UNIVERSITY OF CALCUTTA

Notification No. CSR/ 09 /19

It is notified for information of all concerned that the Syndicate in its meeting held on 27.02.019 (vide Item No.14) subsequently confirmed by the Syndicate dated 28.03.2019 (vide Item No.1) has approved some corrections in the existing syllabus of paper CMS-A-CC-2-3-P for Semester - II, B.Sc. Computer Science (Honours) under CBCS (as notified vide Notification No.CSR/12/18 dated 04.06.2018) under this University and the new amended syllabus for the said paper (CMS-A-CC-2-3-P) is laid down in the accompanying pamphlet.

The above shall be effective from the academic session 2018-19.

SENATE HOUSE
KOLKATA-700073
The 10th April, 2019

10/04/19
(Debabrata Manna)
Deputy Registrar (Acting)

Semester - II

CMS-A-CC-2-3-P: Data Structure Lab. using C.
Core Course- 3: Practical, Credits - 02, Contact hours - 40 hours.

1. Write a program to search an element from a list. Give user the option to perform Linear or Binary search.
2. WAP 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. Include functions for insertion, deletion and search of a number, reverse the list and concatenate two linked lists.
4. Implement Doubly Linked List. Include functions for insertion, deletion and search of a number, reverse the list.
5. Implement Circular Linked List. 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.
8. Perform Queue operations using Array and linked list implementation.
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 polynomials.
11. WAP to create a Binary Search Tree and include following operations in tree:
    - **(a)** Insertion (Recursive and Iterative Implementation).
    - **(b)** Deletion.
    - **(c)** Search a node in BST.
    - **(d)** Display its preorder, postorder and inorder traversals recursively.
    - **(e)** Display its preorder, postorder and inorder traversals iteratively.
    - **(f)** Display its level-by-level traversals.
    - **(g)** Count the non-leaf nodes and leaf nodes.
    - **(h)** Display height of tree.
    - **(i)** Create a mirror image of tree.
12. WAP to reverse the order of the elements in the stack using additional stack.
13. WAP to reverse the order of the elements in the stack using additional Queue.

Note: These are only sample programs, more can be included related to the theory.

Text/ Reference Books

1. Fundamentals of Data Structures in C, Ellis Horowitz, Sartaj Sahni, Susan Anderson-Freed, University Press.
2. Data Structures: A Pseudocode Approach with C, Richard F. Gilberg and Behrouz A. Forouzan, Cengage Learning.
3. Data Structure using C, E Balagurusamy, McGraw Hill.
4. Data Structures Using C and C++, Aaron M. Tanenbaum, Moshe J. Augenstein, Yedidyah Langsam, , 2nd edition., PHI.
5. Classic Data Structures, Debasis Samanta, Second Edition, EEE, PHI.
6. Data Structures, Seymour Lipschutz, Schaum's Outlines, Tata McGraw Hill.
7. Data Structures Through C (A practical approach), G.S Baluja, Dhanpat Rai & Co.
8. The C Programming Language, Brian W. Kernighan and Dennis Ritchie, Prentice Hall.