Course Title: Theory of Computation	***Course Code: 20CS450***
Credits: 3	***Total Contact Hours: 39:0:0***
Type of Course: Theory	***Category: Professional Core Course***
CIE Marks: 50	***SEE Marks: 100***

Pre-requisite: Discrete Mathematical Structure.

Course Objectives: The course should enable the students to:

Sl. No.	Course Objectives
1	Introduce core concepts in Automata and Theory of Computation.
2	Identify different Formal Language Classes and their Relationships.
3	Design finite state machine, Grammars and Recognize for different formal Languages.
4	Prove or disprove theorems in automata theory using their properties.
5	Determine the decidability and intractability of Computational problems.

Unit No.	Course Content	No. of Hours
1	**Introduction to Finite Automata:** The central concepts of Automata theory; Deterministic finite automata; Nondeterministic finite automata; An application of finite automata; Finite automata with Epsilon-transitions; Equivalence and minimization of automata.	8
2	**Regular Expressions and Properties of Regular Languages:** Regular expressions; Converting DFA's to regular expression by eliminating states, Converting Regular Expressions to Finite Automata; Applications of Regular Expressions; Proving languages not to be regular languages; Closure properties of regular Languages.	8
3	**Context-Free Grammars and Languages:** Context-Free Grammars; Parse trees; Applications of Context-Free Grammars; Ambiguity in Grammars and Languages; Normal forms for CFGs.	8
4	**Pushdown Automata** Definition of the Pushdown Automata; The languages of a PDA; Equivalence of PDA's and CFG's; Deterministic Pushdown Automata.	7
5	**Introduction to Turing Machine:** The Turing machine; Programming techniques for Turing Machines, Extension to the basic Turing machine, restricted Turing machine. Undecidability: A language that is not recursively Enumerable, Undecidable problem that is RE, Undecidable Problems about turing Machines, post's correspondence problems, Other Undecidable Problems.	8

Text Books:

Sl. No.	Author/s	Title	Publisher Details
1	John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman	Introduction to Automata Theory, Languages and Computation	3rd Edition, Pearson education, 2007
2	Peter Linz	Finite Automata & Formal Languages	4th Edition, Narosa Publication, 2010.
3	John C Martin	Introduction to Languages and Automata Theory	3rd Edition, Tata McGraw- Hill, 2007.
4	Daniel I.A. Cohen	Introduction to Computer Theory	2nd Edition, John Wiley & Sons, 2004
5	Thomas A. Sudkamp	An Introduction to the Theory of Computer Science, Languages and Machines	3rd Edition, Pearson Education, 2006

Web Resources:

Sl. No.	Web Link
1	https://nptel.ac.in/courses/106104148/
2	https://nptel.ac.in/courses/106/104/106104028/

Course Outcomes: After completion of the course, students are able to:

CO1	Design automata for given regular languages.
CO2	Define Regular expression, and check its equivalence among automata.
CO3	Write grammars for context free languages and parse the given input.
CO4	Design pushdown Automata and show the equivalence of CFG and PDA
CO5	Apply the techniques of Turing machine, decidability and intractability of Computational problems.

Mapping Course Outcomes with Program outcomes & Program Specific outcomes:

Course Outcomes	Program Outcomes												PSO's			
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
CO-1	3	3	3	-	-	-	-	-	-	-	-	-	3	-	2	-
CO-2	3	3	3	-	-	-	-	-	-	-	-	-	3	-	2	-
CO-3	3	3	3	-	-	-	-	-	-	-	-	-	3	-	2	-
CO-4	3	3	3	-	-	-	-	-	-	-	-	-	3	-	2	-
CO-5	3	3	3	-	-	-	-	-	-	-	-	-	3	-	2	-

1-Low association, 2- Moderate association, 3-High association

Course Title: Operating Systems Lab	***Course Code: 20CS46L***
Credits: 1.5	***Contact Hours (L: T: P): 0: 0: 39***
Type of Course: Practical	***Category: Professional Core Course***
CIE Marks: 50	***SEE Marks: 50***

Pre-requisite: Computer Organization and Architecture, Data Structures, C Programming.

Course Objectives: The course should enable the students to:

Sl. No.	Course Objectives
1	Familiarize students with LINUX/UNIX OP and provide necessary skills for developing and debugging programs in these environments.
2	Learn shell script, creation and management of processes, IPC using shared memory and multithreads programing.
3	Analyze and develop process scheduling algorithms and process synchronization.

Weeks	List of Programs	No. of Hours
1	a. Exposure to Linux Operating System and Environment b. Write Shell Scripts for the following: i. Concatenation of two strings ii. Comparison of two strings iii. Maximum of three numbers	3
2	Write Shell Scripts for the following: a. Fibonacci series b. Arithmetic operation using case c. Check whether a given number is palindrome or not	3
3	Write Shell Scripts for the following: a. Finding largest of N numbers (storing numbers in an array) b. Generating prime numbers. c. Reading two matrices and finding sum	3
4	Simulation of ls, rm, cat and grep commands using system calls.	3
5	a. Write a program to perform the following tasks using system calls: i. Parent process should create a child process ii. Both parent child processes should display their pid and parent's pid; parent process should also display its child's pid iii. Load a new program into child process iv. The parent process should terminate after the child process terminates b. Program to demonstrate the creation of Zombie and Orphan processes. c. Program to perform the following task using I/O system calls for file I/O i. Reading first 10 characters from file ii. Skipping 5 characters from current position in the file iii. Going to 5th last character in the file iv. Going to the 3rd character in the file d. Program to read from file and write into a new file using I/O system calls for file I/O	3

6	a. Program to implement Producer-consumer problem using the following shared memory methods: i. using shm_open and mmap ii. shmget and shmat b. Write a program to generate first N ODD numbers with the following requirements: - Parent program should create a child and distribute the task of generating odd numbers to its child. - The code for generating odd numbers should reside in different program. - Child should write the odd numbers to a shared memory. - Parent process has to print the odd numbers by retrieving from the shared memory. i. Implement the above using shmget and shmat Note: Shared object should be removed at the end in the program	3
7	a. Write a program to generate and print Fibonacci series with the following requirements: - Parent program should create a child and distribute the task of generating Fibonacci series to its child. - The code for generating Fibonacci series should reside in different program. - Child should write the generated Fibonacci series to a shared memory. - Parent process has to print by retrieving the Fibonacci series from the shared memory. i. Implement the above using shmget and shmat ii. Implement the above using shm_open and mmap Note: Shared object should be removed at the end in the program. b. Write a program to generate and print Prime numbers between a given range (between M & N) with the following requirements - M & N should be passed as command line arguments - Error checking should be done to verify the required number of arguments at the command line - Parent program should create a child and distribute the task of generating Prime numbers to its child. - The code for generating Prime numbers should reside in different program. - Child should write the generated Prime numbers to a shared memory. - Parent process has to print by retrieving the Prime numbers from the shared memory. i. Implement the above using shm_open and mmap Note: Shared object should be removed at the end in the program.	3
8	a. Write a program with two threads and a main thread. Schedule the task of calculating the natural sum upto ‘n’ terms and factorial of ‘n’ on these threads. Note: The main thread should read ‘n’ from command line and pass it as parameter to remaining threads. Terminate the threads using system calls. b. Implement FCFS scheduling algorithm by defining the process structure.	3
9	Implement the following CPU scheduling algorithms by defining the process structure:	3

	a. SJF b. SJF (Preemptive)	
10	Implement the following CPU scheduling algorithms by defining the process structure: a. Priority b. Priority (Preemptive)	3
11	Implement Round Robin CPU scheduling algorithm with arrival time by defining the process structure.	3
12	Implement the following using mutex and semaphores: a. Producer – Consumer problem b. Reader's writers' problem	3
13	**Lab Test/Event**	3

Reference Books:

Sl. No.	Author/s	Title	Publisher Details
1	Abraham Silberschatz, Peter Baer Galvin, Greg Gagne	Operating System Concepts	9th Edition, Wiley India, 2013
2	William Stallings	Operating Systems: Internals and Design Principles	7th Edition, Prentice Hall of India,2017
3	D.M Dhamdhere	Operating systems - A concept-based Approach	4th Edition, Tata McGraw- Hill, 2013
4	P.C.P. Bhatt	Introduction to Operating Systems	Concepts and Practice, 4th Edition, PHI, 2014

Web Resources:

Sl. No.	Web Link
1	https://youtu.be/783KAB-tuE4 - NPTEL IIT, Madras
2	https://nptel.ac.in/courses/106108101/

Course Outcomes:

CO1	Implement shell programs and design process management and file system management with system calls.
CO2	Design and implement Inter Process Communication and multiple threads application.
CO3	Analyze and implement CPU scheduling algorithms and process synchronization.

Mapping Course Outcomes with Program outcomes & Program Specific outcomes:

Course Outcomes	Program Outcomes												PSO's			
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
CO1	3	3	3	-	-	-	-	-	3	3	-	-	3	-	3	-
CO2	3	3	3	3	-	-	-	-	3	3	-	-	3	-	3	-
CO3	3	3	3	3	-	-	-	-	3	3	-	-	3	-	3	-

1-Low association, 2- Moderate association, 3-High association

Course Title: Design and Analysis of Algorithms Lab	***Course Code: 20CS47L***
Credits: 1.5	***Contact Hours (L: T: P): 0: 0: 39***
Type of Course: Practical	***Category: Professional Core Course***
CIE Marks: 50	***SEE Marks: 50***

Pre-requisite: Data Structures.

Course Objectives: The course should enable the students to:

Sl. No.	Course Objectives
1	Analyze the problem domain; Choose the appropriate data structures and design technique based on the problem domain.
2	Implement algorithms and perform analysis with empirical method.
3	Evaluate the performance of different algorithms using different design techniques for solving the same problem.

Weeks	List of Programs	No. of Hours
1	Implement Euclid's, consecutive integer checking and modified Euclid's algorithms to find GCD of two nonnegative integers and perform comparative analysis.	3
2	Implement the following searching algorithms and perform their analysis for worst case, best-case and average inputs. a) Sequential Search b) Binary Search (Recursive)	3
3	Implement the following elementary sorting algorithms and perform their analysis for worst case, best-case and average inputs a) Selection Sort b) Bubble Sort c) Insertion Sort	3
4	Implement Brute force string matching algorithm to search for a pattern of length 'M' in a text of length 'N' (M<=N) and perform its analysis for worst case, best-case and average inputs.	3
5	Implement Merge Sort algorithm and perform its analysis for worst case, best-case and average inputs.	3
6	Implement Quick Sort algorithm and perform its analysis for worst case, best-case and average inputs.	3
7	a) Implement DFS algorithm to check for connectivity and acyclicity of a graph. If not connected, display the connected components. Perform its analysis for different inputs b) Implement BFS algorithm to check for connectivity and acyclicity of a graph. If not connected, display the connected components. Perform its analysis for different inputs	3

8	a) Implement DFS based algorithm to list the vertices of a directed graph in Topological ordering. Perform its analysis for different inputs b) Implement source removal algorithm to list the vertices of a directed graph in Topological ordering. Perform its analysis for different inputs (Note: use efficient method to identify the source vertex).	3
9	Implement heap sort algorithm with bottom-up heap construction. Analyze its efficiency for worst case, best-case and average case inputs.	3
10	a) Implement Warshall's Algorithm to find the transitive closure of a directed graph and perform its analysis for different inputs. b) Implement Floyd's Algorithm to find All-pair shortest paths for a graph and perform its analysis for different inputs	3
11	a) Implement an algorithm to solve Knapsack problem with Dynamic Programming approach and perform its analysis for different inputs. b) Implement Prim's algorithm to find Minimum Spanning Tree of a graph and perform its analysis for different inputs	3
12	Implement Dijkstra's algorithm to find shortest paths to other vertices in a graph and perform its analysis for different inputs.	3
13	**Lab Test/Event**	3

Reference Books:

Sl. No.	Author/s	Title	Publisher Details
1	Anany Levitin	Introduction to The Design & Analysis of Algorithms	3rd Edition, Pearson Education, 2012.
2	Ellis Horowitz, SartajSahni, Sanguthevar Rajasekaran	Fundamentals of Computer Algorithms	2nd Edition, Universities Press, 2013.
3	Thomas H. Cormen, Charles E. Leiserson, Ronal L. Rivest, Clifford Stein	Introduction to Algorithms	3rd Edition, PHI ,2010
4	R.C.T. Lee, S.S. Tseng, R.C. Chang & Y.T. Tsai	Introduction to the Design and Analysis of Algorithms A Strategic Approach	Tata McGraw Hill, 2012.

Web Resources:

Sl. No.	Web Link
1	https://nptel.ac.in/courses/106101060/
2	https://nptel.ac.in/courses/106106131/

Course Outcomes:

CO1	Implement searching and sorting algorithms.
CO2	Implement graph based searching and sorting algorithms using Divide and Conquer and Decrease and Conquer techniques.
CO3	Implement spanning tree and shortest path algorithms using Greedy and Dynamic techniques.

Mapping Course Outcomes with Program outcomes & Program Specific outcomes:

Course Outcomes	Program Outcomes												PSO's			
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
CO1	3	3	3	-	-	-	-	-	3	3	-	-	3	-	3	-
CO2	3	3	3	3	-	-	-	-	3	3	-	-	3	-	3	-
CO3	3	3	3	3	-	-	-	-	3	3	-	-	3	-	3	-

1-Low association, 2- Moderate association, 3-High association