SYLLABUS OF

Bachelor of Science (Information Technology)

PROGRAM

Department of Computer Science
Gauhati University

FYUGP B.Sc. IT NEW STRUCTURE: JAN-2025

Section 1: To be implemented from 2025 newly admitted Student 2025-26 Onwards:

Section 2: To be implemented from 2025 Existing FYUGP-2024 Batch

Section 3: To be implemented from 2025 Existing FYUGP-2023 Batch

Section 1:

Semester-1			Semester-2		
Type	**Course**	**Credit**	**Type**	**Course**	**Credit**
Core	Introduction to C-Programming	4	Core	Digital Logic Fundamentals	4
Core	Mathematical Foundation in Information Technology	4	Core	Data Structures & Algorithms Using C	4
SEC	SEC-1(Computer Fundamentals and Application Software)	3	SEC	SEC-2(Web Technologies)	3
AEC	AEC-1(Languages/Alt. English)	4	AEC	AEC-2 (Communicative Eng)	4
MDC	MDC-1	3	MDC	MDC-2	3
VAC	VAC-1	2	VAC	VAC-2	2
		20			**20**

Semester-3			Semester-4		
Type	**Course**	**Credit**	**Type**	**Course**	**Credit**
Core	Computer Organization and Architecture	4	Core	Database Management System	4
Core	Operating system	4	Core	Computer Oriented Numerical and Statistical Methods	4
Core	Object Oriented Programming through C++	4	Core	Automata Theory and Languages	4
SEC	Advanced Web Programming	3	Core	Python Programming	4
MDC	MDC-3	3	Core	Design and Analysis of Algorithms	4
VAC	VAC-3	2			
		20			**20**

Semester-5			Semester-6		
Type	**Course**	**Credit**	**Type**	**Course**	**Credit**
Core	Software Engineering	4	Core	Computer Graphics	4
				Optimization Techniques	
Core	Java Programming	4	Core	Artificial Intelligence	4
				Mobile Application Development	
Core	Computer Networks	4	Core	Data Mining and Warehousing	4
Core	Information Security and Cyber Laws	4	Core	Project	4
Internship	Internship	4	Core	Graph Theory/Software Testing	4
		20			**20**

Section 2:

NB: Detailed Syllabus to be followed as same as Section 1

Semester-1			Semester-2		
Type	***Course***	***Credit***	**Type**	**Course**	**Credit**
Core	*Core-1*	*4*	*Core*	Digital Logic Fundamentals	4
Core	*Core-2*	*4*	*Core*	Data Structures & Algorithms Using C	4
Core	*Core-3 (Remains as additional))*	*4*	SEC	SEC-2(Web Technologies)	3
SEC	*SEC-1*	*3*	AEC	AEC-2 (Communicative English)	4
AEC	*AEC-1*	*2*	MDC	MDC-2	3
MDC	*MDC-1*	*3*	VAC	VAC-2	2
VAC	*VAC-1*	*2*			
		22			**20**

Semester-3			Semester-4		
Type	**Course**	**Credit**	**Type**	**Course**	**Credit**
Core	Computer Organization and Architecture	4	*Core*	Database Management System	4
Core	Operating system	4	*Core*	Computer Oriented Numerical and Statistical Methods	4
Core	Object Oriented Programming through C++	4	*Core*	Automata Theory and Languages	4
SEC	Advanced Web Programming	3	*Core*	Python Programming	4
MDC	MDC-3	3	*Core*	Design and Analysis of Algorithms	4
VAC	VAC-3	2	AEC	AEC-3(Special)	2
		20			**22**

Semester-5			Semester-6		
Type	**Course**	**Credit**	**Type**	**Course**	**Credit**
Core	Software Engineering	4	*Core*	Computer Graphics	4
				Optimization Techniques	
Core	Java Programming	4	*Core*	Artificial Intelligence	4
				Mobile Application Development	
Core	Computer Networks	4	*Core*	Data Mining and Warehousing	4
Core	Information Security and Cyber Laws	4	*Core*	Project	4
Internship	Internship	4	Minor	Graph Theory/ Software Testing	4
		20			**20**

Section3:

NB: Detailed Syllabus to be followed as previous 2023 FYUGP syllabus

Semester-1			Semester-2		
	Completed			***Completed***	
		Credit :*22*			**Credit : 22**

Semester-3			Semester-4		
			Type	**Course**	**Credit**
	Completed		*Core*	Database Management System	4
			Core	Design and Analysis of Algorithms	4
			Core	Automata Theory and Languages	4
			Core	Python Programming	4
			AEC	AEC-3	4
		Credit :18			**20**

Semester-5			Semester-6		
Type	**Course**	**Credit**	**Type**	**Course**	**Credit**
Core	Software Engineering Compulsory	4	*Core*	Computer Graphics Elective	4
				Information Security and Cyber Laws	
				Computer Oriented Numerical and Statistical Methods	
Core	Web Technologies	4	*Core*	Artificial Intelligence Advanced Web Programming	4
				Data Mining and Warehousing	
Core	Java Programming	4	*Core*	Optimization Techniques	4
				Mobile Application Development	
				Graph Theory/Software Testing	
	Computer Networks	4	*Core*	Project	4
Internship	Internship	4	VAC	VAC-3	2
		20			**18**

Semester-1			Semester-2		
Type	**Course**	**Credit**	**Type**	**Course**	**Credit**
Core	Introduction to C-Programming	4	Core	Digital Logic Fundamentals	4
Core	Mathematical Foundation in Information Technology	4	Core	Data Structures & Algorithms Using C	4
SEC	SEC-1 Computer Fundamentals and Application Software	3	SEC	SEC-2 Web Technologies	3
AEC	AEC-1	4	AEC	AEC-2	4
MDC	MDC-1	3	MDC	MDC-2	3
VAC	VAC-1	2	VAC	VAC-2	2
		20			**20**

Semester-3			Semester-4		
Type	**Course**	**Credit**	**Type**	**Course**	**Credit**
Core	Computer Organization and Architecture	4	Core	Database Management System	4
Core	Mathematics II	4	Core	Computer Oriented Numerical and Statistical Methods	4
Core	Object Oriented Programming through C++	4	Core	Automata Theory and Languages	4
SEC	Advanced Web Programming	3	Core	Python Programming	4
MDC	MDC-3	3	Core	Design and Analysis of Algorithms	4
VAC	VAC-3	2			
		20			**20**

Semester-5			Semester-6		
Type	**Course**	**Credit**	**Type**	**Course**	**Credit**
Core	Software Engineering	4	Core	Computer Graphics Optimization Techniques	4
Core	Java Programming	4	Core	Artificial Intelligence Mobile Application Development	4
Core	Computer Networks	4	Core	Data Mining and Warehousing	4
Core	Information Security and Cyber Laws	4	Core	Project	4
Internship	Internship	4	Core	Graph Theory/Software Testing	4
		20			**20**

Introduction to C-Programming

1. Learning Outcomes: At the end of the course, students will be able to:
   (a) Understand the basics of C programming like data types and operators
   (b) Understand and write program in C to implement conditions, loops, functions
   (c) Work on arrays, strings and basic file operations

2. Prerequisites: NIL

3. Semester: 1

4. Course type: Compulsory

5. Course level: 100-199

6. Theory credit: 3

7. Practical credit: 1

8. Number of required hours:

   a) Theory: 45 hrs (45 classes)

   b) Practical: 30 hrs (15 classes)

   c) Non Contact: NIL

9. 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) P. Greg, D. Miller. "C Programming: Absolute Beginner's Guide", 3rd ed. Que, 2016.

10. Detailed Syllabus:

A. Theory

Unit 1: Getting started with C programming (10 Lectures)

Introduction to programming languages- High-level vs low level languages, compiled vs interpreted languages. Structure of a C program. Introduction to Header files. Main function and a simple program execution. Compiling and executing a program. C tokens – keywords, identifiers, constants, operators. Statements and expressions in C. Basic data types in C - integers, floats, doubles, characters. Void. Size and range of values of data types. Variables. Constants – integer constant, real constant, character constant, string constant. Declaration and initialization of variables and constants. Assigning values to variables. Operators in C – binary and unary operators. Arithmetic, assignment, logical, comparison, bitwise and conditional operators. Order of precedence of operators. Associativity of operators. Input and output statements – getchar(), getc(), getch(), putchar(), putc(), puts(), scanf(), printf(), format specifiers. Typecasting.

Unit 2: Control Structures in C (9 Lectures)

Control Structures in C. Basic programming constructs- Sequence, selection and iteration. Conditional statements – if, else, switch case. Nested conditions. Loops – for loop, while loop, do-while loop. Using loop for counting iterations. Using while loop for indefinite iterations. Nested loops. Break and continue statements.

Unit 3: Arrays and Strings **(8 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.

Unit 4: Functions and Pointers **(9 Lectures)**

Introduction to Pointers. Pointer declaration and initialization. Pointers and addresses. Pointers and Arrays. Basic concept of dynamic memory allocation, malloc(), calloc(). Introduction to functions. Function declaration and definition. Return types of function. Function arguments. Function calling – call by value vs call by reference. Passing an array as argument to a function. Basic concept of recursion.

Unit 5: Introduction to Structures and Unions **(4 Lectures)**

Basic concept of Structures and Unions in C. Structure declaration and initialization. Union declaration and initialization. Difference between structures and unions.

Unit 6: File Processing in C **(5 Lectures)**

Basic concept of file handling. Opening and closing file using fopen() and fclose(). Binary vs text files. Reading and writing files – fgets(), fscanf(), fprintf(). Random access to files.

B. 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)

(a) Write a program in C to print "Hello World"
(b) Write a program to take input of two numbers and print their sum, product, difference.
(c) Write a program to find the smallest or greatest of three numbers given as input.
(d) Write a program to print the sum and product of digits of an integer.
(e) Write a program to take a number representing a month and print the name of the month using switch case.
(f) Write a program that calculates the grade of a student based on their marks in a subject using nested if-else statements. Also print the range of marks for each grade using switch case.
(g) Write a program to take a number as input and print all the even numbers up to that number using while and for loop.
(h) Write a program to ask the user for an input to stop a loop or continue repeating after printing the iteration count using a do-while loop.
(i) Write a program to find the maximum, minimum, sum and average of n numbers without using array.
(j) Write a program that takes two integers as input and finds their greatest common divisor (GCD) using nested while loops and if statements.
(k) Write a program that calculates the sum of the first n terms of the Fibonacci sequence, where n is entered by the user, using a for-loop.
(l) Write a program that takes an integer as input and checks if it is a prime number.

(m) Write a program that calculates the sum of the first *n* terms of an arithmetic series, where *n*, the first term and common difference of the series are entered by the user.

(n) Write a program to compute the sum of the first *n* terms of the following series

S =1-2+3-4+5……

(o) Write a program to create an array with inputs from the user and print the same.

(p) 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) Print the array in reverse order

(q) Write a program to take a matrix from the user and print the transpose of the same.

(r) Write a program to ask for the name of the user and print the same.

(s) Write a program to take a string of length more than 10 and find the number of vowels in the string. Also print the position of the vowels in the string.

(t) Write a program using pointers to copy a string to another string variable without using library function.

(u) Write a program that swaps two numbers using pointers.

(v) Write a program to calculate Factorial of a number (i) using recursion, (ii) using iteration

(w) 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.

(x) 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.

(y) Write a function to accept two arrays as argument and returns their sum as an array.

(z) 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.

(aa) 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.

(bb) Write a C program that opens a file for reading and displays the contents of the file in binary mode and text mode.

(cc) Write a C program that opens a file for reading and displays the contents of the file line by line on the screen.

(dd) Write a C program that opens a file in append mode and allows the user to add text to the end of the file.

Particulars of course designer:

Name : Risheraj Baruah
Contact No. : +91 8486942427
Email id : rishirajbaruah@gauhati.ac.in

Mathematical Foundation in Information Technology

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

3. Semester: 1

4. Course Type: Compulsory

5. Course Level: 100-199

6. Theory Credit: 4

7. Practical Credit: NIL

8. No of Hours:

a) Theory: 60 hrs (60classes)

b) Practical: NIL

c) Non Contact: NIL

9. List of Reference Books:

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

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.

Books

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

COMPUTER FUNDAMENTALS AND APPLICATION SOFTWARE

1. **Learning Outcome:**
   At the end of the course, students will be able to:
   Understand the basics of Computer System
   Understand the basic of Software
   Work with Word Processing Software
   Work with a Spreadsheet Software
   Work with a Presentation Software
2. **Prerequisite:** NIL
3. **Semester: 1**
4. **Course Type: SEC**
5. **Course Level: 100-199**
6. **Theory Credit: 1**
7. **Practical Credit: 2**
8. **Number of required hours:**
   a) Theory: 15 hrs
   b) Practical: 60 hrs
   c) Non Contact: 5 hrs

8. List of reference books:

a) V. Rajaraman, "Fundamentals of Computer", 4th Ed., PHI, 2006
b) R. Thareja, "Computer Fundamentals & Programming in C", Oxford University Press, 2013.

8. Detailed Syllabus:

Theory

Unit 1: Computer Fundamentals (8 hours)

Computer, Basic components of computer: Input unit, Output Unit and CPU, Memory – primary and secondary memory. Storage devices – magnetic storage devices: Magnetic Tape, Hard Disk, optical storage devices: CD, DVD, Input devices– mouse, keyboard, output devices – CRT and LCD monitors, printers: dot matrix printers, ink jet printers, laser printers.

Unit 2: Programming Basics (7 hours)

Introduction to programming languages. Low-level and high-level language and their characteristics. Compiler vs. interpreter. Software, application software, system software. Word processing software, Spreadsheet software, Presentation Software, Operating systems, functions of operating system, Open source software.

Laboratory

Unit 3: Word Processing Software (20 hours)

Opening and saving documents, Changing document views, Formatting text, Cutting, copying, and moving text, Finding and replacing text, Inserting special characters, Using AutoCorrect, Formatting paragraphs, Format painter, Pasting unformatted text, Creating and formatting tables, Creating and Formatting lists, Creating page headers and footers, displaying page numbers, Changing page margins, Adding a custom watermark to the page background, Adding page break, Adding comments to a document, Creating a table of contents,Working with images, Using Mail Merge, Macros, Linking and cross-referencing within a document, Using master documents, Shortcuts

Unit 4: Spreadsheet Software (30 hours)

Spreadsheets, sheets, and cells, Opening and saving spreadsheet files, Cell navigation, Sheet navigation, Inserting new sheets, Moving and copying sheets, Renaming sheets, Deleting sheets, Changing document

view, Freezing rows and columns, Splitting the screen, Insert Date and time in a cell, auto fill, Sharing content between sheets, Validating cell contents, Replacing data, Multiple lines of text and wrapping, Manual line breaks, Merging cells, Splitting cells, Formatting texts and number in cells, Setting cell borders, setting cell background, AutoFormatting of cells, Conditional formatting, Filtering data, Sorting records, Cell comments, Using formula, Functions: SUM, AVERAGE, MAX, MIN, COUNT,COUNTA, IF, AND, OR, CONCATENATE, LEFT, RIGHT, MID, TODAY, NOW, YEAR, MONTH, DAY, VLOOKUP, INDEX, MATCH, Pivot table, Inserting a page break, Headers and footers, Shortcuts, Column charts, Bar charts. Pie charts, Area charts, Line charts, Scatter, Bubble charts, Net charts

Unit 5: Presentation Software (10 hours)

Opening and saving files, Document views, Inserting a slide, Duplicate slide, Slide layouts, Adding and formatting text and image, Vertical text, Creating bulleted and numbered lists, Inserting images, tables, charts, or media, Inserting tables, OLE, Creating master slide, applying master slides, Adding comments, Printing handouts, starting slide show, hiding slides in slide show, Custom slide show, Setting slide transition effects, Slide element animation effects, Presenter Console

Particulars of course designer:

Name: Dr. Hasin A Ahmed
Contact No.: 8011810533
Mail id: hasin@gauhati.ac.in

Digital Logic Fundamentals

1. Learning Outcomes: After completing this course, students will have grasp of fundamental concepts of digital logic that will make their base to understand the concepts of computer architecture and organization.

2. Prerequisites: NIL

3. Semester: 2

4. Course type: Compulsory

5. Course level: 100-199

6. Theory credit: 4

7. Practical credit: 0

8. Number of required hours:

    a) Theory: 60 hrs (60 classes)

    b) Practical: NIL

    c) Non Contact: NIL

9. Reference books:

    (a) Digital Logic and Computer Design, M. Morris Mano, Pearson India

    (b) Digital Logic and Computer Organization, V. Rajaraman, T. Radhakrishnan, PHI Learning

10. Contents of Syllabus:

Unit I: Introduction to Binary Number System 10 hrs

Binary numbers, number base conversions, octal and hexa decimal numbers, 1's complement and 2's complement, representation of signed binary number: 1's complement, 2's complement and signed magnitude, subtraction with complements, arithmetic addition and subtraction of signed binary numbers, binary codes: BCD, Excess-3, error detection code: parity bit, error correction code: Hamming code, gray code, ASCII, EBCDIC, binary logic, logic gates: AND, OR, inverter, buffer, NAND, NOR, XOR and equivalence

Unit II: Boolean Algebra, Logic Gates and Integrated Circuits 15 hrs

Definition of boolean algebra, two valued boolean algebra, duality principle, theorems and postulates of boolean algebra, precedence of boolean operators, boolean expression and Venn diagram, boolean functions and truth tables, complement of a boolean function, minterms and maxterms, canonical forms of a boolean function, sum of minterms and its short notation, product of maxterms and its short notation, conversion between canonical forms, standard form of a boolean function, digital logic gates, integrated circuits and levels of integration, digital logic families

Unit III: Simplification of Boolean Functions 10 hrs

Map minimization method, two variable map, three variable maps, four variable map, five variable map, NAND and NOR implementation of boolean functions, don't-care conditions, tabulation method

Unit IV: Combinational Circuits 12 hrs

Definition of combinational circuit, design procedure, half adder, full adder, half subtractor, full subtractor, BCD-to-Excess-3 code converter, encoders and decoders, multiplexers, ROM

Unit V: Sequential circuits 13 hrs

Flip flops, RS flip flop, D flip flop, JK flip flop, T flip flop, master slave flip flops and edge triggered flip flops, state table of a sequential circuit, state diagram, characteristic tables of flip flops, Mealy and Moore machine, flip flop excitation tables, design procedure of clocked sequential circuit, 3-bit binary counter, shift register, ripple counter, RAM

Particulars of course designer:

Name: Dr. Hasin A Ahmed
Contact No.: 8011810533
Mail id: hasin@gauhati.ac.in

Data Structures & Algorithms Using C

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

(a) 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.

(b) Analyze the time and space complexity of different algorithms and choose the appropriate algorithm for a given problem.

(c) Develop efficient algorithms to solve various computational problems by utilizing data structures and algorithms covered in the course.

2. Prerequisites: NIL

3. Semester: 2

4. Course type: Compulsory

5. Course level: 100-199

6. Theory credit: 3

7. Practical credit: 1

8. Number of required hours:

a) Theory: 45 hrs (45 classes)

b) Practical: 30 hrs (15 classes)

c) Non Contact: NIL

9. Reference books:

(a) Weiss, Mark Allen. "Data Structures and Algorithm Analysis in C". 3rd ed., Pearson, 2012

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

(c) Goodrich, Michael T., and Roberto Tamassia. "Data Structures and Algorithms in C". 2nd ed., Wiley, 2011.

(d) Gilberg, Richard F., and Behrouz A. Forouzan. "Data Structures: A Pseudocode Approach with C". Narosa Publishing House, 2009.

10. Detailed 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. Array of structures and Structure of Arrays. Array of lists and List of lists.

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, multi-way search trees. Breadth First Search, Depth First Search. 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.

B. List of Practicals

(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.)

(a) 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.

(b) 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.

(c) 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.

(d) Write a program to implement doubly linked list.

(e) Write a function to concatenate two linked lists.
(f) Write a program to take a number k and split the linked list after k^{th} position.
(g) Write a program to merge two sorted linked lists.
(h) Write a program to implement list of lists.
(i) 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.
(j) 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.
(k) Write a program to evaluate a simple postfix expression using stack.
(l) Write a program to convert a decimal number into binary number using stack.
(m) 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.
(n) 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.
(o) Implement binary search and linear search algorithms on arrays.
(p) 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
(q) 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
(r) Implement following sorting algorithms:
    a. Bubble sort
    b. Insertion sort
    c. Selection sort
    d. Counting sort

Particulars of course designer

Name : Risheraj Baruah
Contact No. : +91 8486942427
Email id : rishirajbaruah@gauhati.ac.in

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. Prerequisites: NIL
3. Semester: 2
4. Course Type: Compulsory
5. Course Level: 100-199
6. Theory Credit: 2
7. Practical Credit: 1
8. No of Hours:

Theory: 30 hrs (30 classes)
Practical: 30 hrs (15 classes)
Non Contact: NIL

9. List of Reference 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.

10. Contents of Syllabus:

A. Theory

Unit 1: Introduction to Web Technologies (5 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 (7 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 (7 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 (6 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 (5 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.

Particulars of Course Designer:

Name : Risheraj Baruah
Contact No. : +91 8486942427
Email id : rishirajbaruah@gauhati.ac.in

Computer Organization and Architecture

1. **Learning Outcome:** Student will
   - be able to learn about the structure, function and characteristics of computer systems.
   - understand the design of the various functional units and components of computers.
   - identify the elements of modern instructions sets and their impact on processor design.
   - able to learn about the function of each element of a memory hierarchy.
   - able to learn about identify and compare different methods for computer I/O.
   - Student will able to learn about basics of assembly language.
2. **Prerequisite:** NIL
3. **Semester: 3**
4. **Course Type: Compulsory**
5. **Course Level: 200-299**
6. **Theory credit:** 4
7. **Practical credit:** 0
8. **Number of required hours:**
   (a) Theory: 60 hrs (60 classes)
   (b) Practical: NIL
   (c) Non Contact: NIL
9. **List of reference books:**

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

10. 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.

Particulars of course designer:

Name: Dr Irani Hazarika
Contact No: 8486965773
Email: queensarathi@gmail.com

Operating system

1. **Learning Outcome:** After completing this course, students will have understanding of the internal structure and usage of various components related to an operating system.
2. **Prerequisite:** NIL
3. **Semester: 3**
4. **Course Type: Compulsory**
5. **Course Level: 200-299**
6. **Theory credit:** 3
7. **Practical credit:** 1
8. **Number of required hours:**
   Theory: 45 hrs (45 classes)
   Practical: 30 hrs (15 classes)
   Non Contact: NIL
9. **List of books:**
   a) Operating System Concepts, Abraham Silberschatz, Peter B. Galvin, Greg Gagne, Wiley
   b) Modern Operating Systems, Andrew S. Tanenbaum, Prentice-Hall Of India Pvt. Limited
10. **Detailed 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, multi tasking operating system, distributed operating system, real time operating system, multi user operating system, major functions of operating system: Process Management, Process Synchronization, Memory Management, CPU Scheduling, File Management, I/O Management, Security, virtualization, cloud computing, open source operating system, history of operating system, 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, remote procedure 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, scheduling algorithms: First-Come, First-Served Scheduling, Shortest-Job-First Scheduling, Priority Scheduling, Round-Robin Scheduling, Multilevel Queue Scheduling, Multilevel Feedback Queue Scheduling

Unit III: Process synchronization **8 hrs**

Race condition, critical section problem, Peterson's algorithm, Bakery algorithm, synchronization hardware: 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, no preemption and circular wait, resource allocation graph, deadlock prevention: definition, preventing mutual exclusion, preventing hold & wait, preventing no preemption and preventing circular wait, deadlock avoidance: definition, safe state, safe sequence, resource allocation graph based algorithm and Banker's algorithm, deadlock detection: definition, wait-for graph, algorithm to detect deadlock for single instance resources, algorithm to detect deadlock for multiple instance resources 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 address spaces, 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 simple segmentation, internal and external fragmentation, TLB, virtual memory, demand paging, page fault, locality of reference principle, performance of demand paging, page replacement algorithms: FIFO, Optimal and LRU, allocation of frames: equal allocation and proportional allocation, global and local page replacement algorithms, thrashing

B. Practical

- **Basic linux commands**: pwd, ls, cd, mkdir, rmdir, rm, touch, man, cp, mv, locate, head, tail (***2 Classes/4 hrs***)
- **Advanced 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 (***3 Classes/6 hrs***)
- **Shell scripting in linux:** shell, types of shell, shell script, echo command, shell variables, 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/14 hrs***)
- **Using system calls in C program in linux**: fork(), exec(), exit(), getpid(), mkdir(), rmdir() etc. (***3 Classes/6 hrs***)

Particulars of course designer:

Name: Dr. Hasin Afzal Ahmed
Contact No.: 8011810533
E-mail id: hasin@gauhati.ac.in

Object Oriented Programming through C++

1. **Learning Outcome:** After successful completion of this course, students will be able to:
   - Will be able to imagine real-life concepts as objects; derive their properties and functions to operate.
   - Develop programs using object- oriented features like data abstraction, polymorphism, inheritance, exception handling.
   - Know C++ streams, operators
   - Know file handling techniques in C++.

2. **Prerequisite:** NIL

3. **Semester: 3**

4. **Course Type: Compulsory**

5. **Course Level: 200-299**

6. **Theory credit:** 3

7. **Practical credit:** 1

8. **Number of required hours:**
   1. Theory: 45 hrs (45 classes)
   2. Practical: 30 hrs (15 classes)
   3. Non Contact: NIL

9. **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.

10. **Detailed Syllabus:**

a) Theory Content

UNIT 1: Introduction to object oriented programming (10 Lectures)

Origins of C++, Basic Concepts of Object Oriented Programming, Benefits of OOP, Applications of OOP, Introduction to C++, 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, Constant arguments.

UNIT 2: Classes and objects (10 Lectures)

Introduction - Defining a class-Class Vs structures, Creating objects, Accessing class members, Defining member functions- Outside the class definition, 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, Constructors and destructors, Basic Concepts of constructors, Default constructor, Parameterized constructor, Multiple constructors in a class, Constructor with default arguments, Dynamic initialization of objects, Copy constructor, Dynamic constructors, Destructors\

UNIT 3: Function and operator overloading (9 Lectures)

Overloading Concepts Function Overloading: Functions with different sets of parameters, default andconstant parameters, Rules for overloading operators. Defining operator overloading, OverloadingUnary operators, Prefix and Postfix operators overloading, Overloading Binary operators, overloadingrelational operators, Overloading using friend functions, Overloading subscript operator, Pitfalls ofoperator overloading, Type conversion-Basic to Class, Class to Basic

UNIT 4: Inheritance (8 Lectures)

Introduction-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 Runtimepolymorphism, Pointers to objects, this pointer, Pointer to derived classes, Virtual functions, Rules forvirtual functions, Pure virtual functions.

UNIT 5: Streams (4 Lectures)

C++ stream classes-put() and get() functions, getline() and write() functions, Overloading << and >>operators, Formatted Console I/O operations, ios class functions-width(),precision(),fill(),setf() andunsetf(), Formatting flags, Manipulators, User defined manipulators.

UNIT 6: Files (4 Lectures)

Introduction-Stream classes for files, Opening files using constructor, Opening files using open(), Filemodes, Detecting end of file-eof(), Sequential input and output-put() and get()-Reading and writingobjects-read() and write()-Random Access files-Manipulating file.

b) Practical / Lab work to be performed

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 write member 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 a circle. Write member functions area() and perimeter() to compute the area and perimeter of a circle.
4. Define a class *complex* with two data members *real* and *imag*to represent real and imaginary parts 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. (*This is an example of polymorphism.*)

5. Define a class *point* with two data members *xordinate*and *yordinate*to represent all points in the two dimensional plane by storing their x co-ordinate and y co-ordinate values. Write member 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 friend function to compute the distance between two points.

6. Define a class *string* with the following data members char *p; int size; and write member functions to do the following (without using library function) and using dynamic memory allocation.
- 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 usual sense. Also overload the unary – operator.

8. For the class *string* defined in 6 above, overload the <<, >> and + operators where + is to be used for concatenating two strings.

9. Define a class *time* to store time as hour, minute and second, all being integer values. Write member functions to display time in standard formats. Also overload the ++ and – operators to increase and decrease a given time by one second where the minute and hour values will have to be updated whenever necessary.

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

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

12. 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 artsst class will have additional data members *ph, hs, en*and as to store marks obtained by a student in three subjects Philosophy, 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 to calculate the total marks obtained by a student
takedata() : function to accept values of the data members
showdata(): function to display the marks sheet of a student .

13. 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 articular printer. Write pure virtual function display() in the base class and redefine it in the derived classes.

14. Define a abstract base class *figure* and add to it pure virtual functions. Derive three classes *circle, rectangle* and t*riangle* 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.\

15. Write an interactive program to compute square root of a number. The input value must be tested for validity. If it is negative, the user defined function *my_sqrt()* should raise an exception.
16. Define a class *rational* to store rational numbers as a pair of integers, representing the numerator and denominator. Write a member function for setting the values of the numerator and denominator. This function should raise an exception if attempt is made to set a zero value as the denominator and in such cases it should be set to 1.
17. Write a class template for storing an array of elements. Overload the << and >> operators. Write a member function to sort the array in descending order.
18. Write a class template for representing a singly linked list. Write functions for inserting, deleting, searching and for displaying a linked list. Write a main function to test it on a linked-list of integers and characters.

Particulars of course designer:

Name: Prof. Anjana Kakoti Mahanta
Contact No.: 9864425716
E-mail id : anjana@gauhati.ac.in

Advanced Web Programming

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

(a) Design basic well-structured web page using HTML and CSS
(b) Develop the ability to implement interactive elements and dynamic content using basic JavaScript
(c) Develop a foundational understanding of server-side scripting using PHP
(d) Create a CRUD web application using HTML, CSS, JavaScript, PHP and MySQL.

2. Prerequisites: NIL

3. Semester:3

4. Course Type: Compulsory

5. Course Level: 300-399

6. Theory Credit: 3

7. Practical Credit: 1

8. No of Hours:

a) Theory: 45 hrs (45 classes)
b) Practical: 30 hrs (15 classes)
c) Non Contact: NIL

9. List of Reference Books:

(a) Duckett, J. (2011). HTML and CSS: Design and Build Websites. John Wiley & Sons.
(b) Robbins, J. N. (2018). Learning Web Design: A Beginner's Guide. O'Reilly Media.
(c) Nixon, R. (2014). Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5 (4th ed.). O'Reilly Media.
(d) Duckett, J. (2014). JavaScript and JQuery: Interactive Front-End Web Development. John Wiley & Sons.
(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.

10. Contents of Syllabus:

A. Theory

Unit 1: Advanced HTML (6 Lectures)

Review of basic HTML tags and their usage. Working with forms – validation using HTML5 attributes. HTML5 Semantic Elements – header, nav, section, article, aside, footer. Applying proper semantic markup for improved SEO. Multimedia integration. Embedding images with different attributes. Adding video and audio. Meta information and Document Structure – metadata, viewport settings.

Unit 2: Advanced Design with CSS (12 Lectures)

Review of CSS. CSS Selectors. Specificity and the cascade. Psudo-classes and pseudo-elements.CSS Box sizing. Gradient and Transparent backgrounds. CSS Typography – Line height and letter spacing. Web-safe fonts. CSS Layout. Display property – inline, block, inline-block, none. Positioning – static, relative, absolute, fixed. Floats and clear property. Box alignment – flexbox and grid layout.Responsive Web Design – Media queries and breakpoints. Fluid layouts. Brief concept of CSS preprocessors – Sass, Less. Brief concept of CSS frameworks – Bootstrap, Tailwind.

Unit 3: Advanced JavaScript (12 Lectures)

Review of JavaScript concepts. Functions in JavaScript. Lexical Environment. Arrays and Array manipulation in JavaScript.. JavaScript Events and Event Handling – Event propagation and event delegation. Implementing interactivity with user actions. Introduction to JavaScript APIs. Callback functions and event loop. Promise chain. Asynchronous function with async/wait. DOM manipulation and event handling with jQuery. Overview of AJAX. Brief concept of XMLHttpRequest object.

Unit 4: Server-Side Scripting using PHP (10 Lectures)

Review of PHP as a server-side scripting language. Handling forms and user input with PHP. Interacting with databases and performing CRUD operations using PHP and MySQL. User authentication using PHP. Implementing user registration and login functionality. Session management and Token based authentication. Overview of Cookies and their use in Web applications. Working with cookies in PHP – setting, reading, deleting. Concept of Cross-site scripting (XSS).

Unit 5: Advanced Concepts of Web Programming (5 Lectures)

Overview of web hosting – shared hosting, VPS, dedicated hosting, cloud hosting. Overview of Server-Side Includes (SSI). Brief concepts of Web APIs and data integration. Concept of JavaScript frameworks – React.js and Node.js. Version Control Systems. Brief overview of Continuous Integration and Deployment. Overview of Web security and SSL/TLS. Web analytics and monitoring.

B. List of Practical

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

1) Create a semantic HTML structure for a blog post, including headings, paragraphs, images, and nested elements.
2) Develop an HTML5 video player with custom controls, including play, pause, volume control, and full-screen functionality.
3) Create a responsive HTML layout using CSS Grid or Flexbox that adapts to different screen sizes and orientations.
4) Develop a responsive navigation menu that collapses into a hamburger menu for mobile devices, utilizing media queries and CSS transitions.
5) Implement a CSS animation or transition to create a smooth fade-in effect for an element on page load.
6) Design a CSS grid layout that displays a multi-column card-based UI, where each card has a consistent height but variable width. Each card should display an image, title, and description.
7) Develop a CSS-only tooltip that appears when hovering over an element, with customizable styles and positioning.
8) Design a CSS drop-down menu with multiple levels of nested submenus, allowing users to navigate through the menu hierarchy.
9) Create a CSS layout that implements a sticky header, where the header remains fixed at the top of the page while the content scrolls.
10) Build a responsive landing page using HTML5, including a hero section, feature sections, and a contact form.
11) Implement a CSS grid-based layout for a product catalog, showcasing multiple products with consistent spacing and alignment.

12) Implement a custom dropdown menu using HTML, CSS, and JavaScript, with options that can be selected and displayed.
13) Build a form validation mechanism using HTML5 form validation attributes and JavaScript, ensuring that required fields are filled out correctly. Use CSS to design the form and the validation messages.
14) Develop a slideshow or carousel using JavaScript and the DOM API, with next/previous controls and automatic playback.
15) Implement a dynamic table that allows users to add or remove rows, with the ability to edit and delete individual cells.
16) Develop a live search functionality that filters and displays search results from the content of the web page in real-time as the user types, using JavaScript and DOM manipulation.
17) Use a callback function to perform an asynchronous AJAX request and update the content of a specific HTML element with the response.
18) Implement a callback-based timer that executes a specific function after a certain period of time has elapsed.
19) Create a simple asynchronous form submission process using AJAX, displaying a loading spinner while waiting for the response.
20) Develop a weather application that uses an asynchronous API call to fetch weather data based on user input, displaying the results on the page.
21) Implement a user registration form in PHP, which securely stores user credentials in a database and performs validation checks for email uniqueness and password strength.
22) Create a login page in PHP that verifies user credentials against the stored data in the database and redirects authenticated users to a secure dashboard.
23) Develop a Password reset functionality in PHP, allowing users to request a password reset link via email and securely update their password.
24) Implement a user profile page in PHP, which displays and allows users to edit their personal information such as name, email, and profile picture.
25) Create a session-based shopping cart system in PHP, allowing users to add products, update quantities, and remove items, while maintaining cart information across different pages.
26) Develop an access control system in PHP, where certain pages or features are restricted to logged-in users only and unauthorized users are redirected to a login page.
27) Implement user roles and permissions in PHP, allowing administrators to assign different levels of access to users based on their roles (e.g., admin, moderator, user).
28) Create a "Remember Me" functionality in PHP, using cookies to remember and automatically log in returning users for a certain period of time.
29) Develop a logout mechanism in PHP that destroys the user session and redirects users to a logout confirmation page or the login page.
30) Implement account activation via email in PHP, where new users receive an activation link to verify their email address and activate their account.

Particulars of course designer:

Name : Risheraj Baruah
Contact No. : +91 8486942427
Email id : rishirajbaruah@gauhati.ac.in

Database Management System

1. **Learning Outcome:** On successful completion of this course, the student should be able to:
    a. Learn database concepts and its architectural components.
    b. Describe different data models used for designing a database.
    c. To create a database using relational models and entity relationships concepts
    d. Normalize a database into various normal forms
    e. Design SQL queries to handle a relational database.
2. **Prerequisite:** NIL
3. **Semester: 4**
4. **Course Type: Compulsory**
5. **Course Level: 200-299**
6. **Theory Credit:** 3
7. **Practical Credit:** 1
8. **Number of required hours:**
    a. Theory: 45 hrs (45 classes)
    b. Practical: 30 hrs (15 classes)
    c. Non Contact: NIL

9. **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

10. **Detailed Syllabus:**

A. Theory

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

UNIT 4: 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()*

B. Lab Contents: (30 hrs)

Practical / Lab work to be performed:

a) Implementation of SQL DDL statements in MySQL DBMS: CREATE DATABASE, CREATE TABLE, ALTER TABLE, RENAME, DROP DATABASE/TABLE

b) Use of SQL DML statements in MySQL DBMS: INSERT, SELECT, UPDATE, DELETE SQL commands

c) Implementing following constraints in MySQL DBMS: PRIMARY KEY, FOREIGN KEY, NOT NULL, UNIQUE and DEFAULT

d) Handling following SQL clauses in MySQL DBMS: WHERE, GROUP BY, ORDER BY, HAVING, IN, BETWEEN, LIKE

e) Working with following aggregate functions in MySQL DBMS: COUNT, AVG, MAX, MIN and SUM

f) Working with transaction processing command in MySQL DBMS: START TRANSACTION, COMMIT and ROLLBACK Statements, SET autocommit

Particulars of course designer:

Name : Dwipen Laskar
Contact No : +916000795681
Email-id : laskardwipen@gauhati.ac.in

Computer Oriented Numerical and Statistical Methods

1. **Learning Outcome**: On successful completion of this course, the student should be able to:
    - Learn the properties of Floating Point, Numbers and their accuracy, approximations and errors
    - Learn various probability methods, Interpolation methods etc.
    - To solve basic problems in probability and statistics

2. Prerequisite: NIL

3. Semester: 4

4. Course Type: Compulsory

5**. Course Level**: 300-399

6. Theory credit: 3

7. Practical credit: 1

8. Number of required hours: a) Theory: 45 hrs

b) Practical: 30 hrs

c) Non Contact: 5 hrs

9. List of books:

a) Rajaraman, V, "Computer Oriented Numerical Methods", 3rd edition, Prentice Hall
b) Balaguruswami, E., "Computer Oriented Statistical and Numerical Methods", Macmillan Publishers India Limited

10. Detailed Syllabus:

a) Theory

UNIT-I: Introduction to Computer Arithmetic (8 hrs)

Representation of numbers: Fixed Point and Floating point representations, Normalized Floating Representation, Floating Point Arithmetic, Properties of Floating Point, Numbers and their accuracy, Approximations and errors. Errors: truncation error, rounded off error, absolute error, relative error, percentage error and error propagation

UNIT-II: Algebraic and Transcendental Equations (10 hrs)

Introduction to linear and nonlinear equations, measures of accuracy, Properties of polynomial equations, Initial approximation to a root, Solution of algebraic/transcendental equations: Bisection Method, Iteration method, Method of false position, Newton-Raphson method, Rate of convergence of Iterative methods, Solution of simultaneous linear equations by using Gauss elimination method

UNIT-III: Interpolation (8 hrs)

Polynomial Interpolation, Finite Differences, Newton's Forward Difference Interpolation, Newton's Backward Difference Interpolation, Newton's Divided Difference Interpolation

UNIT-IV: Solution of Differential Equation (6 hrs)

Taylor series method, Euler's method, Runge-Kutta method of 1st, 2nd & 4th order.

UNIT-V: Descriptive Statistics (8 hrs)

Types of Data, Attributes and Variables, Construction of Frequency, Cumulative frequency, Graphical Representation of Frequency distribution: Histogram, Frequency Polygon, Frequency Curve and Cumulative Frequency Curves (Ogive curves), Diagrammatic Representations: Simple bar, Subdivided bar, Pie Diagrams

UNIT-VI: Measure of central tendency (5 hrs)

Measure of central tendency-Mean, Median and Mode. Measure of variation-Range, Interquartile range, Standard Deviation and Variance

b) **Lab Content:** Practical / Lab work to be performed using C/C++/Java programming Language: Apply the Bi-section method for approximation of root for a given polynomial equation.

- Apply the False Position method for approximation of root for a given polynomial equation
- Implement Newton Raphson method for approximation of root for a given polynomial equation.
- Implement Gauss elimination method to solve simultaneous linear equations
- Develop programs to implement Newton's Forward Difference Interpolation
- Develop programs to implement Newton's Backward Difference Interpolation
- Develop programs to implement Newton's Divided Difference Interpolation
- Develop program to apply Taylor's series for e raise to the power x
- Implement Euler's method for solving a differential equation
- Implement Runge-Kutta method of 1st, 2nd & 4th order for solving a differential equation
- Write programs to find Mean, Median and Mode for a given set of data

Particulars of course designer:
Name : Dwipen Laskar
Contact No : +916000795681
Email-id :laskardwipen@gauhati.ac.in

Automata Theory and Languages

1. Learning Outcome: After completing this course, students will

- 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. Prerequisites: NIL

3. Semester: 4

4. Course Type: Compulsory

5. Course Level: 200-299

6. Theory Credit: 4

7. Practical Credit: 0

8. No of Hours:

a) Theory: 60 hrs (60 classes)
b) Practical: NIL
c) Non Contact: NIL

9. 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.

10. Contents of 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 Regularlanguages 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 ingrammars 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 free Languages.

Particulars of course designer:

Name: Prof. Anjana Kakoti Mahanta
Contact No.: 9864425716
E-mail id : anjana@gauhati.ac.in

Python Programming

1. Learning Outcome: After completing this course, students will know about fundamentals of Python Programming and Problem Solving.
2. Prerequisites: NIL
3. Semester: 4
4. Course Type: Compulsory
5. Course Level: 200-299
6. Theory Credit: 3
7. Practical Credit: 1
8. No of Hours:

- Theory: 45 hrs (45 classes)
- Practical: 30 hrs (15 classes)
- Non Contact: NIL

9. List of Reference Books and Materials:

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

10. Contents of Syllabus:
(a) Theory

Unit 1: Introduction to Python Programming (8 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 (8 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
Creating Classes in Python, Instance Methods, Inheritance, Polymorphism, Exception Classes and Custom Exceptions.

Unit 7: Introduction to Libraries in Python (6 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

Particulars of course designer:

Name: Dr. Sanjib Kr Kalita
Contact No.: 8812051150
E-mail id: sanjib959@gauhati.ac.in

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. Prerequisites: NIL

3. Semester: 4

4. Course Type: Compulsory

5. Course Level: 200-299

6. Theory Credit: 4

7. Practical Credit: NIL

8. No of Hours:

a) Theory: 60 hrs (60classes)

b) Practical: NIL

c) Non Contact: NIL

9. List of Reference Books:

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

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 andDeletion 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.

System Software

1. Learning Outcome: After completing this course, students will have understanding of various types of system software.

2. Prerequisites: NIL

3. Semester: 6

4. Course Type: Compulsory

5. Course Level: 300-399

6. Theory Credit: 3

7. Practical Credit: 1

8. No of Hours:

Theory: 45 hrs (45 classes)

Practical: 30 hrs (15 classes)

Non Contact: NIL

9. List of Books:

a) System Software : An Introduction to Systems Programming, Leland L. Beck, D. Manjula, Pearson

b) Systems Programming, Dhananjay Dhamdhere, McGraw Hill Education

10. Contents of Syllabus:

(a) Theory

Unit I: Introduction to Operating System 10 hours

Types of software, Application software and system software, examples of system software, system programming, system software and machine architecture, the simplified instructional computer (SIC): *memory, registers, data formats, instruction formats, addressing modes, instruction set, input and output,* programming examples in SIC

Unit II: Assemblers 12 hours

Assembler definition, basic assembler functions, assembler algorithm and data structure, handling instruction formats and addressing modes, program relocation, handling literals, symbol defining statements, expressions, assembler design options: one pass assemblers and multi pass assemblers, introduction to NASM assembler

Unit III: Loaders and Linkers 7 hours

Loading, relocation and linking, loader, absolute loader, bootstrap loader, relocating loader, program linking, linking loader, linkage editor, static and dynamic linking

Unit IV: Macro processor 6 hours

Definition of macro processor, macro definition and expansion, macro processor algorithm and data structures, conditional macro expansion, general purpose macro processors, macro processing within language translators

Unit V: Compilers 10 hours

Compiler definition, grammars, lexical analysis, syntactic analysis, operator precedence parsing, recursive descent parsing, code generation, intermediate form, code optimization: machine

dependent and machine independent, interpreter

(b) Practical

1) Introduction to NASM assembler (1 class/2 hrs)
2) Introduction to segments and registers (1 class/2 hrs)
3) A simple assembly program to print hello (1 class/2 hrs)
4) Input and output in assembly language (1 class/2 hrs)
5) Conditional statements in assembly language (2 classes/4 hrs)
6) Looping in assembly language (3 classes/6 hrs)
7) An assembly language program that accepts two numbers from the user and displays sum of the numbers (1 class/2 hrs)
8) An assembly language program that changes case of accepted characters (1 class/2 hrs)
9) An assembly program that accepts a number and displays whether the number is odd or even (1 class/2 hrs)
10) An assembly program that accepts a number n from the user and displays “hello world” n number of times. (1 class/2 hrs)
11) An assembly program that accepts a number from the user and displays factorial of the number (1 class/2 hrs)
12) An assembly program that accepts a number n from the user and displays whether the number is prime (1 class/2 hrs)

Particulars of course designer:

Name: Dr. Hasin Afzal Ahmed
Contact No.: 8011810533
E-mail id: hasin@gauhati.ac.in

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

Theory: 60 hrs (60 classes)
Practical: 0 hrs
Non Contact: NIL

9. List of Books:

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

10. Contents of 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.

Particulars of course designer:

Name : Dwipen Laskar
Contact No : +916000795681
Email-id : laskardwipen@gauhati.ac.in

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

a) Theory: 45 hrs (45 classes)
b) Practical: 30 hrs (15 classes)
c) Non Contact: NIL

9. List of Reference Books:

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

10. Contents of Syllabus:
A. Theory

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

Particulars of course designer:

Name: Dr. Hasin Afzal Ahmed
Contact No.: 8011810533
E-mail id: hasin@gauhati.ac.in

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

a) Theory: 45 hrs (45 classes)
b) Practical: 30 hrs (15 classes)
c) Non Contact: NIL

9. 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.

10. Contents of Syllabus:

A. Theory

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/30 hrs)

- 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.
- 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

Particulars of Course Designer:

Name: Dr Irani Hazarika
Contact No: 8486965773
Email: queensarathi@gmail.com

Information Security and Cyber Laws

1. Learning Outcome:
After the completion of the course, the students will be able to develop basic understanding of security, cryptography, system attack and defences against them.

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

a) Theory: 60 hrs (60 classes)
b) Practical: NIL
c) Non Contact: NIL

9. List of Books:

(a) Merkow, M., & Breithaupt, J.(2005) Information Security Principles and Practices. 5th edition. Prentice Hall.
(b) Cryptography And Network Security Principles And Practice, Fourth or Fifth Edition, William Stallings, Pearson Edition.
(c) Cyber Law & Cyber Crimes, Advocat Prashant Mali; Snow White publications, Mumbai
(d) The Information Technology Act, 2000; Bare Act – Professional Book Publishers, New Delhi

10. Contents of Syllabus:

UNIT 1: Introduction (15 Lectures)

Basic components of security (Confidentiality, Integrity and Availability), Attacks, Computer Crime, Security Services, Security Mechanism, Cyber Crimes, information Technology ACT, Cryptography, Substitution Cipher, Transposition Cipher, Block Cipher, Stream Cipher, Confusion, Diffusion, Symmetric Key, Asymmetric Key, Encryption, DES Algorithm, Hash Function, Digital Signature, Digital Certificate.

UNIT 2: Program Security (10 Lectures)

Program Security, Program Errors, Buffer Oveflow, Incomplete mediation, Time-of-check to Time-of- use Errors, Malicious codes, Virus, Threats, Control against Programs, Program Security Issues. Protection in OS: Memory and Address protection, Access control, File protection, User Authentication.

UNIT 3: Database Security (10 Lectures)

Reliability, Integrity, Sensitive Data, Inference, Multilevel Security,Issues regarding the right to access information: Protecting Data, Multiple security level and categorization of data and users, Loss of integrity, Loss of availability, Loss of confidentiality, Access control, Inference control, flow control, data encryption

UNIT 4: Security in Networks (Cyber Attack) (15 Lectures)

Threats in Networks, Security Controls- Architecture, Encryption, Content Integrity, Strong Authentication, Firewalls: Design and Types of Firewalls, Intrusion Detection System, Secure Email, Denial-of-service attacks, Man in the middle Attack, Phishing, Spoofing and Spam Attacks, Drive-by attack, SQL Injection, Birthday attack, Social Engineering attack, Password Attack. Cross site scripting Attack, Malware Attack, Administering Security, Security Planning, Risk Analysis, Organisational Security Policy, Web Servers and Browsers, HTTP, Cookies, Caching, Secure Socket Layer (SSL), Secure Electronic Transaction (SET), E-mail Risks, Spam, E-mail Protocols, Simple Mail Transfer

Protocol (SMTP), Post office Protocol (POP), Internet Access Message protocol (ICMP), Secured Mail: Pretty Good Privacy (PGP), S/MIME (Secure/Multipurpose Internet Mail Extensions)

UNIT 5: Cyber Laws (10 Lectures)

Cyber crime, Types of crimes, Information technology Act 2000: Salient Feature of IT Act 2000, various authorities under IT Act and their powers, Penalties & Offences, amendments, Sections under the Information Technology Act such as:

- [Section 43] Penalty and compensation for damage to computer etc.
- [Section 65] Penalty for temping with the computers source documents
- [Section 66] Punishment for hacking with computer system, data alteration etc
- [Section 66A] Punishment for sending offensive messages through any communication services
- [Section 66B] Receiving stolen computer's resources or communication devices dishonestly
- [Section 66C] Punishment for identity theft
- [Section 66D] Punishment for cheating by impersonation by using computer resource
- [Section 66E] Punishment for violation of privacy
- [Section 66F] Punishment for cyber terrorism
- [Section 67] Punishment for publishing or transmitting obscene material in electronic form
- [Section 67A] Punishment for publishing or transmitting of material containing sexually explicit act, etc. in electronic form
- [Section 67B] Punishment for publishing or transmitting of material depicting children in sexually explicit act, etc. in electronic form
- [Section 72] Breach of confidentiality and privacy

Particulars of course designer:

Name : Dr. Pranamika Kakati
Contact No.: 9864201965
E-mail id: pranamikakakati20@gmail.com

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. Prerequisites: NIL

3. Semester: 6

4. Course Type: Elective

5. Course Level: 300-399

6. Theory Credit: 3

7. Practical Credit: 1

8. No of Hours:

a) Theory: 45 hrs (45 classes)
b) Practical: 30 hrs (15 classes)
c) Non Contact: NIL

9. List of Reference 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 & Practice 2nd 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.

10. Contents of Syllabus:

A. Theory

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 Scaling Matrix 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-Sutherland algorithm, Area clipping, Text clipping, Polygon clipping. 3-D concepts: Display methods-Parallel projection, perspective projection 3-D geometric transformations: Transformation, Translation, Rotation and Scaling around axes, 3-D Viewing Projections – Parallel and Perspective.

UNIT 4: Geometric Modelling **(8 Hours)**

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 **(5 Hours)**

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 **(5 Hours)**

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

B. 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.

Particulars of course designer:

Name: Dr. Diganta Kumar Pathak
Contact No.: 9707737222
E-mail id: digantakumarpathak@gauhati.ac.in

Optimization Techniques

1. Learning Outcome:
On successful completion of the course, students will be able to get thorough knowledge on formulation of optimization model and solution methods on optimization.

2. Prerequisites: NIL

3. Semester: 6

4. Course Type: Elective

5. Course Level: 300-399

6. Theory Credit: 4

7. Practical Credit: 0

8. No of Hours:

a) Theory: 60 hrs (60 classes)
b) Practical: NIL
c) Non Contact: NIL

9. List of Reference Books:

a) Rao S.S,"Optimization – Theory and applications", Wiley Easter Ltd., 1979

b) Cordan C.C. Beveridge and Robert S. Schedther, "Optimization, Theory and Practice" McGraw Hill Co.1970.

c) HarndyA.Tahh. "Operations Research, An Introduction", Macmillan Publishers Co.NewYork,1982

10. Contents of Syllabus:

UNIT 1: Introduction (5 Lectures)

Concept of Optimization- classification of optimization –problems, Simulation of Models, Art of Modeling.

UNIT 2: Modelling with Linear Programming (10 Lectures)

Linear Programming Model, Two variable LP Model, Types of Formulation of Simplex Method, Dual Simplex Method, Sensitivity Analysis, LP Model in Equation Form, Transportation Problem, Network Model, Minimal Spanning Tree Algorithm, Shortest route Problem. Necessary and sufficient conditions for finding extrema point, Bisection method.

UNIT 3: Queuing Theory (10 Lectures)

Queuing Model, Elements of Queuing Model, Pure Birth and Death Models, Queues with combined arrival and departures, random and series queues, Generalized and Specialized Queing Models.

UNIT 4: Unconstrained Optimization (10 Lectures)

Newton and Quasi-Newton methods, Conjugate gradient methods, Linesearch and Trust Region methods Quadratic programming problem-Wolfe's method & Beale's method.

UNIT 5: Constrained Optimization (10 Lectures)

Linear programming, Equality and inequality linear constraints. Barrier and augmented Lagrangian methods, Sequential quadratic programming, Infeasible start Newton method, Interior-point methods (inequality constrained minimization; barrier method; primal-dual interior point method Goal Programming-Basics of goal programming, goal programming formulation, goal programming algorithms: Weights method & preemptive method, Graphical solution

Particulars of course designer:

Name : Dr. Pranamika Kakati
Contact No.: 9864201965
E-mail id: pranamikakakati20@gmail.com

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 and able to apply basic principles of AI in solutions that require problem solving, inference, perception, knowledge representation, and learning.

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

a) Theory: 45 hrs (45 classes)
b) Practical: 30 hrs (15 classes)
c) Non Contact: NIL

9. List of Reference Books:

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

(b) Russell & Norvig, Artificial Intelligence-A Modern Approach, LPE, Pearson Prentice 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.

10. Contents of Syllabus:
A. Theory

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 (6 Hours)
Truth Maintenance System, Default Reasoning, Probabilistic Reasoning, Bayesian Probabilistic Inference, Possible World Representations.

UNIT 5: Understanding Natural Languages (5 Hours)

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

B. 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 at all 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 all levels.
- 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 the greater of two numbers X and Y.
- Write a Prolog program to implement maxlist(List,Max) so that Max is the greatest number in the list of numbers List.
- Write a Prolog program to implement sumlist(List,Sum) so that Sum is the sum 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 odd length respectively.
- Write a Prolog program to implement reverse (List, Reversed List) that reverses lists.
- Write a Prolog program to implement maxlist (List, Max) so that Max is the greatest 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/Frame Structures.

Particulars of course designer:

Name: Dr. Diganta Kumar Pathak
Contact No.: 9707737222
E-mail id: digantakumarpathak@gauhati.ac.in

Mobile Application Development

1. Learning Outcome: After completing this course, students will know:

(a) Fundamentals of Mobile Application Development.

(b) Difference between Native and Cross Platform Applications. Pros and Cons of Each Approach.

(c) To Design and Build a Complete Native Android Application with Both UI and Backend.

(d) To Design and Build a Complete Cross Platform Application with Both UI and Backend using Flutter.

2. Prerequisites: NIL

3. Semester: 6

4. Course Type: Elective

5. Course Level: 300-399

6. Theory Credit: 3

7. Practical Credit: 1

8. No of Hours:

a) Theory: 45 hrs (45 classes)
b) Practical: 30 hrs (15 classes)
c) Non Contact: NIL

9. List of Reference Books and Materials:

i. Android Programming: The Big Nerd Ranch Guide - Bill Phillips, Chris Stewart, Kristin Marsicano, Brian Gardner
ii. Professional Android - Reto Meier, Ian Lake
iii. Android Documentation - https://developer.android.com/

10. Contents of Syllabus:

a) Theory

Unit I: Introduction to Mobile Application Development **5 hrs**

Fundamentals of Mobile Application – Understanding Mobile Application Development Basics, Major Mobile OSs and their market share, Understanding Cross Platform and Native Application Development, The advantages and disadvantages of each approach, Components of a Mobile Application, Basic Design Principles of Mobile UI including Wireframing, Typography and Content Flow.

Unit II: Getting Familiar with Android **5 hrs**

Introduction to Android Operating System. History and Versions of Android. Understanding the Basics of Android Operating System including OS architecture, Anatomy of an Android Application(apk), learning about various approaches of Android Application development like Native Application Development using Java/Kotlin or Cross Platform Application Development with Flutter/React-Native/Ionic etc. In-depth understanding of each approach and their pros and cons.

Unit III: Getting Started with Native Android Application Development **10 hrs**

Setting up Android Studio and getting familiarized with the IDE, Setting up JDK and Android Emulator, Creating the First App – Hello World App, Understanding various essential folders and

files associated with an Android App stored inside*manifests, java* and *res* directories. Basic understanding about *Gradle*.Running the App for the first time. Getting started with USB Debugging at a physical Android Device. Understanding debugging facilities available with Android Studio. Getting Started with XML for Android UI Design, Learning Various UI Components of Android. Working with various UI resources like Images, Colors, Fonts etc. Creating a UI oriented App from Scratch. Working withlayout switching in Portrait and Landscape mode. Providing functionality to an Android Application using Java, Understanding Android *Activity* and its lifecycle, various events associated with Activity Lifecycle – *onCreate()*, *onStart()*, *onPause()*, *onResume()*, *onStop()*, *onRestart()*, *onDestroy()* etc. Broadcast Receivers, Intent and Filters. Advanced Layouts in Android including ListView, CardView, RecyclerView etc., Fragments, Material Design in Android – Principles and Implementation, Styles and Themes.

Unit IV: Advanced Android Application Development 10 hrs

Working with System Components in Android – File System Access, Location based Services, Phone, SMS, Bluetooth, Camera, Sensors etc. and App Permission Management, Working with Multimedia Content like Audio & Video in Android. Working with API Calls and Web Services,Packaging and Publishing Android Applications.

Unit V: Working with Databases in Android 5 hrs

Building database driven Apps in Android, Working with SQLite, Interacting with Remote Databases using JSON, Performing CRUD operations in both Local and Remote Databases. Understanding Realtime Databases and getting started with Firebase. Implementing Firebase backend in previously developed CRUD application.

Unit VI: Cross Platform Mobile Application Development using Flutter 10 hrs

Getting Started with Flutter and Dart, Understanding Flutter Architecture, Considering other alternatives, setting up the Development Environment, Material Design and System Services. Working with CRUD and HTTP Requests, Publishing and Packaging Apps for both Android and iOS and publishing at different platforms.

b) Practical Assignments

a) Build a Calculator App in Android.
b) Build a Tic-Tac-Toe Game. The game should keep records of Each Player and Game Time of each match.
c) Build an Android News Reader app which fetches news from an online API like Google News and shows the stories in a list. Whenever the user clicks on the heading of a particular story, the full story appears with the featured image.
d) Build a Simple Chat App in Flutter using Firebase. Export the app to both Android and iOS.

Particulars of course designer:

Name: Mr. Nibir Borpuzari
Contact No.: 8822306808
E-mail id: nibir@gauhati.ac.in

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

a) Theory: 45 hrs (45 classes)
b) Practical: 30 hrs (15 classes)
c) Non Contact: NIL

9. List of Reference 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

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.

B. 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.

Particulars of Course Designer:

Name: Dr Irani Hazarika
Contact No: 8486965773
Email: queensarathi@gmail.com

PROJECT

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

   - Develop practical software solutions using programming and database concepts.
   - Analyze real-world problems and design structured solutions.
   - Plan and manage projects using scheduling tools like Gantt charts.
   - Implement security, performance optimization, and system design techniques.
   - Improve technical writing, documentation, and presentation skills.

2. **Semester:** 6
3. **Course Type:** Compulsory
4. **Course Level:** 300-399
5. **Credits:** 4

A. Project Overview

The project should be undertaken in consultation with a guide and must involve **software development/Research work/Experimental work**. The proposal should clearly define the **objectives, scope, and technical environment** of the project.

B. Project Categories

Students can choose a project from the following domains:

- Relational Database Management Systems (RDBMS)
- Object-Oriented Programming Systems (OOPS)
- Networking
- Multimedia Applications
- Artificial Intelligence & Expert Systems
- Research-Based Projects
- Web Application/ Mobile Application

C. Project Synopsis & Report Guidelines

The project proposal and final report should include the following sections:

1. **Title of the Project**
2. **Introduction & Objectives** – Define the purpose and goals of the project.
3. **Project Category** – Specify the domain (RDBMS, AI, Networking, etc.).
4. **Technology Stack** – Specify the tools, platforms, hardware, and software requirements.
5. **Problem Definition & Specifications** – Define the problem, functional requirements, and technical specifications.
6. **Project Planning & Scheduling** – Provide a **Gantt chart** or **PERT chart** for project milestones.

7. **Scope of the Solution** – Describe the expected outcomes and limitations.
8. **System Analysis & Design**
    - **Diagrams:** DFDs, ER diagrams, or Class diagrams (as required).
    - **Module Breakdown:** Number of modules with descriptions.
    - **Data Structures:** Specify data structures used in each module.
    - **Process Logic:** Describe logic for each module.
    - **Implementation Methodology** – Describe the software development approach.
    - **Reports:** List of reports generated by the system.
9. **Network Architecture** (if applicable) – Provide a high-level system/network design.
10. **Security Implementation** – Describe security measures at various levels.
11. **Future Scope & Enhancements** – Suggest improvements for future development.
12. **Bibliography** – List references and sources used.

Students are encouraged to use Latex for documentation and project report preparation for professional formatting and quality.

Particulars of course designer:
Name: Dr. Deepjyoti Kalita
Contact No.: 9707464850
E-mail id : deepjyoti111@gmail.com

Graph Theory

1. Learning Outcome: After completing this course, students will have understanding of graph theoretic concepts, problems and associated algorithmic solutions.
2. Prerequisites: NIL
3. Semester: 6
4. Course Type: Compulsory
5. Course Level: 300-399
6. Theory Credit: 4
7. Practical Credit: 0
8. No of Hours:

a) Theory: 60 hrs (60 classes)
b) Practical: NIL
c) Non Contact: NIL

9. 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, PHI

10. Contents of 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

Particulars of course designer:

Name: Dr. Hasin Afzal Ahmed
Contact No.: 8011810533
E-mail id: hasin@gauhati.ac.in

SOFTWARE TESTING

1. LEARNING OUTCOMES:

By the end of this course, students will be able to:

- Understand the principles and importance of software testing.
- Design effective test cases and test plans.
- Identify and apply different testing techniques and strategies.
- Use automated testing tools for software verification.
- Analyze software quality and defect management processes.

2. COURSE OUTCOMES:

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

CO1: Demonstrate knowledge of software testing fundamentals and life cycle processes.

CO2: Apply appropriate testing techniques to design and execute test cases.

CO3: Utilize software testing metrics for defect management and quality assurance.

CO4: Implement and analyze both manual and automated testing methodologies.

CO5: Develop and evaluate test strategies for real-world software applications.

3. PREREQUISITE:

- Basic knowledge of software engineering

4. Semester: 6

5. Course Type: Elective

6. Course Level:300-399

7. Theory Credit: 4

8. Practical Credit: 0

9. No of Required Hours: 60

10. TEXTBOOKS/ RECOMMENDED READINGS:

a) Software Testing Fundamentals: Methods and Metrices: Marniw L. Huncheson:

Wiley Publishing

b) Software testing – Principles and Practices: Srinivasan Desikan , GopalaSwamy & Ramesh. (2008): Pearson Education.
c) Software Testing: Yogesh Singh, Cambridge University Press
d) Software Engineering: Rajib Mall: PHI Learning.
e) Software Testing: Louise Tamre, Pearson Education
f) Effective Software Testing: Elfriede Dustin: Pearson Education
g) Effective methods for Software Testing: William, E. Perry: John Wiley & Sons.

11. COURSE CONTENT:

Unit No & Name	**Components of the Unit**	**No of contact hours**
UNIT-I: Introduction	Software Testing Process, Concept of Testing and Objectives, Testing Techniques, Software Testing Life Cycle (STLC), Characteristics of STLC, Phases of STLC, Types of Error, Stubs and Drivers, Verification and Validation, Different Types of verification and validation methods, Concept of Code Review, Code Inspection, Code Walkthroughs	15
UNIT-II: Testing Methods	Software Testing Methods, Fundamentals of Testing, Test case Design, White Box Testing, White Box Testing techniques: *Statement Coverage, Branch Coverage, Condition Coverage, Multiple Condition Coverage, Path Testing,* Black Box Testing, Black Box Testing Techniques: *Equivalence Class Partitioning, Boundary Value Analysis,* Unit Testing, Integration Testing, System Testing, Validation Testing, Test Planning	15
UNIT-III: Software Metrics	Concept of Testing Metrics, Deloping Software Testing Metrics, Types of metrics, Software Testing Complexity Metrics, Project metrics-progress metrics-productivity metrics. What is win runner-Methods of testing in win runner, Definition of Defects, Defect management Process, Defect Repoting, Defect manegemtn metrics, Process Improvement using defects	15

UNIT-IV: Testing tools	Testing tools: static, dynamic tools. Characteristics of modern tools. Manual vs Automated Testing, Basics of Automated Testing, Advantages of Automated testing. Factors for Automated Testing, Drawback of manuel testing, Types of Automation testing tools. Case studies on Testing tools	15
	Total	60

Particulars of course designer:
Name : Dr. Deepjyoti Kalita
Contact No : +91-9707464850
Email-id : deepjyoti111@gmail.com

The END