University of Engineering & Management, Kolkata

Department of Computer Science & Engineering

DATA STRUCTURE LAB

PROJECT LIST

Subject Code - CS392

SESSION- 2018 ODD SEM

Submission Date-29.10.2018

RULES & REGULATIONS

- **Lab Project Carries 20 marks of the final semester lab examination.**
- **Each group consists of 4 members.**
- **Each group having a Project number which is same with the group number.**
- **The project is needed to be submitted on or before 29.10.2018.There will be no extension of the date of submission.**
- **The project is needed to be run on the day of submission and a project presentation will be there group wise.**
- **To complete the project it is advised to use "language C" only.**

1.Decision Tree:

Explore Decision tree classifier and implement it. Use a suitable tree data structure to implement the classifier. Use the following data to train and test the program;

RID	*age*	*Income*	*student*	*Credit rating*	*Class: buys Computer*
1	youth	High	no	Fair	No
2	youth	High	no	excellent	No
3	middle	High	no	fair	Yes
4	senior	Medium	no	fair	Yes
5	senior	Low	yes	fair	Yes
6	senior	Low	yes	excellent	No
7	middle	Low	yes	excellent	Yes
8	youth	Medium	no	fair	No
9	youth	Low	yes	fair	Yes
10	senior	Medium	yes	fair	Yes
11	youth	Medium	yes	excellent	Yes
12	middle	Medium	no	excellent	Yes
13	middle	High	yes	fair	Yes
14	senior	Medium	no	excellent	No

2.Trie:

Trie is a kind of search tree used to store dynamic sets and the keys in each node are usually strings. One of the applications of trie is predictive text or autocomplete feature in word processors, web browsers etc. Implement a trie and use it to produce a probable list of words that a user can write using first 3 to 5 letters typed by the user. Use a dictionary having reasonable number of words to test the application.

3.File management system:

The system should follow the working principle of ‘ls’ command in Linux Operating system. It should be able to show the files, folders under the current directory. The user should be able to add file, folders under specific folder. If the folder under which the user want to add the files do not exist, it should be created first and then the files/folders should be added. Use tree data structure to implement the system. Please note that this files or folders mentioned above need not to be created physically. Storing the names would be sufficient.

4.Routing:

A network consists of several computing devices. There could be many different ways through which a message from one computer can reach another computer. The network can be thought of as a graph. The weights of the graph indicate the cost associated with different links (connection between nodes). Given two different nodes, the system should first check if a path is available to reach the destination node or not. If the path is available then find the path having least cost. Also implement a buffer to store last 10 request results, if in future the same route is searched, it can be served faster.

5.Tic-Tac-Toe:

Implement an application to play tic-tac-toe game against computer. Use tree data structure to store the game states.

6.Spell Checker:

Spell checker is used to check if a user typed word is correctly spelled using a dictionary of correct words. Word processors (e.g. MS-Word), electronic mail, web search engines and many other applications use spell checkers. The application should be able to check if a user typed word is correct or not based on dictionary look up. Also the application should suggest the nearest word from the dictionary. To test the application, use a dictionary having reasonable number of words.

7. Dictionary

Create a project to search the meaning of a word using linked list. Application will behave as a dictionary which will provide synonyms, antonyms as well as meaning.

8. Graph Traversal Algorithm

Implement the concept of Breadth First Search & Depth First Search. Check that no cycle should be there.

9. Expression Tree

Create expression tree with respect to given expression. Try to handle all the different types of expression.

10. Spanning Tree

Create a project which will identify spanning tree from a graph.

11. Flattening a Linked List

Given a **L**inked **L**ist where every node represents a linked list and contains two pointers of its type: (i) a **next** pointer to the next node

(ii) a **bottom** pointer to a linked list where this node is head.

You have to **flatten** the linked list to a **single linked list** which is **sorted.**

The node structure has **3** fields as mentioned. A **next pointer**, a **bottom** pointer and a **data** part.

```
5 -> 10 -> 19 -> 28
|    |     |     |
V    V     V     V
7    20    22    35
|          |     |
V          V     V
8          50    40
|                |
V                V
30               45
```

and after flattening it, the sorted linked list looks like:

5->7->8->10->19->20->22->28->30->35->40->45->50

12. Intersection Point in Y Shapped Linked Lists

There are two singly linked lists in a system. By some programming error the end node of one of the linked list got linked into the second list, forming a inverted Y shaped list. Write a program to get the point where two linked lists merge.

The function should return data value of a node where two linked lists merge. If linked list do not merge at any point, then it should return -1.

13. Producer-Consumer Problem

The problem describes two processes, the producer and the consumer, which share a common, fixed-size buffer used as a queue.

The producer's job is to generate data, put it into the buffer, and start again.
At the same time, the consumer is consuming the data (i.e. removing it from the buffer), one piece at a time.

Make sure that the producer won't try to add data into the buffer if it's full and that the consumer won't try to remove data from an empty buffer.

14. Merge K sorted linked lists

Given K sorted linked list your task is to merge them.

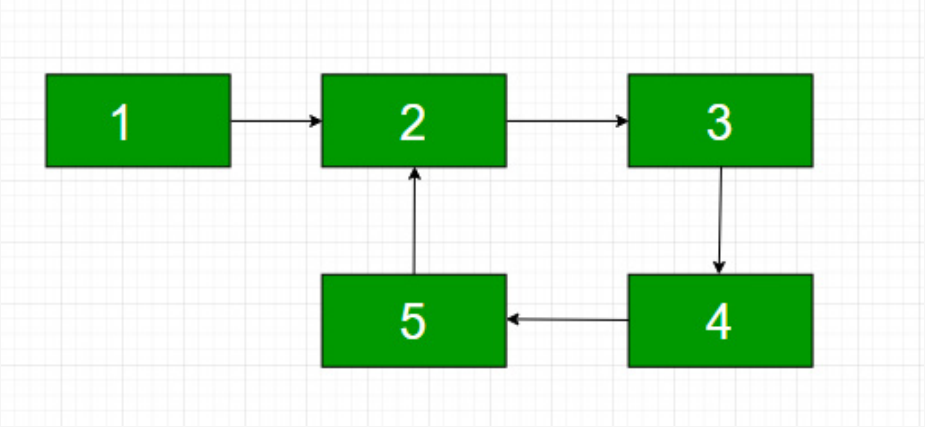
Explanation
4 sorted linked list of size 3, 2, 2, 2
1st list 1 -> 2-> 3
2nd list 4->5
3rd list 5->6
4th list 7->8
The merged list will be 1->2->3->4->5->5->6->7->8

15. Find length of Loop

Given a linked list of size N, your task is to check whether a given Linked List contains loop or not and if loop is present then return the count of nodes in loop or else return 0.

For Example:

Size of the loop in the below linked list is 4.



16. First non-repeating character in a stream

Given an input stream of n characters consisting only of small case alphabets the task is to find the first non-repeating character each time a character is inserted to the stream. If no non-repeating element is found print -1.

Flow in stream : a, a, b, c

a goes to stream : 1st non repeating element a (a)

a goes to stream : no non repeating element -1 (a, a)

b goes to stream : 1st non repeating element is b (a, a, b)

c goes to stream : 1st non repeating element is b (a, a, b, c)

17. XOR Linked List

An ordinary Doubly Linked List requires space for two address fields to store the addresses of previous and next nodes. A memory efficient version of Doubly Linked List can be created using only one space for address field with every node. This memory efficient Doubly Linked List is called XOR Linked List or Memory Efficient as the list uses bit-wise XOR operation to save space for one address.

XOR List Representation:
Let us call the address variable in XOR representation npx (XOR of next and previous)
Node A:
npx = 0 XOR add(B) // bitwise XOR of zero and address of B

Node B:
npx = add(A) XOR add(C) // bitwise XOR of address of A and address of C

Node C:
npx = add(B) XOR add(D) // bitwise XOR of address of B and address of D

Node D:
npx = add(C) XOR 0 // bitwise XOR of address of C and 0

18. Linked List in Zig-Zag fashion
Given a Linked list, rearrange it such that converted list should be of the form a < b > c < d > e <f .. where a, b, c are consecutive data node of linked list and such that the order of linked list is sustained.
For example:
In 11 15 20 5 10 we consider only 11 20 5 15 10 because it satisfies the above condition and the order of linked list. 5 20 11 15 10 is not considered as it does not follow the order of linked list.

19. Implement Railway Reservation System using the concept of linked list.
20. Implement Banking System using the concept of linked list.
21. Implement University Student Admission System using the concept of linked list.
22. Implement Library Management System using the concept of linked list..
23. Implement Tourism Industry System using the concept of linked list.
24. Implement Student Record Management System using the concept of linked list.

CONTACT THE FOLLOWING FACULTY MEMBER ACCORDING TO PROJECT NUMBER IF PROJECT CORRESPONDING ANY QUEURY IS THERE

PROJECT NUMBER	CONCERNED FACULTY
1-6	SUDIPTO KUMAR MONDAL
7-12	SREYASHI BHATTACHARYA
13-18	SUMAN BHATTACHARJEE
19-24	SUSETA DUTTA