-address instructions, Data transfer and manipulation- Data transfer instructions, Data manipulation instructions, Arithmetic instructions, Logical and Bit manipulation instructions, Shift instructions, Program Control-Status bit conditions, Conditional branch instructions, Subroutine call and return, Instruction execution cycle, CISC and RISC architectures.

UNIT 5: Memory Organization (10 Lectures)

Semiconductor memories, Memory cells - SRAM and DRAM cells, Concept of hierarchical memory organization, Interleaved memories, Cache memory unit - Concept of cache memory, Mapping methods, Organization of a cache memory unit, Cache replacement policies, Write policy, Concept of virtual memory.

UNIT 6: I/O Organization (10 Lectures)

Access of I/O devices, I/O ports, I/O control mechanisms - Program controlled I/O, Interrupt driven I/O, DMA controlled I/O, Interrupts: Types of interrupts, Enabling and disabling interrupts, Handling interrupts.

UNIT 7: Basics of Microprocessor and Assembly Language (10 Lectures)

Introduction to microprocessors, 8085 Microprocessor and its operation, 8085 instruction sets, Addressing modes in 8085, Classifications of instructions and addressing mode, Assembly language programming basics, Assembling, Executing and debugging the programs, Developing counters and Time delay routines, Interfacing concepts.

COM030104: Object Oriented Programming using C++

1. **Learning Outcomes:** After successful completion of this course, students will be able to:
   - Differentiate between Structured programming and Object-Oriented Programming.
   - Learn the concept of objects and develop the ability of imagining real life concepts asobjects and derive their properties and functions to operate these objects.
   - Develop programs using different object- oriented programming features such as data abstraction, polymorphism, inheritance, exception handling etc.

2. **COURSE OUTCOMES:**

   At the end of the course, students will be able to:

   CO1: Define OOP and describe basic data types, different operators, control structures, Function prototyping and call by reference method used in C++.

   CO2: Discuss about the concept of class, object, memory allocation for objects, concepts of constructors and destructors used in C++.

   CO3: Explain the concept of function overloading and operator overloading.

   CO4: Illustrate the concept of inheritance, polymorphism and exception handling mechanism.

   CO5: Design C++ program to illustrate different concepts like class, object, constructor, inheritance, virtual functions used in OOP.

3. **Prerequisites: NIL**
4. **Semester: 3**
5. **Course Type:** Compulsory
6. **Course Level:** 200-299
7. **Theory Credit:** 3
8. **Practical Credit: 1**
9. **No of required hours:**
   a) Theory: 45 hrs
   b) Practical: 30 hrs
   c) Non Contact: 5 hrs
10. **List of Reference Books:**

a) M. T. Somashekara, D. S. Guru et-al; *Object-Oriented Programming with C++, 2nd Edition,* PHI,2012.

b) Bjarne Stroustrup, The *C++ Programming Language, Special Edition,* Pearson Education, 2004.

c) Deitel&Deitel, *C++ How to program,* Pearson Education Asia, 6th Edition, 2008

d) Schildt Herbert, *The Complete Reference C++,* Tata McGraw Hill, 4th Edition, 2003.

11. Contents of Syllabus:

A. Theory

UNIT 1: Introduction to object-oriented programming (3 Lectures)

Basic Concepts of Object-Oriented Programming and design, Benefits and applications of OOP.

UNIT 2: Introduction to C++ (6 lectures)

Structure of a Simple C++ program, Output operator, Input operator, Cascading of I/O operators, Tokens- keyword, identifiers, constants, strings and operators. Basic data types, User defined data types, Dynamic initialization of variables, Reference variables, Operators in C++, Scope resolution operator & applications, Member dereferencing operators, Memory Management operators, new and delete, Control Structures-simple if, if else, nested if, switch, while do, break and continue statements, Introduction to Functions-Function Prototyping, Call-by-reference, Return by reference, Inline functions, Default arguments, Const arguments.

UNIT 3: Classes and objects (11 Lectures)

Introduction - Defining a class; class versus structures, creating objects, accessing class members, defining member functions- outside the class definition and inside the class definition, outside functions as inline. Nesting of member functions, private member functions, memory allocation for objects. Array-declaring an array, accessing elements of an array, array of objects. Friendly functions. Basic Concepts of constructors and destructors with examples. Default constructor, Parameterized constructor, Multiple constructors in a class. Constructor with default arguments, Copy constructor. Dynamic initialization of objects. Dynamic constructors and destructors.

UNIT 4: Function and operator overloading (10 Lectures)

Concept of Overloading. Function Overloading: Functions with different sets of parameters,default and constant parameters, Rules for overloading operators, definingoperator overloading.Overloading unary operators -prefix and postfix operators. Overloading Binary operators and relational operators, Overloading using friend functions.

UNIT 5: Inheritance (12 Lectures)

Concept of Inheritance -defining derived classes. Types of inheritances, Making a private member inheritable, multilevel inheritance, multiple inheritance, Hierarchical inheritance, Hybrid inheritance, Virtual base classes, Abstract classes, Constructors in derived classes, nesting of classes, polymorphism-Compile time and Runtime polymorphism, Pointers to objects, “this” pointer, Pointer to derived classes, Virtual functions, Rules for virtual functions, Pure virtual functions.

UNIT 6: Exception Handling (3 lectures)

Examples of exceptions and handling exceptions using try, catch and throw statements.

B. Practicals

Following Practical / Lab works to be performed preferably in Linux Environment

1. Define a class named “triangle” to represent a triangle using the lengths of the three sides.Write a constructor to initialize objects of this class, given the lengths of the sides. Also writemember functions to check

(a) if a triangle is isosceles

(b) if a triangle is equilateral

Write a main function to test your functions.

2. Define a structure “employee” with the following specifications.

empno : integer

ename : 20 characters

basic, hra, da : float

calculate() : a function to compute net pay as basic+hra+da with float return type.

getdata() : a function to read values for *empno*, ename, basic, hra, da.

dispdata() : a function to display all the data on the screen

Write a main program to test the program.

3. Define a class “circle” to represent circles. Add a data member radius to store the radius of acircle. Write member functions *area*() and *perimeter*() to compute the area and perimeter of acircle.

4. Define a class “complex” with two data members “real” and “imag” to represent real and imaginaryparts of a complex number. Write member functions

rpart() : to return the real part of a complex number

ipart() : to return the imaginary part of a complex number

add() : to add two complex numbers.

mul() : to multiply two complex numbers.

Write constructors with zero, one and two arguments to initialize objects.

5. Define a class “point” with two data members “*xordinate*” and “*yordinate*” to represent all pointsin the two-dimensional plane by storing their x co-ordinate and y co-ordinate values. Writemember functions

dist() : to return the distance of the point from the origin.

slope(): to return the slope of the line obtained by joining this point with the origin.

Write constructors with zero, one and two arguments to initialize objects. Also write a friendfunction to compute the distance between two points.

6. Define a class “string” with the following data members char *p; int size; and write memberfunctions to do the following (without using library function) and using dynamic memoryallocation.

- Length of the string
- Compare two strings
- Copy one string to another
- Reverse the string

Write suitable constructors and destructors. Also write a copy constructor for the class.

7. For the class “complex” defined in 4 above, overload the <<, >>, + and * operators in the usualsense. Also overload the unary – operator.

8. Define a class “time” to store time as hour, minute and second, all being integer values. Writemember functions to display time in standard formats. Also overload the ++ and – operatorsto increase and decrease a given time by one second where the minute and hour values willhave to be updated whenever necessary.

9. Define a class to store matrices. Write suitable friend functions to add and multiply two matrices.

10. Write a class-based program implementing static members.

11. Define a class student with the following specification:

rollno : integer sname : 20 characters

Derive two classes *artst* and *scst*. The class artst will represent students belonging to arts streamand the class scst will represent students belonging to science stream. The *artst* class will haveadditional data members ph, hs, en and as to store marks obtained by a student in three subjectsPhilosophy, History, English and Assamese. The class scst will have additional data members*ph*, *ch*, *ma* and *en* to store marks obtained in *Physics*, *Chemistry*, *Mathematics* and *English*.

Write the following member functions in the classes artst and scst;ctotal() : a function tocalculate the total marks obtained by a student;*takedata*() : a function to accept values of the datamembers andshowdata() : a function to display the marks sheet of a student .

12. Define an abstract base class printer. Derive three classes laser-printer, line-printer and inkjet-printer. The derived classes will have data members to store the features of that particularprinter. Write pure virtual function display() in the base class and redefine it in the derivedclasses.

13. Define a abstract base class figure and add to it pure virtual functions

display() : to display a figure
get() : to input parameters of the figure
area() : to compute the area of a figure
perimeter() : to compute the perimeter of a figure.

Derive three classes circle, rectangle and triangle from it. A circle is to be represented by itsradius, rectangle by its length and breadth and triangle by the lengths of its sides. Write amain function and write necessary statements to achieve run time polymorphism.

14. Write an interactive program to compute square root of a number. The input value must betested for validity. If it is negative, the user defined function *my_sqrt*() should raise anexception.

COM040104: Data Structure

1. Learning Outcomes: At the end of the course, students will be able to:

- Understand and apply the fundamental data structures and algorithms – such as arrays, linked lists, stacks, queues, trees, sorting and searching algorithms using C programming language.
- Analyze the time and space complexity of different algorithms and choose the appropriate algorithm for a given problem.
- Develop efficient algorithms to solve various computational problems by utilizing data structures and algorithms covered in the course.

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO1: Distinguish between linear and non linear data structures.

CO2: Explain different operations performed on array, link list, stack, queue and binary tree.

CO2: Discuss different operations performed on binary tree, representation of binary tree using array and link list..

CO3: Explain about different sorting and searching algorithms applied on data.

CO4: Analyze the time complexity and space complexity of algorithm.

CO5: Design C++ program to show the operations performed on different data structure.

3. **Prerequisites**: NIL

4. Semester: 4

5. Course Type: Elective

6. Course Level: 200-299

7. Theory Credit: 3

8. Practical Credit: 1

9. No of required hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

10. List of Reference Books:

e) Weiss, Mark Allen. “Data Structures and Algorithm Analysis in C”. 3rd ed., Pearson, 2012

f) Sedgewick, Robert. “Algorithms in C, Parts 1-5 (Bundle): Fundamentals, Data Structures, Sorting, Searching, and Graph Algorithms”. 3rd ed., Addison-Wesley Professional, 2002.

g) Goodrich, Michael T., and Roberto Tamassia. “Data Structures and Algorithms in C”. 2nd ed., Wiley, 2011.

h) Gilberg, Richard F., and Behrouz A. Forouzan. “Data Structures: A Pseudocode Approach with C”. Narosa Publishing House, 2009.

11. Contents of Syllabus:

A. Theory

Unit 1: Data Structures Overview and Arrays (8 Lectures)

Concepts of Data Types, Abstract Data Type, Data Structure, Fundamental and Derived Data Types. Importance of data structures. Array as a data structure (characteristics, advantages, disadvantages). Representation of arrays – single and multidimensional. Address calculation of array element using column and row major ordering. Address translation functions for one & two dimensional arrays. Insertion and deletion in arrays. Use of arrays for large number representation.

Unit 2: Linked Lists (9 Lectures)

Initialization and implementation of structures. Structure and pointers. Self referential structure. Introduction to linked lists. Singly linked list, doubly linked list, circular linked list. Operations on lists – creation, insertion, deletion, traversal, merging and splitting.

Unit 3: Stacks and Queues (9 Lectures)

Definition of Stack and Queue. Representation of stacks and queues using arrays and linked lists. Stack operations – push, pop. Queue operation – enqueue, dequeue. Circular Queue, Priority Queue, Conversion of infix arithmetic expression containing arithmetic operators and parenthesis to postfix and prefix expression. Evaluation of postfix expression.

Unit 4: Binary Trees (8 Lectures)

Definition of Trees – General tree and Binary tree. Basic terminologies – parent, child, height, depth, leaf, node, internal nodes, external nodes. Brief concept of Forest, ordered trees, strictly binary tree, complete binary tree. Representation of trees using arrays and linked lists. Binary tree traversal methods – pre-order, in-order, post-order. Recursive and non-recursive algorithms for traversal methods. Binary search trees. Operation on BST – creation, insertion and deletion of a node. Definition and characteristics of threaded binary trees. Min heap and Max heap.

Unit 5: Searching and Sorting (6 Lectures)

Linear and binary search. Indexed search. Hashing. Hash Functions – division method, mid square method, folding. Conflict resolution – linear and quadratic probe. Sorting algorithms – Insertion sort, Selection sort, Bubble sort, Merge sort, Quick sort, Counting sort, Heap sort. In-place sorting and stable sorting.

Unit 6: Analysis of Algorithm and Complexity (5 Lectures)

Complexity measures of an algorithm – Time and space complexity. Average case and worst case analysis. Asymptotic notation as a measure of algorithm complexity, O and θ notations. Analysis of sorting algorithms and Searching algorithms in terms of time and space complexity in best, average and worst case.

Time and Space complexity of algorithms, average case and worst case analysis, asymptotic notation as a measure of algorithm complexity, Θ and O notation. Analysis of sorting algorithms-

Selection sort, Bubble sort, Insertion sort, Heap sort, Quick sort and analysis of searching algorithms – linear search and binary search.

List of Practical

(This is a suggestive list only. Questions need not be restricted to this list. The practical are advised to be performed in Linux environment using C programming language.)

31. Write a program to declare an array and initialize the values according to the user. Now ask the user for a number *n* and return the n^{th} element from the array.
32. Write a program to implement array initialized with the numbers divisible by three up to 30. Write a function which accepts the array and return the positions of the even numbers in the array.
33. Implement linked list in a program by writing functions for the following:
    a. Create a singly linked list of *n* nodes
    b. Count the number of nodes in the list
    c. Print the values of all the nodes
    d. Add a node at first, last and k^{th} position in the linked list
    e. Delete a node from first, last and k^{th} position
    f. Search for an element in the list. If found, return the position of the node. If not found, return a negative value.
34. Write a program to implement doubly linked list.
35. Write a function to concatenate two linked lists.
36. Write a program to take a number *k* and split the linked list after k^{th} position.
37. Write a program to merge two sorted linked lists.
38. Write a program to implement list of lists.
39. Write a program to implement stack using array. Use push and pop operations on the array representation of the stack. Check whether the stack is full or empty.
40. Write a program to implement stack using linked list. Use push and pop operations on the stack by inserting nodes and deleting nodes from the linked list. Also check if the stack is full or empty.
41. Write a program to evaluate a simple postfix expression using stack.
42. Write a program to convert a decimal number into binary number using stack.
43. Write a program to implement queue using array. Add new elements to the queue and remove elements from the queue represented by array. Check whether the queue is full or empty.
44. Write a program to implement queue using linked list. Add new elements to the queue and remove elements from the queue represented by linked list. Also check whether the queue is full or empty.
45. Implement binary search and linear search algorithms on arrays.
46. Implement binary search tree using array by writing a program to:
    a. Create a binary search tree using array
    b. Print the prefix notation of the BST
    c. Print the infix notation of the BST

d. Print the postfix notation of the BST
e. Search for an element in the BST

47. Implement binary search tree using linked list by writing a program to:
    a. Create a binary search tree using linked list
    b. Print the prefix notation of the BST
    c. Print the infix notation of the BST
    d. Print the postfix notation of the BST
    e. Search for an element in the BST
48. Implement following sorting algorithms:
    Bubble sort, Insertion sort ,Selection sort, Counting sort

COM040204: Database Management System

1. **Learning Outcome:** On successful completion of this course, the student should be able to:
    - Learn database concepts and its architectural components.
    - Describe different data models used for designing a database.
    - To create a database using relational models and entity relationships concepts
    - Normalize a database into various normal forms
    - Design SQL queries to handle a relational database.

2. **COURSE OUTCOMES:**

    At the end of the course, students will be able to:

    CO 1: Describe database management system architectures, its types, advantages and disadvantages.
    CO 2: Apply Entity Relationship modeling to real world problems and mapping ER to relational model.
    CO 3: Interpret functional dependencies and categorize a table to specific normalization.
    CO 4: Analyze various concurrency problems that may happen in database design process.
    CO 5: Construct complex SQL queries to fetch, delete, update, and create data in DBMS software.

3. **Prerequisite:** NIL
4. **Semester: 4**
5. **Course Type: Compulsory**
6. **Course Level: 200-299**
7. **Theory Credit:** 3
8. **Practical Credit:** 1
9. **Number of required hours:**
    a) Theory: 45 hrs
    b) Practical: 30 hrs
    c) Non Contact: 5 hrs

10. **List of reference books:**
    a) Dr. Satinder Bal Gupta and Aditya Mittal, *Introduction to Database Management System*¸University Science Press
    b) A. Silberschatz, H.F. Korth, S. Sudarshan, *Database System Concepts*, McGraw Hill
    c) R. Elmasri, S.B. Navathe, *Fundamentals of Database Systems*, Pearson Education
    d) Dr. Rajive Chopra, *Database Management System (DBMS): A Practical Approach*, S. Chand Publication

11. Detailed Syllabus:

UNIT-1: **Introduction to Database Management Systems (5 Lectures)**

Basic Definition and Concepts: *Data, Information, Meta Data, Data Dictionary, Database, Fields, Records* and *Files*. Definition of Database Management System (DBMS), Primary Functions of DBMS, Traditional File approach, Traditional file approach versus database management system approach, Disadvantages of Traditional File System, Need of a DBMS, Components of a DBMS, Advantages of DBMS, Disadvantages of Database Systems, Various uses of database System Applications, Database Users: *End users or naive users*, *Onlineusers*, *ApplicationProgrammers*, *DatabaseAdministrator*(*DBA*), Responsibilities of DBA.

UNIT 2: Database Management System Architecture (6 Lectures)

Definition of *Schemas*, *sub-schema* and *Instances*. Data Independence: *Physical Data Independence* and *Logical data Independence*. Three-tier architecture of DBMS, Advantages of three-level Architecture, basic concept of data model, Characteristics of Data Models, Types of Data models: *Record Based Data Models*, *Object Based Data Model* and *Physical Data Models*. Relational Data Model, Types of database Systems: *Single-user* database systems, *Multiuser* database systems, *Centralized* database systems, *Distributed* database systems and *Client/Server* database systems.

UNIT 3: E-R Modeling (8 Lectures)

Basic Concepts: *Entity, Attributes, Entity Sets, Domain*. Types of attributes: *Simple and Composite Attributes*, *Single Valued and Multi-valued Attributes*, *Derived Attributes and Stored Attributes*. Types Of Entity Sets*: Strong Entity Sets* and *Weak Entity Sets.* Concept of Relationship and Relationship sets, Types of Relationship: One-to-One, One-to-Many, Many-to-One and Many-to Many, Various Symbols used in ER Diagram, Mapping constraints: *MappingCardinalities(Cardinality Ratios)* and *ParticipationConstraints*. Definition of Key, Types of Keys: *SuperKey, Candidate Key, Primary Key, Alternate Key* and *Foreign Key*. Symbols used in E-R diagrams, Conversion of an ER and Diagram in to Relational Tables

UNIT4: Relational Model and Relational Algebra (7 Lectures)

Definition of Relation, Data Structure of Relational Database: *Relation, Tuples, Attributes Domain, Degree* and *Cardinality.*Integrity Constraints, Domain Constraints, Key Constraints, Advantages and Disadvantages of Relational Model, Relational, Definition of Relational algebra, Operations in Relational Algebra: *Selection, Projection, Division, Rename, Union, Intersection, Set Difference, Natural-join operation, Outer join, Inner Join, Cartesian Product* and *Assignment operation*. Aggregate Functions and Operations: *Average, Maximum, Minimum, Sum* and *Count*.

UNIT 5: Functional Dependency and Normalization(8 Lectures)

Definition of Functional Dependency, Armstrong's Axioms in Functional Dependency, Types of Functional Dependency: *Partial Dependency, Full Functional Dependency, Transitive and Non-transitive Functional Dependency*,Armstrong's Axiom, Closure of a set of Functional Dependency, Closure of an Attribute,Definition of Canonical Cover, Algorithmto find the

canonical cover of a FD set, Anomalies in relational database*: Insertion*, *Deletion* and *Update* anomalies, Concepts of Normalization, Benefits of Normalization, Types of Normal Forms: First Normal Form (1NF), Second Normal Form (2NF), Third Normal Form (3NF) and Boyce–Codd Normal Form (BCNF)

UNIT 6: Transaction and Concurrency Control (4 Lectures)

Definition of Transaction, ACID Properties of transaction, Transaction States, Definition of Concurrency Control, Need of Concurrency Control, The Lost Update Problem, The Uncommitted Dependency Problem, The Inconsistent Analysis Problem, Serializability: *View Serializability* and *Conflict Serializability*

UNIT 7: SQL Queries (7 Lectures)

Database Languages (Data Definition Languages, Data Manipulation Languages), Characteristics of SQL, Basic data types in SQL, Data-definition language (DDL) commands: *Create Database*, *Create Table*, *Drop Table*, *Alter Table*. SQL Constraints: *Primary Key*, *Foreign Key, Not Null, Unique, Check, Defaul,*.Data Manipulation Language (DML) commands: *Insert Into*, Delete, *Select*, *Update*. SQL clauses: *Where, Order By, Having, Group By* and *Like*. SQL join operations: *Inner Join, Left Outer Join, Right Outer Join* and *Full Join*. SQL aggregate functions: *sum(), count(), max(), min()* and *avg()*

Lab Contents:

Practical / Lab work to be performed:

- Implementation of SQL DDL statements in MySQL DBMS: CREATE DATABASE, CREATE TABLE, ALTER TABLE, RENAME, DROP DATABASE/TABLE
- Use of SQL DML statements in MySQL DBMS: INSERT, SELECT, UPDATE, DELETE SQL commands
- Implementing following constraints in MySQL DBMS: PRIMARY KEY, FOREIGN KEY, NOT NULL, UNIQUE and DEFAULT
- Handling following SQL clauses in MySQL DBMS: WHERE, GROUP BY, ORDER BY, HAVING, IN, BETWEEN, LIKE
- Working with following aggregate functions in MySQL DBMS: COUNT, AVG, MAX, MIN and SUM
- Working with transaction processing command in MySQL DBMS: START TRANSACTION, COMMIT and ROLLBACK Statements, SET auto commit

COM040304: Mathematical Foundation of Computer Science

1. Learning Outcome: After successful completion of this course, students will be able to:

- Learn the concepts of set, relation, and function from Computer Science point of view.
- Understand the basic idea of counting and use it in counting under various constraints.
- Understand graphs and its different representations in Computers. How to model real life problems using graphs. Learn a few basic graph traversal algorithms.
- Understand Mathematical Logic from algorithmic point of view.

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO 1: Define and describe sets, relations, functions, its properties and operations.
CO 2: Associate with the principles of counting, inclusion, induction and its applications.
CO 3: Analyze the growth of functions and discuss asymptotic notations.
CO 4: Distinguish and create different graphs, discuss its properties and applications.
CO 5: Solve problems that are revolving around mathematical logic e.g. connectives, truth tables, tautologies and contradictions.

3. Prerequisites: Nil
4. Semester: 4
5. Course Type: Elective
6. Course Level: 200-299
7. Theory Credit: 4
8. Practical Credit: 0
9. No of Hours:

a) Theory: 60 hrs
b) Practical: 0 hrs
c) Non Contact: 5 hrs

10. List of Books:

a) *Elements of Discrete mathematics*, C.L. Liu , D.P. Mahopatra; 2nd Edition , Tata McGraw Hill, 1985,
b) **Discrete Mathematics and Its Applications**, Kenneth Rosen, Sixth Edition ,McGraw Hill 2006.
c) *Introduction to Algorithms*, T.H. Coremen, C.E. Leiserson, R. L. Rivest; 3rd edition Prentice Hall of India, 2009.
d) *Discrete Mathematics and Graph Theory*; Grimaldi, 5th Edition; 2019, Pearson.

11. Contents of Syllabus:

A. Theory

UNIT 1: (16 Lectures)

Sets, Relations and Functions

Sets: definition of set, cardinality of sets, finite, countable and infinite sets. Operations on sets, Venn diagram. Principle of inclusion and exclusion and their applications on simple problems. Multisets.

Relations: Definition and properties of binary relations, closures of relations, equivalence relations, equivalence classes and partitions, n-ary relations and representation of n-ary relations as tables. Partial ordering relations and lattices,

Functions: Definition of function**,** one-to-one and onto, principles of mathematical induction. Concave and convex functions.

UNIT 2: Combinatorics (15lectures)

Basic of counting principles, principle of inclusion-exclusion, application of inclusion and exclusion, Mathematical Induction. Pigeonhole principle, generalized Pigeonhole principle and its application, permutations and combinations, circular permutations, permutations with repetitions, combinations with repetitions, permutations of sets with indistinguishable objects

UNIT 3: Growth of Functions (5 Lectures)

Asymptotic behavior of functions, Asymptotic Notations - Big-O and Theta. Summation formulas and properties, Bounding Summations.

UNIT 4: Graph Theory (12 Lectures)

Basic Definition of graph, Directed, Undirected and Weighted Graphs. Representation of graphs in Computers – Adjacency Matrix and Adjacency Lists. Degree of vertices – indegree and outdegree. Paths, Cycles and Acyclic graphs. Simple operations on graphs and amount of computations required for each operation. Connected graph, Tree and Forest. Bipartite graph, Algorithms on graph traversals- Breadth first search, Depth first search.

UNIT 5: Mathematical Logic (12 Lectures)

Connectives, truth tables, Tautologies and Contradictions, Equivalence and Implications, NAND and NOR, Normal forms- CNF, DNF, Converting expressions to CNF and DNF, Theory of inference, Propositional Calculus, Predicate calculus (only introduction), predicates and quantifiers.

COM040404: Operating System

1. **Learning Outcomes**: After successful completion of this course, students will be able to:
   - Learning Outcomes: After completing this course, students will have understanding of theinternal structure and usage of various components related to an operating system.

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO1: Identify different operating systems and its types, functions, applications.
CO2: Explain process, threads and solve process scheduling problems.
CO3: Compare solutions of process synchronizations methods.
CO4: Discuss process deadlock handling techniques.
CO5: Solve memory management issues and discuss memory management solutions.

3. **Prerequisites**: NIL
4. Semester: 4
5. Course Type: Elective
6. Course Level: 200-299
7. Theory Credit: 3
9. Practical Credit: 1
10. No of required hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

11. List of Reference Books:

i) Operating System Concepts, Abraham Silberschatz, Peter B. Galvin, Greg Gagne,Wiley
j) Modern Operating Systems, Andrew S. Tanenbaum, Prentice-Hall Of India Pvt.Limited

12. Contents of Syllabus:

A. Theory

Unit I: Introduction (7 hrs)

Application vs system software, operating system as system software, operating structure structure, types of operating systems: batch operating system, multiprogramming operating system, multitasking operating system, distributed operating system, real time operating system, multi useroperating system, major functions of operating system: Process Management, ProcessSynchronization, Memory Management, CPU Scheduling, File Management, I/O Management,Security, virtualization, cloud computing, open source operating system, history of operatingsystem, the shell, system call, system boot

Unit II: Process and threads (10 hrs)

Process, process states: new, running, waiting, ready and terminated, Process Control Block (PCB),information stored in PCB, scheduling queue: job queue, ready queue and device

queue,schedulers: long term schedulers, medium term scheduler and long term scheduler, swapping,degree of multiprogramming, I/O-bound and CPU-bound processes, context switching, inter-process communication: shared memory systems and message passing systems, socket, remoteprocedure call, threads, user threads, kernel threads, multi threading models: Many-to-One Model,One-to-One Model, Many-to-Many Model, CPU scheduling, Scheduling Criteria, schedulingalgorithms: First-Come, First-Served Scheduling, Shortest-Job-First Scheduling, PriorityScheduling, Round-Robin Scheduling, Multilevel Queue Scheduling, Multilevel Feedback QueueScheduling

Unit III: Process synchronization (8 hrs)

Race condition, critical section problem, Peterson's algorithm, Bakery algorithm, synchronizationhardware: locking, synchronization software tools: mutex lock, semaphore (counting and binary),semaphore implementation, classic synchronization problems: bounded buffer problem, the readers–writers Problem, the dining-philosophers problem, monitor, synchronization in windows,synchronization in Linux

Unit IV: Deadlock (10 hrs)

Deadlock, operations of a process performs while using a resource: Request. Use and Release,physical and logical resources, Necessary conditions: mutual exclusion, hold & wait, nopreemption and circular wait, resource allocation graph, deadlock prevention: definition,preventing mutual exclusion, preventing hold & wait, preventing no preemption and preventingcircular wait, deadlock avoidance: definition, safe state, safe sequence, resource allocation graphbased algorithm and Banker's algorithm, deadlock detection: definition, wait-for graph, algorithmto detect deadlock for single instance resources, algorithm to detect deadlock for multiple instanceresources and recovery from deadlock: process termination and resource preemption

Unit V: Memory Management (10 hrs)

Memory hierarchy, base register, limit register, address binding, logical and physical addressspaces, memory management unit, relocation register, swapping, contiguous memory allocation:definition, memory protection, fixed partition scheme, variable partition scheme, first-fit, best-fit &worst-fit allocation strategies, non-contiguous memory allocation: simple paging and simplesegmentation, internal and external fragmentation, TLB, virtual memory, demand paging, pagefault, locality of reference principle, performance of demand paging, page replacement algorithms:FIFO, Optimal and LRU, allocation of frames: equal allocation and proportional allocation, globaland local page replacement algorithms, thrashing

Practicals:

3. Basic linux commands: pwd, ls, cd, mkdir, rmdir, rm, touch, man, cp, mv, locate, head, tailAdvanced commands: echo, cat, sudo, df, tar, apt-get, chmod, hostname, useradd, passwd, groupadd, grep, sed, uniq, wc, od, gzip, gunzip, find, date, cal, clear, top, ps, kill
4. Shell scripting in linux: shell, types of shell, shell script, echo command, shell variables,

5. special variables ($$, $0, $n, $#, $?, $!), array, assignment operator (=), equality operator (==), not equality operator (!=), arithmetic operators (+,-, *, /, %), comparison operators (-eq, -neq, -gt, -lt, - ge, -le), logical operators (!,-o, -a), if...else statement, case...esac statement, while loop, for loop,break statement, continue statement, shell functions 7 classes
6. Using system calls in C program in linux: fork(), exec(), exit(), getpid(), mkdir(), rmdir() etc.

COM050104: Computer Networks

1. Learning Outcome: After completing this course, students

- Student will able to learn about the general principles of data communication.
- Student will able to learn about how computer networks are organized with the concept of layered approach.
- Student will able to learn about how signals are used to transfer data between nodes.
- Student will able to learn about how packets in the Internet are delivered.
- Student will able to learn about how routing protocols work.
- Student will able to learn about functions of transport layer
- Student will able to learn about functions of application layer

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO1: Define internet and describe different network topologies and types of networks.
CO2: Discuss about different transmission mediums, encoding methods, transmissions modes and switching techniques.
CO3: Explain the concept of framing, error control, stop and wait protocols sliding window protocols, protocols of MAC sublayer.
CO4: Illustrate the concept of IP address, three way handshaking and DNS.
CO5: Compare different routing algorithms used by network layer.

3. Prerequisites: NIL
4. Semester: 5
5. Course Type: Elective
6. Course Level: 300-399
7. Theory Credit: 3
8. Practical Credit: 1
9. No of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

10. List of Books:

a) B. A. Forouzan: *Data Communications and Networking*, Fourth edition, THM, 2007.
b) A. S. Tanenbaum: *Computer Networks***,** Fourth edition**,**PHI , 2002.

11. Contents of Syllabus:
A. Theory
Detailed Syllabus:
UNIT 1: Introduction to Computer Networks (5 Lectures)

Data communication system and its components, Definition of network, Types of network, Network topologies, Network protocol, Layered network architecture, Overview of OSI reference model, Overview of TCP/IP protocol suite.

UNIT 2: Physical Layer Communication (10 Lectures)

Analog and digital signal, Definition of bandwidth, Maximum data rate of a channel, Line encoding schemes, Transmission modes, Modulation techniques, Multiplexing techniques- FDM and TDM, Transmission media-Guided and Unguided, Switching techniques- Circuit switching, Packet switching, Connectionless datagram switching, Connection-oriented virtual circuit switching.

UNIT 3: Data Link Layer Functions and Protocol (10 Lectures)

Definition of Framing, Framing methods, Error detection techniques, Error correction techniques, Flow control mechanisms- Simplex protocol, Stop and Wait ARQ, Go-Back-N ARQ, Point to Point protocol.

UNIT 4: Multiple Access Protocol and Networks(5 Lectures)

Basics of ALOHA protocols, Basics of CSMA/CD protocols, Ethernet LANS, Connecting LAN and back-bone networks- Repeaters, Hubs, Switches, Bridges, Router and Gateways

UNIT 5: Networks Layer Functions and Protocols (8 Lectures)

Connection oriented vs Connectionless services, Definition of Routing, Routing algorithms, IP protocol, IP addresses, ARP, RARP

UNIT 6: Transport Layer Functions and Protocols (4 Lectures)

Transport services, TCP vs UDP protocol, TCP connection establishment- Three way handshakes, TCP connection release

UNIT 7: Overview of Application Layer Protocols (3 Lectures)

Overview of DNS, Overview of WWW, URL, Email architecture, HTTP protocol

B. Practical / Lab work to be performed (15 Practical Classes)

- Implement the data link layer farming methods such as Bit Stuffing.
- Study of different types of Network cables.
- Study of network IP.
- Connect the computers in Local Area Network.
- Study of basic network command and Network configuration commands.
- Socket programming in C language.
- Configure a Network topology using packet tracer software.
- Simulate Cyclic Redundancy Check (CRC) error detection algorithm for noisy channel.
- Simulate and implement Stop and Wait protocol for noisy channel.
- Simulate and implement Go-Back-N sliding window protocol.
- Simulate and implement Selective Repeat sliding window protocol.
- Simulate and implement Dijkstra Algorithm for shortest path routing.
- Simulate and implement Distance vector routing algorithm

COM050204: Java Programming

1. **Learning Outcome:**

After completing this course, students will be

- Familiar with the core concepts of java programming and classes of swing package.

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO 1: Describe java programming constructs and operate different java IDEs.
CO 2: Explain working principle of various building blocks of java platform.
CO 3: Apply object oriented approach using java language to solve required programming problems.
CO 4: Differentiate different java string, array methods and its usage.
CO 5: Design and develop GUI applications using advanced java packages.

3. Prerequisites: NIL
4. Semester: 5
5. Course Type: Elective
6. Course Level: 300-399
7. Theory Credit: 3
8. Practical Credit: 1
9. No of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

10. List of Books:

a) *Java: The Complete Reference*, Herbert Schildt, McGrawHill
b) *Java How to Program*, Paul Deitel, Harvey Deitel, Pearson

11. Contents of Syllabus:
A. Theory
Detailed Syllabus:

Unit I: Introduction (3 hrs)

High level language, compiled and interpreted languages, history of java programming language, compilation of java code, bytecode, java interpreter, javac and java command, path environmental variable, Java IDE, features of java programming language: simple, object oriented, robust, architecture neutral and interpreted

Unit II: Data types, operators and control statements (12 hrs)

Java as strongly typed language, primitive data types, integer data types: byte, short, int and long, floating point data types: float and double, character data type, boolean data type, literals: integer literals, floating-point literals, boolean literals, character literals and string literals, declaring a variable, dynamic Initialization, the scope and lifetime of variables, type-casting in java, one dimensional array, multi dimensional array, arithmetic operators: the basic arithmetic operators, the modulus operator, arithmetic compound assignment operators, increment operator and

decrement operator, bitwise operators, relational operators, short circuit logical operator, the assignment operator, branching statements: if-else and switch-case statements, looping statements: while, do-while, for and for-each statements, jump statements: break and continue

Unit III: Object oriented features of java (10 hrs)

Defining a class, member variable and member methods, access specifiers: default, private and public, declaring objects, assigning object reference variables, constructors, parameterized constructors, the this keyword, garbage collection, the finalize( ) method, overloading methods, overloading constructor, static keyword, final keyword, command line arguments in java, inheritance, super class and sub class, protected access specifier, super keyword, constructor call in multilevel inheritance, method overriding, dynamic method dispatch, abstract class, interfaces, type wrappers

Unit IV: String handling and packages (5 hrs)

String class, String constructors, String length, special string operations: string literals, string concatenation, string concatenation with other data types, string conversion and toString( ), character extraction: charAt( ), getChars( ), string Comparison: equals( ) and equalsIgnoreCase(), regionMatches( ), startsWith( ) and endsWith( ), equals( ) Versus ==, compareTo( ), searching strings, data conversion using valueOf( ), StringBuffer, StringBuffer constructors, length( ) and capacity( ), ensureCapacity( ), setLength( ), charAt( ) and setCharAt( ), getChars( ), package, defining a package, CLASSPATH, importing packages

Unit V: Exception handling and I/O (5 hrs)

Exception-handling, exception types, uncaught exceptions, try and catch block, multiple catch blocks, nested try statements, throw, throws, finally, java's built-in exceptions, creating own exception classes, java I/O classes, reading console input, writing console output, reading and writing files

Unit VI: Swing package and database connectivity (10 hrs)

Swing package, simple GUI-Based Input/Output with JoptionPane, Jframe, JLabel, JTextField, Jbutton, handling event in a Jframe object, layout managers: BorderLayout, FlowLayout, GridLayout, CardLayout, GridBagLayout, JtoggleButton, JCheckBox, JRadioButton, Jlist, JcomboBox, JDBC, JDBC driver, connectivity steps, connectivity with MySQL, DriverManager class, Connection class, Statement class, ResultSet class, PreparedStatement class

(b) Practical

- Java programs to demonstrate the use of data types and operators
- Java input through Scanner class and JOptionPane class
- Java programs to demonstrate the use of control statements.
- Java programs to demonstrate the use of classes, objects, visibility modes, constructors and destructor.
- Java programs to demonstrate the use of inheritance and polymorphism.
- Java programs to demonstrate the use of polymorphism.
- Java programs to handle strings,Java programs implementing exception handling.

- Demonstrating the use and creation of packages in java.
- Java program with JFrame, JTextfield and JButton with event handling
- Using JLabel, JTextArea and JPasswordField in java with event handling
- Working with layout managers in JFrame
- Using JCheckBox, JRadioButton and JComboBox in a JFrame
- Connecting JFrame components to a DBMS

COM050304: Python Programming

1. Learning Outcome:

After completing this course, students

- Know about fundamentals of Python Programming and Problem Solving.

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO1: Identify different keywords, operators and input/output formatting in Python.
CO2: Explain control statements and functions in Python
CO3: Apply object-oriented programming principles in Python.
CO4: Analyze the use cases of Python libraries and their significance in various applications.
CO5: Assess the robustness of Python code by applying exception handling and understanding the implications of database operations.

3. Prerequisites: NIL
4. Semester: 5
5. Course Type: Elective
6. Course Level: 300-399
7. Theory Credit: 3
8. Practical Credit: 1
9. No of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

10. List of Books:

c) *Core Python Programming*, R. Nageswara Rao, Dreamtech Press.
d) *Python: The Complete Reference*, Martin C. Brown, McGraw Hill Education.
e) http://docs.python.org/3/tutorial/index.html

11. Contents of Syllabus:

A. Theory

Detailed Syllabus:

Unit 1: Introduction to Python Programming (7 hrs)

Introduction, Installation of Python Interpreter, Python Shell, Code Indentation, Identifiers and Keywords, Literals, Strings, Operators (Arithmetic, Relational, Logical, Assignment, Ternary, Bitwise, Increment and Decrement Operators), Input and output statements, Output Formatting.

Unit 2: Control Statements and Functions (7 hrs)

Branching, Looping, Conditional Statement, Exit Functions, Break, Continue, Pass, Defining Functions, Default Arguments. Scope of Functions, Function Documentation, Lambda Functions & Map.

Unit 3: Python Data Structures **(6 hrs)**

List (List, Nested List, List as Matrix), Tuple, Set, Dictionary.

Unit 4: Exception Handing **(4 hrs)**

Errors, Exception Handling with try, Multiple Exception Handling, Writing own Exception.

Unit 5: File Handling **(6 hrs)**

Understanding read function, read(), readline() and readlines(), Understanding write functions, write() and writelines(), Programming using file operations, Reading config files, Writing log files in python.

Unit 6: OOP in Python **(3 hrs)**

Creating Classes in Python, Instance Methods, Inheritance, Polymorphism, Exception Classes and Custom Exceptions.

Unit 7: Introduction to Libraries in Python **(5 hrs)**

NumPy, Matplotlib, OpenCV, Tkinter.

Unit 8: Python SQL Database Access **(7 hrs)**

Introduction to database driven program, Database Connection, Database Operations: INSERT, READ, UPDATE, DELETE, COMMIT AND ROLLBACK.

(b) Practical

- Introduction to Python console, operators, input and output statements.
- Python control statements and functions
- Data Structures in python
- Exception Handling
- File Handling
- Object Oriented Python programming
- Introduction to libraries (NumPy, Matplotlib, OpenCV)
- Python SQL Database Connection and database operations

COM050404: Software Engineering

1. Learning Outcome: On successful completion of this course, the student should be able to:

- Determine the primary problems that impact all software development processes.
- Choose relevant software development processes models, methodologies, and strategies for managing a specific software development process, and justify the choices
- Implement different software estimation metrics such as cost, effort size, staffing etc.
- Describe various software design approaches and various coding and testing strategies used in software engineering principles
- Know about software reliability and how to calculate software maintenance cost

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO1: Identify different software development processes and their challenges.
CO2: Explain software requires specification and translate it into an implementable design, following a structured and organized process.
CO3: Apply different software estimation metrics such as cost, effort size, staffing etc.
CO4: Design effective use of UML, along with design strategies such as defining software architecture, separation of concerns and design patterns.
CO5: Develop a system using OOP concepts

3. Prerequisites: NIL
4. Semester: 5
5. Course Type: Elective
6. Course Level: 300-399
7. Theory Credit: 4
8. Practical Credit: 0
9. No of Hours:

a) Theory: 60 hrs
b) Practical: 0 hrs
c) Non Contact: 5 hrs

10. List of Books:

c) Rajib Mall: *Fundamentals of Software Engineering*; PHI Learning Pvt. Ltd.
d) Roger S. Pressman: *Software Engineering: A practitioner's Approach*; McGraw Hill.

11. Contents of Syllabus:
A. Theory
Detailed Syllabus:
Unit 1: Introduction (4 Lectures)

Definition of Software Engineering, differentiation between Computer Science, Software Engineering and System Engineering, Program V/s software product, Exploratory style and

modern style of software development, need of software engineering, characteristics of good software product

Unit 2: Software Development Life Cycle models (7 Lectures)

Definition of software development Life cycle (SDLC) models, Various life cycle modes: Classical Waterfall model, Iterative Waterfall model, Prototyping model, Evolutionary (Incremental) model, Spiral model, Agile Model, Agile V/s traditional SDLC Models, SCRUM model, Advantages and disadvantages of each of these SDLC models.

Unit 3: Requirement Analysis and Specification (7 Lectures)

What is Requirement Analysis and Gathering, Concept and Importance of Feasibility Study in Software design, Types of Feasibility: *Technical*, *Economical* and *Operational* feasibility, Software Requirement Specification (SRS) document, Components of an SRS (Software Requirement Specification): Functional and Non-Functional Component, Properties of a good SRS, Different users of SRS, Techniques to represent Complex Logic in SRS: Decision Tree and Decision Table.

Unit 4: Software Project Management (15 Lectures)

Basic idea of Software Project Management, Job Responsibilities of a Software Project Manager, Need of SPMP (Software Project Management Plan) document, Contents of SPMP, Need of Software documentation, Internal and External documentation, Software size estimation using Lines of Code (LOC), Merits and Demerits of LOC metric, Function Point Metric, 3D Function Point metrics, Project Estimation Techniques: *Empirical estimation* and *Heuristics estimation* techniques. Empirical estimation techniques: *Delphi Cost Estimation* and *Delphi Cost Estimation.* Heuristic Estimation Techniques: *Basic COCOMO model* and *Intermediate COCOMO model.* Project Scheduling: *Work break down structure, Activity Networks* and *Critical Path Method.* Project Team structure: *Chief Programmer team* and *Democratic team* structure.

Unit 5: Software Design principles and Methodology (12 Lectures)

Top down and bottom up approach, External Design, Architectural Design and Detailed design, Concept of Cohesion in software design, Classification of Cohesions, Basic concept of Coupling, Classification of Couplings, Introduction to software Analysis and Software Design (SA/SD), Introduction to Data Flow Diagram, Symbols used in DFD, Context Diagram in DFD, Advantages and Disadvantages of DFDs., Balanced DFD, Structured Design: *Transaction Analysis* and *Transform Analysis.* Need of Object Oriented Design and Analysis, UML (Unified Modeling Language), different views of UML, Various UML Diagrams: *Use Case diagram, Class Diagram, Object Diagram, Sequence Diagram* and *Collaboration diagram.*

Unit 6: Coding and Testing (9 Lectures)

Goals of coding, Code Review techniques: Code Walkthrough, Code Inspection, Definition of Test cases, test suits, negative testing and positive testing. Different levels of software testing: *unit testing*, *Integration Testing*, *System Testing* and *acceptance testing*. Differentiation between Verification and Validation, Black box testing approaches: *Equivalent Class Partitioning* and

Boundary Value Analysis, White Box testing approaches: *Statement Coverage, Branch Coverage, Condition* Coverage and Path *Coverage. Approach*, MaCabe's Cyclomatic Complexity, Basic idea of various system testing approaches: *Smoke testing, Stress testing, Volume testing* and *Compatibility testing*

Unit 7: Software Reliability and Maintenance (6 Lectures)

What is reliability? Reliability metrics of Software Products: ROCOF, MTTF, MTTR, MTBF, POFOD and availability. ISO 9000 Certification, need of ISO Certification, How to get ISO 9000 certification, Definition of Software Maintenance, Types of Software maintenance: *Corrective, Adaptive* and *Perfective* maintenance, Estimation of Software Maintenance Cost.

COM050504: Web Technologies

1. Learning Outcome: At the end of the course, students will be able to:

- Understand the basic concept of web applications and web services.
- Design basic well-structured web page using HTML and CSS
- Develop the ability to implement interactive elements and dynamic content using basic JavaScript
- Develop a foundational understanding of server-side scripting using PHP

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO1: Define and describe the working principle of the internet and World Wide Web.
CO2: Analyze a web page and identify its elements and attributes.
CO3: Apply HTML and CSS in designing a website and web application.
CO4: Implement server-side functionality using PHP and JavaScript to create interactive web applications.
CO5: Develop a web project and identify its elements and attributes and build customized web sites and web applications.

3. Prerequisites: NIL
4. Semester: 5
5. Course Type: Compulsory
6. Course Level: 300-399
7. Theory Credit: 3
8. Practical Credit: 1
9. No of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

10. List of Books:

a) Jackson J.C. (2007). *Web Technologies: A Computer Science Perspective*. Pearson.
b) Duckett, J. (2011). *HTML and CSS: Design and Build Websites*. John Wiley & Sons.
c) Robbins, J. N. (2018). *A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics*. O'Reilly Media.
d) Robbins, J. N. (2018). *Learning Web Design: A Beginner's Guide.* O'Reilly Media.
e) Haverbeke, M. (2018). *Eloquent JavaScript*. No Starch Press.
f) Welling, L., & Thomson, L. (2016). *PHP and MySQL Web Development* (5th ed.). Addison-Wesley Professional.

11. Contents of Syllabus:

A. Theory

Detailed Syllabus:

Unit 1: Introduction to Web Technologies (8 Lectures)

Concepts of the Internet and the World Wide Web (WWW), Overview of web browsers and their functionalities. Client-Server Architecture in Web Applications. Communication Protocols – HTTP, HTTPS, FTP. Working of DNS. Brief concepts of port, URL, cache and cookies. Web Content Accessibility Guidelines. Privacy concerns and data protection regulations, GDPR. Introduction to Web Hosting and control panels.

Unit 2: Front End Development using HTML (10 Lectures)

Website and Webpage. Basic concept of Markup Language. Introduction to HTML. Basic HTML structure. Text formatting Tags – headings, paragraph, line break, horizontal rule. Link and Navigation – anchor tags. Lists - ordered, unordered, definition list. Image and multimedia tags. Tables in HTML. Forms and Input types – text, email, password, radio, select, checkbox, textarea, date, url, submit, button. Semantic HTML. Sectioning elements – header, nav, main, section, article, aside, footer.

Unit 3: Front End Design using CSS (9 Lectures)

Introduction to CSS. CSS syntax and rule structure. Inline, Internal and External CSS. CSS selectors – element, class, ID, attribute. Combinators – descendant, child, adjacent sibling, general sibling. Understanding the CSS Box Model – content, padding, border, margin. CSS colours and backgrounds – background-color, background-image, background-repeat. CSS typography – font properties, text properties.

Unit 4: Client-Side Scripting with JavaScript (10 Lectures)

JavaScript as a high-level interpreted language. JavaScript code execution in web browsers – JavaScript execution context. JavaScript syntax and datatypes. JavaScript variables – var, let, const. Assignment and scope of JavaScript variables. Operators in JavaScript – arithmetic, comparison, logical, assignment. Conditional Statements. Looping Structures. Function declaration and Invocation in JavaScript. Introduction to the Document Object Model. Accessing HTML elements in DOM – by id, by tag name, by class name, query selectors. Manipulating DOM elements – create, add, append, remove. InnerText vs InnerHTML. Manipulating CSS styles using DOM. Event handling and delegation with the DOM using JavaScript. Client-side form validation using JavaScript. Handling form validation and processing data.

Unit 5: Server-Side Programming with PHP (8 Lectures)

Introduction to PHP and role in Web development. PHP syntax and variables. Basic PHP functions – Built-in PHP functions, string manipulation functions, mathematical functions, date and time functions. PHP forms and form handling. Form submission methods – GET and POST. Handling form data with PHP. Uploading files with PHP. Introduction to the tech-stack. Role of Apache, PHP, MySQL etc. Introduction to Databases and SQL. Connecting to databases with

PHP. Executing SQL queries with PHP. Retrieving, inserting, updating and deleting data from databases using PHP.

B. List of Practical

(This is a suggestive list only. Questions need not be restricted to this list.)

1. Create a basic HTML webpage structure with a heading, paragraph, and an image.
2. Build a navigation menu using an unordered list (<ul>) with clickable links.
3. Implement a form with input fields for name, email, and a submit button.
4. Create a table with multiple rows and columns to display tabular data.
5. Design an image gallery using HTML and CSS with proper padding and border.
6. Embed a YouTube video on a webpage using the <iframe> tag.
7. Implement an ordered list (<ol>) to display a step-by-step tutorial or instructions.
8. Create a dropdown select menu (<select>) with multiple options.
9. Use HTML5 semantic tags (such as <header>, <nav>, <section>, <article>, <footer>) to structure and organize content on a webpage.
10. Build a registration form with fields for name, email, password, date of birth, address and other such fields with a submit button. Include appropriate input types, labels and placeholders.
11. Style a heading element with a custom font, colour and background.
12. Apply different background colors to alternate rows in a table.
13. Implement a hover effect on a button that changes its background colour or adds a solid border.
14. Style a form input field with custom border, padding, and background color.
15. Implement a CSS tooltip that displays additional information when hovering over an element.
16. Build a simple JavaScript calculator that can perform basic arithmetic operations.
17. Create a button that, when clicked, appends a new paragraph element with a specific text content to an existing div element.
18. Implement a function that changes the innerText of a paragraph element to display a random number between 1 and 10 every time a button is clicked.
19. Build a form with input fields for name and email. When the form is submitted, use innerHTML to display a confirmation message with the entered name and email on the webpage.
20. Build a form with input fields for email, password and confirm password. When the form is submitted, use an alert to display a success message if the password and confirm password values matches, otherwise show an error alert. Use JavaScript for the validation.
21. Create a list of items. Add a click event listener to each item so that when clicked, the background color of the clicked item changes.
22. Write a PHP script to display the current date and time on a webpage.
23. Write a PHP script to connect to a MySQL database and fetch data from a table.

24. Create a registration form with fields for username, email, and password. Implement server-side validation to check for duplicate usernames or invalid email formats. Store the user registration data in a MySQL database. Provide feedback to the user upon successful registration or display appropriate error messages.
25. Design a webpage that displays a list of notices retrieved from a MySQL database. Implement functionality to add new notices to the database using a form. Allow users to view and delete individual notices. Apply appropriate styling to the notices and ensure proper validation and sanitization of user input.

COM060404: Artificial Intelligence

1. Learning Outcome:

After completing this course, students will know-

- The fundamentals of artificial intelligence (AI),
- Identify problems where artificial intelligence techniques are applicable.
- Able to apply basic principles of AI in solutions that require problem solving, inference, perception, knowledge representation, and learning.

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO1: Describe important historical and current trends addressing artificial intelligence.
CO2: Identify and distinguish heuristics searching techniques.
CO3: Apply logic programming concepts in AI.
CO4: Assess the performance and reliability associated with uncertainty and inconsistencies within AI systems.
CO5: Analyze the structure and interpretation of natural languages.

3. Prerequisites: NIL

4. Semester: 6

5. Course Type: Elective

6. Course Level: 300-399

7. Theory Credit: 3

8. Practical Credit: 1

9. No of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

10. List of Books:

a) *Rich & Knight, Artificial Intelligence* – Tata McGraw Hill, 2nd edition, 1991.
b) *Russell &Norvig, Artificial Intelligence*-A Modern Approach, LPE, PearsonPrentice Hall, 2nd edition, 2005.
c) *W.F. Clocksin and Mellish, Programming in PROLOG,*Narosa Publishing House,3rd edition, 2001.
d) *DAN.W. Patterson, Introduction to A.I and Expert Systems* – PHI, 2007.
e) *Ivan Bratko, Prolog Programming for Artificial Intelligence,* Addison-Wesley,Pearson Education, 3rd edition, 2000.

11. Contents of Syllabus:

A. Theory

Detailed Syllabus:

UNIT 1: Introduction (4 Hours)

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 (16 Hours)

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 (14 Hours)

Introduction to First Order Predicate Logic, Resolution Principle, Unification, Semantic Nets, Conceptual Dependencies, Frames, and Scripts, Production Rules, Conceptual Graphs. Programming in Logic (PROLOG)

UNIT 4: Dealing with Uncertainty and Inconsistencies (6Hours)

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

UNIT 5: Understanding Natural Languages (5Hours)

Parsing Techniques, Context-Free and Transformational Grammars, Recursive and Augmented Transition Nets.

Practical:

- Write a prolog program to calculate the sum of two numbers.
- Write a prolog program to find the maximum of two numbers.
- Write a prolog program to calculate the factorial of a given number.
- Write a prolog program to calculate the nth Fibonacci number.
- 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 nth element into every list atall levels.
- Write a Prolog program to remove the nth item from a list.
- Write a Prolog program, remove nth (Before, After) that asserts the After list is the Before list with the removal of every nth item from every list at alllevels.
- Write a Prolog program to implement append for two lists.
- Write a Prolog program to implement palindrome (List).
- Write a Prolog program to implement max(X,Y,Max) so that Max is thegreater of two numbers X and Y.
- Write a Prolog program to implement maxlist(List,Max) so that Max is thegreatest number in the list of numbers List.
- Write a Prolog program to implement sumlist(List,Sum) so that Sum is thesum of a given

list of numbers List.

- Write a Prolog program to implement two predicates evenlength(List) and oddlength (List) so that they are true if their argument is a list of even or oddlength respectively.
- Write a Prolog program to implement reverse (List, Reversed List) thatreverses lists.
- Write a Prolog program to implement maxlist (List, Max) so that Max is thegreatest number in the list of numbers List using cut predicate.
- Write a Prolog program to implement GCD of two numbers.
- Write a prolog program that implements Semantic Networks/FrameStructures.

COM060104: Automata Theory and Languages

1. Learning Outcome: After completing this course, students

- Understand the Mathematical model of a finite state machine. Know deterministic and non-deterministic versions of Finite automata.
- Grasp the mathematical concepts of languages and grammar.
- Know Pushdown Automata and the associated grammar/language.
- Know the properties of Regular languages and Context free languages.

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO1: Define the concept of finite automata and language accepted by it.
CO2: Discuss regular grammar and the properties of regular languages.
CO3: Explain the concept of PDA and language accepted by it.
CO4: Analyze the simplification of context free language and Normal forms of context free language.
CO5: Formulate the concept of pumping lemma for regular language and context free language

3. Prerequisites: NIL
4. Semester: 6
5. Course Type: Compulsory
6. Course Level: 300-399
7. Theory Credit: 4
8. Practical Credit: 0
9. No of Hours:

a) Theory: 60 hrs
b) Practical: 0 hrs
c) Non Contact: 5 hrs

10. List of Books:

a) *An introduction to Formal Languages and Automata*, Peter Linz,Narosa.
b) *Introduction to Automata Theory, Languages and Computation*, Hopcroft, Motwani and Ullman, Pearson.
c) *Theory of Computer Science (Automata, Languages and Computation*), K. L. P. Mishra, N. Chandrasekaran; P. H.I.

11. Contents of Syllabus:
A. Theory
Detailed Syllabus:
UNIT 1: Finite Automata (10 Lectures)
DFA, NFA, NFA with empty-moves, Equivalence of DFA and NFA, Reduction of the number of states in finite automata.

UNIT 2: Regular Languages and Regular Grammar (12 Lectures)

Concept of languages and grammar, Regular expressions, Connection between regular expressions and regular languages, Regular grammars, Right and Left-Linear Grammars, Equivalence between Regular languages and Regular grammars.

UNIT 3: Properties of Regular Languages (13 Lectures)

Closure under simple set operations- union, intersection, concatenation, complementation and star closure, Decision algorithms for emptiness, finiteness and infiniteness, equality, Proof of non- regularity using Pigeonhole principle and using pumping lemma for regular languages.

UNIT 4: Context Free languages (15 Lectures)

Context-free grammars, leftmost and rightmost derivations, derivation trees, Parsing and Ambiguity in grammars and languages, Simplification of Context free Grammars- removing useless productions, empty-productions and unit-productions. Normal forms- Chomsky and Greibach normal forms, Pumping Lemma for CFL, Using Pumping Lemma to show that certain languages are not Context free.

UNIT 5: Pushdown Automata (10 Lectures)

Definition and language accepted (acceptance by empty stack and final state and their equivalence), Pushdown Automata and Context free languages. Deterministic PDA and Deterministic Context freeLanguages.

COM060204: Cloud Computing

1. Learning Outcome:

After completing this course, students will know about cloud computing environment, it need and applications.

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO1: Describe client server architecture.
CO2: Illustrate the cloud computing architecture.
CO3: Explain about different Service Level Agreements (SLAs).
CO4: Illustrate data management techniques and security methods used in cloud computing.
CO5: Formulate different case studies on open source and commercial clouds

3. Prerequisites: NIL
4. Semester: 6
5. Course Type: Elective
6. Course Level: 300-399
7. Theory Credit: 4
8. Practical Credit: 0
9. No of Hours:

a) Theory: 60 hrs
b) Practical: 0 hrs
c) Non Contact: 5 hrs

10. List of Books:

a) *Cloud Computing: Principles and Paradigms*, Editors: Rajkumar Buyya, James Broberg, Andrzej M. Goscinski, Wiley,2011

b) *Enterprise Cloud Computing - Technology*, Architecture, Applications, Gautam Shroff, Cambridge University Press, 2010

c) *Cloud Computing Bible*, Barrie Sosinsky, Wiley-India, 2010

d) *Cloud Security: A Comprehensive Guide to Secure Cloud Computing*, Ronald L. Krutz, Russell Dean Vines, Wiley- India,2010

e) *Cloud computing*, Ashish Bhatnagar, KATSON Books.

f) *NPTEL :Cloud computing*, By Prof. Soumya Kanti Ghosh, IIT Kharagpur

11. Contents of Syllabus:

A. Theory

Detailed Syllabus:

Unit 1: Introduction to Cloud Computing (10 Lectures)

Introduction, Definition, basic concepts and terminology, characteristics, goals and benefits, risks and challenges, historical developments, clouds types, Role of networks in cloud computing, Virtualization Technology, Enterprise knowledge clouds, Cloud Computing(NIST Model), Client server Architecture, Client server model vs. Cloud model.

Unit 2: Cloud Computing Architecture (10 Lectures)

Introduction, Cloud Computing stack, Service models(XaaS) : Infrastructure as a Services(IaaS), Platform as a service(PaaS), Software as a Service(SaaS), Application of XaaS, Deployment Models, Microsoft Azure vs Amazon EC2

Unit 3: Service Management in Cloud Computing (10 Lectures)

Service Level Agreements(SLAs), SLA contents, Web Service SLA, Difference between Cloud SLA and Web service SLA, Types of SLA, Service level objectives, Service level management, Considerations for SLA, SLA requirements, Cloud properties: Economic viewpoint

Unit 4: Data Management in Cloud Computing (10 Lectures)

Introduction: Relational database, Google File system, BigTable, MapReduce, Data Storage Techniques, Looking at Data, Scalability & Cloud Services, Database & Data Stores in Cloud, Large scale data processing, Parallel database.

Unit 5: Cloud Security (10 Lectures)

Security – Basic components, Security attacks, Infrastructure Security, Data Security and Storage, Identity and Access Management, Access control, Trust, Reputation, Risk.

Unit 6: Case Study on Open Source and Commercial clouds (10 Lectures)

OpenStack, OpenStack Capability, OpenStack History, OpenStack Architecture, OpenStack components, Meghamala(IITKGP), Google Cloud Platform, Microsoft Azure

COM060304: Complier Design

1. Learning Outcome:

a) Use compiler construction tools and describes the Functionality of each stage of compilation process
b) Construct Grammars for Natural Languages and find the Syntactical Errors/Semantic errors during the compilations using parsing techniques
c) Analyze different representations of intermediate code.
d) Construct new compiler for new languages.
e) Participate in GATE, PGECET and other competitive examinations

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO1: Describe about compilers, its phases and working of compilers.

CO2: Computer Regular Expressions, NFA, DFA to recognized tokens.

CO3: Design different parsers such as Top down parser, Bottom up parsers like LR, SLR and LALR

CO4: Explain intermediate representation i.e. generation of three address codes, representation of it using DAG and devise code optimization solution for a given problem.

CO5: Develop Lexical analyzer using Lex and parser generator using Yacc.

3. Prerequisites: NIL
4. Semester: 6
5. Course Type: Elective
6. Course Level: 300-399
7. Theory Credit: 4
8. Practical Credit: 0
9. No of Hours:

a) Theory: 60 hrs
b) Practical: 0 hrs
c) Non Contact: 5 hrs

10. List of Books:

a) *Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman* (2007), *Compilers:Principles, Techniques and Tools*, 2nd edition, Pearson Education, New Delhi, India.
b) Alfred V. Aho, Jeffrey D. Ullman (2001), *Principles of compiler design*, Indian student edition, Pearson Education, New Delhi, India.
c) *Kenneth C. Louden (1997),Compiler Construction– Principles and Practice*, 1st edition, PWS Publishing.
d) *K. L. P Mishra, N. Chandrashekaran (2003*), *Theory of computer science- Automata Languages and computation*, 2nd edition, Prentice Hall of India, New Delhi, India.
e) *Andrew W. Appel (2004*), *Modern Compiler Implementation C*, Cambridge University Press, UK.
f) *John R. Levine, Tony Mason, Doug Brown, Lex &Yacc*, O'reilly

11. Contents of Syllabus:

A. Theory

Detailed Syllabus:

UNIT 1: Introduction to Compiler(12 Lectures)

Definition of compiler, Phases of a compiler, Lexical analysis, Role of lexical analyzer, Tokens, Patterns, Lexemes, Input buffering, Specification of tokens-strings and languages, operations on languages, regular expressions, regular definitions, Recognition of tokens, Lexical analyzer generator- Lex, Finite automata, From Regular expressions to automata.

UNIT 2: Syntax Analysis (16 Lectures)

Parsing, Role of parser, Context free grammar, Parse tree and derivations, Ambiguity, Eliminating ambiguity from dangling-else grammar, Elimination of left recursion, Left factoring, Top Down Parsing- Recursive descent parser, Predictive parser- LL(1) Grammar, construction of predictive parsing table.

Bottom Up Parsing- Reductions, Handle pruning, Shift-Reduce parsing, Conflicts during shift-reduce parsing, LR Parser-Items, Kernel items, Non-kernel items, closure of Item Sets, The function GOTO, LR (0) automaton, Construction of SLR parsing table, Basics of LALR parser, Automatic parser generator-YACC.

UNIT 3: Syntax Directed Translation (12 Lectures)

Syntax directed definition- inherited and synthesized attributes, evaluating an SDD at the nodes of a parse tree, Evaluation orders of SDD's- dependency graphs, ordering the evaluation of attributes, S-attributed and L-attributed definitions, Applications of syntax-directed translation-construction of syntax trees, the structure of a Type, Syntax directed translation schemes-postfix translation schemes, SDT's with actions inside productions, eliminating left recursion from SDT's, Variants of syntax trees- directed acyclic graphs (DAG) for expressions, The value-number method for constructing DAG's, Three address code- Quadruples, Triples and Indirect triples, Static single- assignment form, Types and Declarations, Translation of expressions, Type Checking, Basics of Control flow, Basics of Backpatching.

UNIT 4: Run Time Environments (10 Lectures)

Storage organization, Stack allocation of space, Access to non-local data on Stack, Basics of Heap management, Basics of garbage collection

UNIT 5: Code Generation and optimization (10 Lectures)

Machine dependent code generation, Issues in design of code generator, The target language, Addresses in the target code, Basic blocks and flow graphs,Optimization of basic blocks- the DAG representation of Basic blocks, Finding local common sub-expression, dead code elimination, A simple code generator, Basics of Peephole optimization, The Principal Sources of Optimization, Introduction to Data-Flow Analysis.

.

COM060504: Computer Graphics

1. Learning Outcome:

After completing this course, students will know about-

- Basic elements of Computer Graphics
- Fundamental of Computer graphics algorithms along with basic mathematical foundations of computer graphics.

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO1: Describe basic knowledge of core concepts of computer graphics input/output systems.
CO2: Explain different techniques for designing computer graphics software.
CO3: Design and construct complex geometric models using advanced modeling techniques.
CO4: Utilize visible surface detection algorithms to render realistic scenes.
CO5: Apply basic illumination models to simulate lighting effects and enhance the visual realism of rendered scenes.

3. Prerequisites: NIL
4. Semester: 6
5. Course Type: Elective
6. Course Level: 300-399
7. Theory Credit: 3
8. Practical Credit: 1
9. No of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

10. List of Books:

a) D. Hearn, M. Baker: Computer Graphics, Prentice Hall of India 2008.
b) J.D.Foley, A.Van Dan, Feiner, Hughes Computer Graphics Principles & Practice2nd edition Publication Addison Wesley 1990.
c) D.F.Rogers Procedural Elements for Computer Graphics, McGraw Hill 1997.
d) D.F.Rogers, Adams Mathematical Elements for Computer Graphics, McGraw Hill, 2nd edition 1989.

11. Contents of Syllabus:

A. Theory

Detailed Syllabus:

UNIT 1: Introduction **(2 Hours)**

Basic elements of Computer Graphics, Applications of Computer Graphics

UNIT 2: Graphics Hardware **(5 Hours)**

Input Devices: Keyboard, Mouse, Trackball & Space ball, Joystick, Data Glove, Digitizers, Image Scanners, Touch panels, Light Pens systems. Output display devices: Refresh CRT,

Raster-Scan display and Random-scan display technique, Color display techniques-Beam penetration method and Shadow-mask method, Direct view storage tubes, Emissive & Non-emissive flat-panel, Displays-Plasma panels, LED and LCD monitor, Three-dimensional viewing devices and Virtual-Reality systems Display processor: Raster-scan systems, Random-scan systems

UNIT 3: Fundamental Techniques in Graphics (20 Hours)

Line-drawing algorithms:DDA algorithm and Bresenham's Line drawing Algorithm, Midpoint Algorithm for Circle and Ellipse Generation, Curve generation.Attributes for output primitives: Area-filling Algorithms - Scan-line Polygon-fill, 2-D Geometric Transformations: Basic transformations-translation, Rotation and ScalingMatrix representations and Homogeneous Co-ordinate representations, Composite transformations among translation, Rotation and Scaling, 2-D viewing: Definition, Viewing transformation pipeline, Window-to-viewport Co-ordinate transformation.

2-D Clipping: Concept and Algorithm: Point clipping, Line clipping - Cohen-Sutherlandalgorithm, Area clipping, Text clipping, Polygon clipping.3-D concepts: Display methods-Parallel projection, perspective projection3-D geometric transformations: Transformation, Translation, Rotation and Scaling around axes, 3-D Viewing Projections – Parallel and Perspective.

UNIT 4: Geometric Modelling (8Hours)

Representing curves and surface, Bezier curves and surfaces – Definition of Bezier curve and its properties, Algorithms for Bezier curves and surfaces, Hermite curve

UNIT 5: Visible Surface determination (5Hours)

Definition, approaches for visible surface detection, object-space methods- Back-Face Detection, Image space methods: Depth Buffer Methods, A Buffer Method, Scan Line Method, Depth-Sorting Method

UNIT 6: Surface rendering (5Hours)

Definition and importance, light sources, Basic illumination models-Ambient light,Diffuse reflection, Specula reflector and Phong model

Practical:

- Write a program to implement DDA algorithm for line drawing.
- Write a program to implement Bresenham's line drawing algorithm.
- Write a program to implement mid-point circle drawing algorithm.
- Write a program to clip a line using Cohen-Sutherland line clipping algorithm.
- Write a program to clip a polygon using Sutherland Hodgeman algorithm.
- Write a program to apply 2D translation on a 2D object (use homogenous coordinates).
- Write a program to apply 2D rotation on a 2D object (use homogenous coordinates).
- Write a program to apply 2D scaling on a 2D object (use homogenous coordinates).
- Write a program to apply 2D reflection of a 2D object (use homogenous coordinates).
- Write a program to apply 2D shear operation on a 2D object (use homogenous coordinates).
- Write a program to apply 3D translation on a 3D object (use homogenous coordinates).
- Write a program to apply 3D rotation on a 3D object (use homogenous coordinates).

- Write a program to apply 3D scaling on a 3D object (use homogenous coordinates).
- Write a program to apply 3D reflection of a 3D object (use homogenous coordinates).
- Write a program to apply 3D shear operation on a 3D object (use homogenous coordinates).
- Write a program to draw Hermite/Bezier curve.

COM060604: Data Mining and Warehousing

1. Learning Outcome:

a) Understanding the process of Knowledge Discovery in Databases
b) Understand the functionality of the various data warehousing component.
c) Characterize the kinds of patterns that can be discovered by association rule mining.
d) Analysis of different types of data by clustering and classification.

2. COURSE OUTCOMES:

CO1: Describe the components and architecture of data warehouse architecture
CO2: Illustrate different data mining techniques such as association rule mining, clustering and classification.
CO3: Develop various algorithms like A priori, Pincer Search and DIC used in association rule mining
CO4: Explain the working principle of various clustering and classification algorithms like K-means, K-Mediods, PAM, DBSCAN, K-Nearest Neighbor Classifiers etc.
CO5: Analyze the uses of web mining, text mining, big data mining, neural networks etc.

3. Prerequisites: NIL
4. Semester: 6
5. Course Type: Elective
6. Course Level: 300-399
7. Theory Credit: 3
8. Practical Credit: 1
9. No of Hours:

a) Theory: 45 hrs
b) Practical: 30 hrs
c) Non Contact: 5 hrs

9. List of Books:

a) A.K. Puzari, *Data Mining Techniques*, University Press.
b) J. Han, J. Pie and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann.
c) P. Tan, M. Steinbach and V. Kumar, *Introduction to Data Mining*, Pearson Education (LPE).
d) G. K. Gupta, *Introduction to Data Mining with Case Studies*, PHI.

10. Contents of Syllabus:

A. Theory

Detailed Syllabus:

UNIT 1: Overview **(4 Lectures)**

What is Data Mining?, Knowledge Discovery in Databases (KDD) vs. Data Mining, Types of Data, Basic Data Mining Tasks, Predictive and Descriptive data mining techniques, Supervised and Unsupervised learning techniques, Basics of Pre-processing methods- Data Cleaning, Data Integration and Transformation, Data Reduction, Data Visualization.

UNIT 2: Data Warehousing **(6 Lectures)**

What is Data Warehouse? Multidimensional Data Model, Data Cube, Basic Components of Multidimensional Data Model, OLAP Operations- Slicing, Dicing, Drilling, Drill-Up, Drill-Down, Drill-Within, Drill-Across, Pivot(Rotate), Schema of Warehouse, Data Warehouse Architecture, Metadata.

UNIT 3: Association Rule Mining **(12 Lectures)**

What is Market Basket Data?, k-Itemset, Support of an Itemset, Frequent Itemsets, Infrequent Itemsets, Maximal Frequent Itemsets, Closed Frequent Itemsets, Association Rules, Confidence of a Rule, Problem of Mining Association Rules, Algorithm for Mining Frequent Itemsets- Apriori Algorithm, Pincer-Search Algorithm, DIC (Dynamic Itemset Counting) Algorithm, Steps of Mining Association Rules.

UNIT 4: Clustering **(12 Lectures)**

What is Clustering, Partitional vs Hierarchical Clustering, Types of Data in Clustering, Distance Measures used in Clustering- Euclidean Distance, Manhattan Distance, Similarity Measures used in Clustering- Cosine Similarity, Jacquard Coefficient, Partitional Clustering Methods- K-Means, K-Mediods, PAM, CLARA, CLARANS, Density Based Clustering Methods- DBSCAN, Introduction to Hierarchical Clustering.

UNIT 5: Classification **(8 Lectures)**

What is Classification? Issues Regarding Classification, K-Nearest Neighbor Classifiers, Bayesian classification, Introduction to Decision Tree.

UNIT 6: Recent Trends and Techniques used in Data Mining **(3 Lectures)**

Basic Concepts of- Web Mining, Spatial Data Mining, Temporal Data Mining, Big Data Mining, Concept of Neural Network, Genetic Algorithm.

Practical / Lab work to be performed

- Implement ***any one*** from the following-
    - Write a computer program to implement A priori algorithm to mine all frequent itemsets from a transactional dataset. Use hashing to store the item sets in the level wise generation of candidate sets.
    - Write a computer program to implement the Pincer Search algorithm.
    - Write a computer program to implement the DIC (Dynamic Item set) algorithm.
- Implement ***any four*** from the following-
    - Write computer program to implement the K-Means algorithm using different distance measures stated in the syllabus.

- Write computer program to implement the PAM algorithm using different similarity measures stated in the syllabus.
- Write a computer program to implement the CLARA algorithm.
- Write a computer program to implement the CLARANS algorithm.
- Write a computer program to implement the DBSCAN algorithm.
- Write a computer program to implement the K-NN algorithm.

COM060704: Design and Analysis of Algorithms

1. Learning Outcome:

After successful completion of this course, students will:

- Know how to analyze algorithms.
- Learn the different algorithm design techniques.
- Be acquainted with the advanced sorting and searching algorithms and their complexities.
- Know graph representation techniques together with traversal algorithms.
- Know why tree balancing is required and how to achieve this.

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO1: Recognize different algorithm analyzing techniques.
CO2: Identify basic algorithm design techniques.
CO3: Explain various sorting and searching techniques.
CO4: Implement balanced trees and graph algorithms.
CO5: Analyze the complexity of string matching algorithms.

3. Prerequisites: NIL

4. Semester: 6

5. Course Type: Elective

6. Course Level: 300-399

7. Theory Credit: 4

8. Practical Credit: 0

9. No of Hours:

a) Theory: 60 hrs
b) Practical: 0 hrs
c) Non Contact: 5 hrs

10. List of Books:

a) *Introduction to Algorithms*,Cormen. T. H., Leiserson C. E. and Rivest. R. L., 3rd edition (2010)Tata-McgrawHill Publishers.
b) *Fundamentals of Computer Algorithms*; Horowitz and Sahani; (2nd Edition), Galgotia.
c) *Design and Analysis of Computer Algorithms*; Aho.A, Hopcroft J.E. and Ullman J.D.; (2011), PearsonEducation.
d) *Introduction to the Design and Analysis of Algorithms*, Levitin, 3/e 2017, Pearson Education.

11. Contents of Syllabus:

A. Theory

Detailed Syllabus:

UNIT 1: Introduction (6 Hours)

Analysisof Algorithms – worst case and average case analysis; Time and space complexity of algorithms; Asymptotic notations O and θ. Proving correctness of algorithms.

UNIT 2: Algorithm Design Techniques (10 Hours)

Iterative techniques, Divide and Conquer, Dynamic Programming, GreedyAlgorithms. Applications of these techniques in problems like sorting, searching, matrix multiplication, LCS (Longest Common Sequence) problem, Knap-sack problem.

UNIT 3: Sorting and Searching Techniques (20 Hours)

Elementary sorting techniques–Bubble Sort, Insertion Sort, Merge Sort, AdvancedSorting techniques - Heap Sort, Quick Sort, Sorting in Linear Time - Bucket Sort,Radix Sort and Counting Sort, Searching Techniques, Medians & Order Statistics,complexity analysis of all the techniques.

UNIT 4: Balanced Trees (9 Hours)

Tree balancing, Height of a Red-Black tree, Rotations - Left Rotations, Right Rotations, Insertion and
Deletion in Red-Black trees.

UNIT 5: Graph Algorithms (9 Hours)

Representations of Graphs; Adjacency Matrix and Adjacency Lists. Simple operations like computing degree, indegree, outdegree of vertices using the representation techniques and computing work done in all cases. Graph traversal algorithms–Breadth First Search, Depth First Search and their Applications.

UNIT 6: String Processing (6 Hours)

String Matching, KMP Technique.

COM060804: Graph Theory

1. Learning Outcome:

- After completing this course, students will have understanding of graph theoretic concepts, problems and associated algorithmic solutions.

2. COURSE OUTCOMES:

At the end of the course, students will be able to:

CO1: Discuss different types of graph and graph traversing methods.
CO2: Explain graph connectivity and solve various shortest path problems.
CO3: Distinguish between tree graph algorithms.
CO4: Discuss classical problems in graph theory and digraphs.
CO5: Apply graph coloring algorithms in real world problems.

3. Prerequisites: NIL
4. Semester: 6
5. Course Type: Elective
6. Course Level: 300-399
7. Theory Credit: 4
8. Practical Credit: 0
9. No of Hours:

a) Theory: 60 hrs
b) Practical: 0 hrs
c) Non Contact: 5 hrs

10. List of Books:

a) *Introduction to Graph Theory*, Douglas B. West, Pearson
b) *Introduction to Graph Theory*, Robin J. Wilson, Pearson Education Limited
c) *Graph Theory with Applications to Engineering and Computer Science*, Narasingh Deo, PH

11. Contents of Syllabus:

A. Theory

Detailed Syllabus:

Unit I: Introduction (5 hrs)

Graph, directed and undirected graph, weighted and unweighted graph, simple and multigraph, degree, in degree and out degree, Handshaking theorem, complete graph, bipartite graph, cut set, cut vertices, graph representations: incidence matrix, adjacency matrix and adjacency list, BFS traversal and DFS traversals on a graph using stack and queue data structures, isomorphism, homomorphism

Unit II: Connectivity, paths and cycle (15 hrs)

Walk, path and cycle, connected graphs, disconnected graphs, components, Hamiltonian path, Hamiltonian cycle, Hamiltonian graphs, Dirac's theorem, Eulerian path, Eulerian cycle, Euler

graphs, Fleuri's algorithm, 2-connected graphs, connectivity and digraph, k-connected and k-edge connected graphs, application of Menger's theorem, Shortest path problem, variations of shortest path problem: single source shortest path problem, single pair shortest path problem and all pairs shortest path problem, Dijkstra's algorithm, Bellman Ford algorithm, Floyd Warshall's algorithm, Johnson's algorithm

Unit III: Tree **(12 hrs)**

Tree, forest, properties of tree, spanning tree, spanning forest, counting trees, Cayley's theorem, matrix-tree theorem, minimum spanning tree, Kruskal's algorithm, Prim's algorithm, disjoint spanning trees, graph decomposition, graceful labeling, graceful graph, binary tree, binary search tree, AVL tree, multiway search tree, B tree, B+ tree

Unit IV: Matching and coloring **(13 hrs)**

Matching, bipartite matching, maximum bipartite matching, Ford Fulkerson's algorithm for finding maximal bipartite matching, perfect bipartite matching, non-bipartite matching, maximal non-bipartite matching, largest maximal matching, perfect non-bipartite matching, Hall's Marriage theorem, vertex cover, vertex cover and matching, independent sets, dominating sets, atable matching, Hungarian algorithm, introduction to Edmonds Blossom shrinking algorithm, vertex coloring, k-colorable graph, chromatic number, Brook's theorem, clique number, map coloring problem

Unit V: Digraph **(7 hrs)**

Digraph, simple digraph, connected and strongly connected digraph, orientable graph, Eulerian digraph, Hamiltonian digraph, tournament, Markov chains, Flow networks, residual graph, augmenting path, Ford Fulkerson's algorithm

Unit VI: Classical problems **(8 hrs)**

Travelling Salesman Problem, variants of Travelling Salesman Problem, Chinese Postman Problem, variants of Chinese Postman Problem, the minimum connector problem, Huffman coding and Huffman tree, Konisgsberg bridge problem, three utilities problem