University of Engineering & Management, Jaipur

Syllabus for BSc Computer Science Admission Batch 2026

SEMESTER 3

S. No.	Subject Code	Subject Name	L	T	P	Total Contact Hrs	Credit Points
Theory							
1	BSCCS301	Data Structures	3	1	0	4	4
2	BSCCS302	Operating System	3	1	0	4	4
3	BSCCS303	Computer Networks	3	1	0	4	4
4	BSCCS304	General Studies & Current Affairs -III	3	1	0	4	4
Practical							
5	BSCCS391	Data Structures Lab	0	0	2	2	2
6	BSCCS392	Operating System Lab	0	0	2	2	2
7	BSCCS393	Computer Networks Lab	0	0	2	2	2
Sessional							
8	BSCCS394	Principles of Management	0	2	0	2	2
		Total	12	6	6	24	24

Subject Code: BSCCS301

Subject Name: Data Structures

Credit Points: 04

Course Objective

This course aims at developing the ability to use basic data structures like array, stacks, queues, lists, trees and hash tables to solve problems. C++ is chosen as the language to understand implementation of these data structures.

Course Learning Outcomes

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

1. Implement and empirically analyse linear and non-linear data structures like Arrays, Stacks, Queues, Lists, Trees, Heaps and Hash tables as abstract data structures. (RBT L2/3)
2. Write a program, choosing a data structure, best suited for the application at hand. (RBT L3/4)
3. Re-write a given program that uses one data structure, using a more appropriate/efficient data structure (RBT L4)
4. Write programs using recursion for simple problems. Explain the advantages and disadvantages of recursion.(RBT L2/L3)
5. Identify Ethical Dilemmas.

Unit 1

Arrays: single and multi-dimensional arrays, analysis of insert, delete and search operations in arrays (both linear search and binary search), implementing sparse matrices, applications of arrays to sorting: selection sort, insertion sort, bubble sort, comparison of sorting techniques via

empirical studies. Introduction to Vectors.

Unit 2

Linked Lists: Singly- linked, doubly-linked and circular lists, analysis of insert, delete and search operations in all the three types, implementing sparse matrices. Introduction to Sequences.

Unit 3

Queues: Array and linked representation of queue, de-queue, comparison of the operations on queues in the two representations. Applications of queues.

Unit 4

Stacks: Array and linked representation of stacks, comparison of the operations on stacks in the two representations, implementing multiple stacks in an array; applications of stacks: prefix, infix and postfix expressions, utility and conversion of these expressions from one to another; applications of stacks to recursion: developing recursive solutions to simple problems, advantages and limitations of recursion

Unit 5

Trees and Heaps: Introduction to tree as a data structure; binary trees, binary search trees, analysis of insert, delete, search operations, recursive and iterative traversals on binary search trees. Height-balanced trees (AVL), B trees, analysis of insert, delete, search operations on AVL and B trees. Introduction to heap as a data structure. analysis of insert, extract-min/max and delete-min/max operations, applications to priority queues.

Unit 6

Hash Tables: Introduction to hashing, hash tables and hashing functions -insertion, resolving collision by open addressing, deletion, searching and their analysis, properties of a good hash function

References

1. Drozdek, A., (2012), Data Structures and algorithm in C++. 3rd edition. Cengage Learning.
2. Goodrich, M., Tamassia, R., & Mount, D., (2011). Data Structures and Algorithms Analysis in C++. 2nd edition. Wiley.

Additional Resources:

1. Foruzan, B.A. (2012) Computer Science: A Structured Approach Using C++, Cengage Learning

2. Lafore, R. (2008). Object Oriented Programming in C++. 4th edition. SAMS Publishing.

3. Sahni, S. (2011). Data Structures, Algorithms and applications in C++. 2ndEdition, Universities Press

4. Tenenbaum, A. M., Augenstein, M. J., & Langsam Y., (2009), Data Structures Using C and C++. 2nd edition. PHI.

Subject Code: BSCCS391

Subject Name: Data Structure Lab

Credit Points: 02

Practical

1. Write a program to search an element from a list. Give user the option to perform Linear or Binary search. Use Template functions.
2. WAP using templates to sort a list of elements. Give user the option to perform sorting using Insertion sort, Bubble sort or Selection sort.
3. Implement Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list and concatenate two linked lists (include a function and also overload operator +).
4. Implement Doubly Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.
5. Implement Circular Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.
6. Perform Stack operations using Linked List implementation.
7. Perform Stack operations using Array implementation. Use Templates.
8. Perform Queues operations using Circular Array implementation. Use Templates.
9. Create and perform different operations on Double-ended Queues using Linked List implementation.
10. WAP to scan a polynomial using linked list and add two polynomial.
11. WAP to calculate factorial and to compute the factors of a given no. (i)using recursion, (ii) using iteration
12. (ii) WAP to display fibonacci series (i)using recursion, (ii) using iteration
13. WAP to calculate GCD of 2 number (i) with recursion (ii) without recursion

14. WAP to create a Binary Search Tree and include following operations in tree:

(a) Insertion (Recursive and Iterative Implementation)

(b) Deletion by copying

(c) Deletion by Merging

(d) Search a no. in BST

(e) Display its preorder, postorder and inorder traversals Recursively

(f) Display its preorder, postorder and inorder traversals Iteratively

(g) Display its level-by-level traversals

(h) Count the non-leaf nodes and leaf nodes

(i) Display height of tree

(j) Create a mirror image of tree

(k) Check whether two BSTs are equal or not

15. WAP to convert the Sparse Matrix into non-zero form and vice-versa.

16. WAP to reverse the order of the elements in the stack using additional stack.

17. WAP to reverse the order of the elements in the stack using additional Queue.

18. WAP to implement Diagonal Matrix using one-dimensional array.

19. WAP to implement Lower Triangular Matrix using one-dimensional array.

20. WAP to implement Upper Triangular Matrix using one-dimensional array.

21. WAP to implement Symmetric Matrix using one-dimensional array.

22. WAP to create a Threaded Binary Tree as per inorder traversal, and implement operations like finding the successor / predecessor of an element, insert an element, inorder traversal.

23. WAP to implement various operations on AVL Tree.

24. WAP to implement heap operations.

References

1. Drozdek, A., (2012), Data Structures and algorithm in C++. 3rd edition. Cengage Learning.
2. Goodrich, M., Tamassia, R., & Mount, D., (2011). Data Structures and Algorithms Analysis in C++. 2nd edition. Wiley.

Additional Resources:

1. Foruzan, B.A. (2012) Computer Science: A Structured Approach Using C++, Cengage Learning

2. Lafore, R. (2008). Object Oriented Programming in C++. 4th edition. SAMS Publishing.

3. Sahni, S. (2011). Data Structures, Algorithms and applications in C++. 2ndEdition, Universities Press

4. Tenenbaum, A. M., Augenstein, M. J., & Langsam Y., (2009), Data Structures Using C and C++. 2nd edition. PHI.

Subject Code: BSCCS302

Subject Name: Operating System

Credit Points: 04

Course Objective

The course introduces the students to different types of operating systems. Operating system modules such as memory management, process management and file management are covered in detail.

Course Learning Outcomes

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

1. Implement multiprogramming, multithreading concepts for a small operating system.
2. Create, delete, and synchronize processes for a small operating system.
3. Implement simple memory management techniques.
4. Implement CPU and disk scheduling algorithms.
5. Use services of modern operating system efficiently
6. Implement a basic file system.

Detailed Syllabus

Unit 1

Introduction: Operating systems (OS) definition, Multiprogramming and Time Sharing operating systems, real time OS, Multiprocessor operating systems, Multicore operating systems, Various computing environments

Unit 2

Operating System Structures: Operating Systems services, System calls and System programs, operating system architecture (Micro Kernel, client server) operating

Unit 3

Process Management: Process concept, Operation on processes, Multi-threaded processes and models, Multicore systems, Process scheduling algorithms, Process synchronization. The Critical-section problem and deadlock characterization, deadlock handling.

Unit 4

Memory Management: Physical and Logical address space; Memory allocation strategies - Fixed and Variable Partitions, Paging, Segmentation, Demand Paging and virtual memory, Page Replacement algorithm.

Unit 5

File and I/O Management: Directory structure, File access methods, Disk scheduling algorithms.

References

1. Silberschatz, A., Galvin, P. B., & Gagne, G. (2008). Operating Systems Concepts. 8th edition.. John Wiley Publications.

Additional Resources

1. Dhamdhere, D. M. (2006). Operating Systems: A Concept-based Approach. 2nd edition. Tata McGraw-Hill Education.

2. Kernighan, B. W., & Rob Pike, R. (1984). The Unix programming environment (Vol. 270). Englewood Cliffs, NJ: Prentice-Hall

3. Stallings, W. (2018). Operating Systems: Internals and Design Principles. 9th edition. Pearson Education.

4. Tanenbaum, A. S. (2007). Modern Operating Systems. 3rd edition. Pearson Education.

Subject Code: BSCCC392

Subject Name: Operating System Lab

Credit Points: 02

Practical

1. Write a program (using fork() and/or exec() commands) where parent and child execute: a) same program, same code. b) same program, different code. - c) before terminating, the parent waits for the child to finish its task.

2. Write a program to report behaviour of Linux kernel including kernel version, CPU type and model. (CPU information)

3. Write a program to report behaviour of Linux kernel including information on 19 configured memory, amount of free and used memory. (memory information) 4. Write a program to print file details including owner access permissions, file access time, where file name is given as argument.

5. Write a program to copy files using system calls. 6. Write a program to implement FCFS scheduling algorithm.

7. Write a program to implement Round Robin scheduling algorithm. 8. Write a program to implement SJF scheduling algorithm.

9. Write a program to implement non-preemptive priority based scheduling algorithm.

10. Write a program to implement preemptive priority based scheduling algorithm.

11. Write a program to implement SRJF scheduling algorithm.

12. Write a program to calculate sum of n numbers using thread library.

13. Write a program to implement first-fit, best-fit and worst-fit allocation strategies.

Subject Code: BSCCS303

Subject Name: Computer Network

Credit Points: 02

Course Objective

This course covers the concepts of data communication and computer networks. It comprises of the study of the standard models for the layered protocol architecture to communicate between autonomous computers in a network and also the main features and issues of communication protocols for different layers. Topics covered comprise of introduction to OSI and TCP/IP models also.

Course Learning Outcomes

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

1. Describe the hardware, software components of a network and their interrelations.
2. Compare OSI and TCP/IP network models.
3. Describe, analyze and compare different data link, network, and transport layer protocols.
4. Design/implement data link and network layer protocols in a simulated networking environment.

Detailed Syllabus

Unit 1

Introduction: Types of computer networks, Internet, Intranet, Network topologies, Network classifications.

Unit 2

Network Architecture Models: Layered architecture approach, OSI Reference Model, TCP/IP Reference Model.

Unit 3

Physical Layer: Analog signal, digital signal, digital modulation techniques (ASK, PSK, QAM), encoding techniques, maximum data rate of a channel, transmission media (guided transmission media, wireless transmission, satellite communication), multiplexing (frequency division multiplexing, time division multiplexing, wavelength division multiplexing).

Unit 4

Data Link MAC Layer: Data link layer services, error-detection and correction techniques, error recovery protocols (stop and wait, go back n, selective repeat), multiple access protocols, (TDMA/FDP, CDMA/FDD/CSMA/CD, CSMA/CA), Datalink and MAC addressing, Ethernet, data link layer switching, point-to-point protocol

Unit 5

Network layer: Networks and Inter networks, virtual circuits and datagrams, addressing, sub netting, Routing- (Distance vector and link state routing), Network Layer Protocols- (ARP, IPV4, ICMP, IPV6).

Unit 6

Transport and Application Layer: Process to process Delivery- (client server paradigm, connectionless versus connection oriented service, reliable versus unreliable); User Datagram Protocols, TCP/IP protocol, Flow Control.

Unit 7

Protocols: FTP (File Transfer protocol), SMTP (Simple, Mail Transfer Protocol), Telnet and remote login protocol, WWW (World Wide Web), HTTP (Hyper Text Transfer protocol), Uniform Resource Locator, HTML and forms.

References

1. Forouzan, B. A. (2017). Data Communication and Networking. McGraw-Hill Education

2. Tanenbaum, A.S. & Wethrall,D.J. (2012). Computer Networks. Pearson Education

Additional Resources

1. Kozierok, C.M. The TCP/IP Guide, free online resource. (2005.). Retrieved from http://www.tcpipguide.com/free/index.htm
2. Kurose, J. F., & Ross, K. W. (2017). Computer Networking: A Top-Down Approach. Pearson Education India
3. Stallings, W. (2017). Data and Computer Communications. 10th edition. Pearson Education India.

Subject Code: BSCCS393

Subject Name: Computer Network Lab

Credit Points: 02

Practical

1. Simulate Cyclic Redundancy Check (CRC) error detection algorithm for noisy channel.

2. Simulate and implement stop and wait protocol for noisy channel.

3. Simulate and implement go back n sliding window protocol.

4. Simulate and implement selective repeat sliding window protocol.

5. Simulate and implement distance vector routing algorithm

6. Simulate and implement Dijkstra algorithm for shortest path routing.