MCA Department

Batch - 1

2022-23

1 Write a program to compare two strings stored in singly linked list.
[Given two strings, represented as linked lists (every character is a node in a linked list). Write a function compare () that works similar to strcmp(), i.e., it returns 0 if both strings are the same, 1 if the first linked list is lexicographically greater, and -1 if the second string is lexicographically greater.]

2 Write a program to implement stack using linked list which converts infix to prefix.

3 Write a Program to Reverse a Linked List in groups of given size
Given a linked list, write a function to reverse every k node (where k is an input to the function).
Example:
Input: 1->2->3->4->5->6->7->8->NULL, K = 3
Output: 3->2->1->6->5->4->8->7->NULL
Input: 1->2->3->4->5->6->7->8->NULL, K = 5
Output: 5->4->3->2->1->8->7->6->NULL]

4 Write a program to implement a phone directory using a singly circular linked list with following operations. Node has info like cust_id, name, phone_number.

- Insert from first
- Insert from last
- Insert at specific position
- Delete from specific position
- Delete from first
- Delete from last
- Display in sorted order
- Search by name
- Search by cust_id
- Search by phone_number
- Delete by name
- Delete by cust_id
- Delete by phone_number

5 Write a program to implement priority queue using linked list.

MCA Department

Batch - 2

2022-23

1 Write a Program to Merge a linked list into another linked list at alternate positions
[Given two linked lists, insert nodes of second list into first list at alternate positions of first list. For example, if first list is 5->7->17->13->11 and second is 12->10->2->4->6, the first list should become 5->12->7->10->17->2->13->4->11->6 and second list should become empty. The nodes of the second list should only be inserted when there are positions available. For example, if the first list is 1->2->3 and second list is 4->5->6->7->8, then first list should become 1->4->2->5->3->6 and second list to 7->8.]

2 Write a program to implement stack using linked list which converts infix to postfix.

3 Write a program to find Union and Intersection of two Linked Lists
[Given two Linked Lists, create union and intersection lists that contain union and intersection of the elements present in the given lists. Order of elements in output lists doesn't matter.
Example:
Input:
List1: 10->15->4->20
lsit2: 8->4->2->10
Output:
Intersection List: 4->10
Union List: 2->8->20->4->15->10]

4 Write a program to implement a phone directory using a singly circular linked list with following operations. Node has info like cust_id, name, phone_number.

- Insert from first
- Insert from last
- Insert at specific position
- Delete from specific position
- Delete from first
- Delete from last
- Display in sorted order
- Search by name
- Search by cust_id
- Search by phone_number
- Delete by name
- Delete by cust_id
- Delete by phone_number

5 Write a program to implement priority queue using linked list.

MCA Department

Batch - 3

2022-23

1 Write a Program for a given a singly linked list, to swap elements pairwise.
[Explanation and example:
If a linked list is 1 → 2 → 3 → 4 → 5
Then the output will be: 2 → 1 → 4 → 3 → 5
If the linked list is 1 → 2 → 3 → 4 → 5 → 6
Then the output will be: 2 → 1 → 4 → 3 → 6 → 5]

2 Write a program to implement stack using linked list which converts infix to prefix.

3 Write a Program to Reverse a Linked List in groups of given size [Given a linked list, write a function to reverse every k nodes (where k is an input to the function).
Example:
Input: 1->2->3->4->5->6->7->8->NULL, K = 3
Output: 3->2->1->6->5->4->8->7->NULL
Input: 1->2->3->4->5->6->7->8->NULL, K = 5
Output: 5->4->3->2->1->8->7->6->NULL]

4 Write a program to implement a phone directory using a singly circular linked list with following operations. Node has info like cust_id, name, phone_number.

- Insert from first
- Insert from last
- Insert at specific position
- Delete from specific position
- Delete from first
- Delete from last
- Display in sorted order
- Search by name
- Search by cust_id
- Search by phone_number
- Delete by name
- Delete by cust_id
- Delete by phone_number

5 Write a program to implement priority queue using linked list.

MCA Department

Batch - 4

2022-23

1 Write a program to Append Last N Nodes to First in the Linked List [Given a linked list and an integer n, append the last n elements of the LL to front. Assume given n will be smaller than length of LL. [Input format: Line 1: Linked list elements (separated by space and terminated by -1
Sample Input 1 :
1 2 3 4 5 -1
3
Sample Output 1 :
3 4 5 1 2

2 Write a program to implement stack using linked list which converts infix to postfix.

3 Write a program to find Union and Intersection of two Linked Lists [Given two Linked Lists, create union and intersection lists that contain union and intersection of the elements present in the given lists. Order of elements in output lists doesn't matter.
Example:
Input:
List1: 10->15->4->20
lsit2: 8->4->2->10
Output:
Intersection List: 4->10
Union List: 2->8->20->4->15->10]

4 Write a program to implement a phone directory using a singly circular linked list with following operations. Node has info like cust_id, name, phone_number.

- Insert from first
- Insert from last
- Insert at directory sorting position based on cust_id
- Delete from specific position
- Delete from first
- Delete from last
- Display in sorted order
- Search by name
- Search by cust_id
- Search by phone_number
- Delete by name
- Delete by cust_id
- Delete by phone_number

5 Write a program to implement priority queue using linked list.

MCA Department

Batch - 1,2,3,4

2022-23

1 For this program, you will generate two different types of graphs and compute using them. [Generate from provided two files]

File format

- Input will be based on file.
- Assume vertices are numbered 0..n-1.
- In this case, we will assume each file contains exactly one graph.
- Every graph has a two line "header".
  - Line 1: isDirected isWeighted
  - Line 2: n m

  On line 1, if isDirected==0 the graph is undirected, else it is directed. If isWeighted==0 the graph is unweighted else it is weighted.
  On line 2, n is the number of vertices (nodes) and m is the number of edges.

The next m lines contain information about the edges. If the graph isWeighted, the next m lines each contain three integers, u, v and w, where u and v are the endpoints of the edge and w is the weight on that edge. If the graph is not weighted, each of the m lines contains only u and v. If the graph is undirected, there is an edge (u, v) and an edge (v, u). If the graph isDirected, the edge is from u to v.

```
IF Directed (and Unweighted)
      Generate Adjacency Matrix
      Instantiate directed graph
      Traverse the graph with DFS and BFS
Else Undirected
      Generate Adjacency List
      Instantiate undirected graph
      Traverse the graph with DFS and BFS
```