(viii) Concatenate two circularly linked lists

4. Implement a stack as an ADT using Arrays.

5. Implement a stack as an ADT using the Linked List ADT.

6. Write a program to evaluate a prefix/postfix expression using stacks.

7. Implement Queue as an ADT using the circular Arrays.

8. Implement Queue as an ADT using the Circular Linked List ADT.

9. Write a program to implement Binary Search Tree as an ADT which supports the following operations:
   (i) Insert an element x
   (ii) Delete an element x
   (iii) Search for an element x in the BST and change its value to y and then place the node with value y at its appropriate position in the BST
   (iv) Display the elements of the BST in preorder, inorder, and postorder traversal
   (v) Display the elements of the BST in level-by-level traversal
   (vi) Display the height of the BST

10. Write a program to implement a balanced search tree as an ADT.

**Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.**

## DISCIPLINE SPECIFIC CORE COURSE – 8 (DSC-8): Operating Systems

**Credit distribution, Eligibility and Prerequisites of the Course**

| **Course title & Code** | **Credits** | **Credit distribution of the course** | | | **Eligibility criteria** | **Pre-requisite of the course (if any)** |
|---|---|---|---|---|---|---|
| | | **Lecture** | **Tutorial** | **Practical/ Practice** | | |
| **DSC 08 Operating Systems** | **4** | **3** | **0** | **1** | **Passed 12th class with Mathematics** | Programming using Python/Object Oriented Programming with C++, Computer System Architecture |

## Learning Objectives

The course provides concepts that underlie all operating systems and are not tied to any particular operating system. The emphasis is on explaining the need and structure of an operating system using its common services such as process management (creation, termination etc.), CPU Scheduling, Process Synchronization, Handling Deadlocks, main memory management, virtual memory, secondary memory management. The course also introduces various scheduling algorithms, structures, and techniques used by operating systems to provide these services.

## Learning outcomes

On successful completion of the course, students will be able to:

- Describe the need of an operating system and define multiprogramming and Multithreading concepts.
- Implement the process synchronization service (Critical Section, Semaphores), CPU scheduling service with various algorithms.
- Implement Main memory Management (Paging, Segmentation) algorithms, Handling of Deadlocks
- Identify and appreciate the File systems Services, Disk Scheduling service

## SYLLABUS OF DSC-8

### Unit 1 (6 hours)

**Introduction:** Operating Systems (OS) definition and its purpose, Multiprogrammed and Time Sharing Systems, OS Structure, OS Operations: Dual and Multi-mode, OS as resource manager.

### Unit 2 (9 hours)

**Operating System Structures:** OS Services, System Calls: Process Control, File Management, Device Management, and Information Maintenance, Inter-process Communication, and Protection, System programs, OS structure- Simple, Layered, Microkernel, and Modular.

### Unit 3 (10 hours)

**Process Management:** Process Concept, States, Process Control Block, Process Scheduling, Schedulers, Context Switch, Operation on processes, Threads, Multicore Programming, Multithreading Models, PThreads, Process Scheduling Algorithms: First Come First Served, Shortest-Job-First, Priority & Round-Robin, Process Synchronization: The critical-section problem and Peterson’s Solution, Deadlock characterization, Deadlock handling.

### Unit 4 (11 hours)

**Memory Management:** Physical and Logical address space, Swapping, Contiguous memory allocation strategies - fixed and variable partitions, Segmentation, Paging.
Virtual Memory Management: Demand Paging and Page Replacement algorithms: FIFO Page Replacement, Optimal Page replacement, LRU page replacement.

**Unit 5 (9 hours)**

**File System:** File Concepts, File Attributes, File Access Methods, Directory Structure: Single-Level, Two-Level, Tree-Structured, and Acyclic-Graph Directories.
Mass Storage Structure: Magnetic Disks, Solid-State Disks, Magnetic Tapes, Disk Scheduling algorithms: FCFS, SSTF, SCAN, C-SCAN, LOOK, and C-LOOk Scheduling.

**Essential/recommended readings**

1. Silberschatz, A., Galvin, P. B., Gagne G. *Operating System Concepts,* 9th edition, John Wiley Publications, 2016.
2. Tanenbaum, A. S. *Modern Operating Systems*, 3rd edition, Pearson Education, 2007.
3. Stallings, W. *Operating Systems: Internals and Design Principles,* 9th edition, Pearson Education, 2018.

**Additional References**

1. Dhamdhere, D. M., *Operating Systems: A Concept-based Approach,* 2nd edition, Tata McGraw-Hill Education, 2017.
2. Kernighan, B. W., Rob Pike, R. *The Unix Programming Environment*, Englewood Cliffs, NJ: Prentice-Hall, 1984.

**Suggested Practical List (If any): (30 Hours)**

**Practical exercises such as**

1. Execute various Linux commands for:
   i. Information Maintenance: wc, clear, cal, who, date, pwd
   ii. File Management: cat, cp, rm, mv, cmp, comm, diff, find, grep, awk
   iii. Directory Management : cd, mkdir, rmdir, ls
2. Execute various Linux commands for:
   i. Process Control: fork, getpid, ps, kill, sleep
   ii. Communication: Input-output redirection, Pipe
   iii. Protection Management: chmod, chown, chgrp
3. Write a programme (using fork() and/or exec() commands) where parent and child execute:
   i. same program, same code.
   ii. same program, different code.
   iii. Before terminating, the parent waits for the child to finish its task.
4. Write a program to to report behaviour of Linux kernel including kernel version, CPU type and model. (CPU information)

5. Write a program to report behaviour of Linux kernel including information on 19 configured memory, amount of free and used memory. (Memory information)
6. Write a program to copy files using system calls.
7. Use an operating system simulator to simulate operating system tasks.
8. Write a program to implement scheduling algorithms FCFS/ SJF/ SRTF/ non-preemptive scheduling algorithms.
9. Write a program to calculate the sum of n numbers using Pthreads. A list of n numbers is divided into two smaller lists of equal size, and two separate threads are used to sum the sublists.
10. Write a program to implement first-fit, best-fit and worst-fit allocation strategies.

**Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.**

## DISCIPLINE SPECIFIC CORE COURSE– 9 (DSC-9): Numerical Optimization

**Credit distribution, Eligibility and Pre-requisites of the Course**

| **Course title & Code** | **Credits** | **Credit distribution of the course** | | | **Eligibility criteria** | **Pre-requisite of the course (if any)** |
|---|---|---|---|---|---|---|
| | | **Lecture** | **Tutorial** | **Practical/ Practice** | | |
| **DSC09 Numerical Optimization** | **4** | **3** | **0** | **1** | **Passed 12th class with Mathematics** | Programming using Python/Object Oriented Programming with C++ |

### Learning Objectives
The course aims to provide students with the experience of mathematically formulating a large variety of optimization/decision problems emerging out of various fields like data science, machine learning, business, and finance. The course focuses on learning techniques to optimize problems in order to obtain the best possible solution.

### Learning outcomes
At the end of the course, students will be able to:
- Mathematically formulate the optimization problems using the required number of independent variables.
- Define constraint functions on a problem.
- Check the feasibility and optimality of a solution.
- Apply conjugate gradient method to solve the problem.