Course code: CSL 201 Course Name : Data Structures Lab

Faculty In Charge: Dr Binu V P

L-T-P-Credits 0-0-3-2

Pre-requisite: CST 201 Data Structures, EST 102 C programming skills.

Operating System to Use in Lab : Linux

Compiler/Software to Use in Lab : gcc

Programming Language to Use in Lab : Ansi C

Preamble: The aim of the Course is to give hands-on experience for Learners on creating and using different Data Structures. Data Structures are used to process data and arrange data in different formats for many applications. The most commonly performed operations on data structures are traversing, searching, inserting, deleting and few special operations like merging and sorting.

Lab Cycle-4

Learning Outcome: Learn and implement various operations on single linked list.

Date of submission: on or before 7-1-2022

(Write these programs in fair record-show the output in lab and get it signed by the staff in charge. There will be viva voce in every lab.)

Linked lists were developed in 1955-56 by Allen Newell, Cliff Shaw and Herbert Simon at RAND Corporation as the primary data structure for their Information Processing Language. IPL was used by the authors to develop several early artificial intelligence programs, including the Logic Theory Machine, the General Problem Solver, and a computer chess program

a **linked list** is one of the fundamental data structures, and can be used to implement other data structures and has got several applications. It consists of a sequence of nodes, each containing arbitrary data fields and one or two references ("links") pointing to the next and/or previous nodes. The principal benefit of a linked list over a conventional array is that the order of the linked items may be different from the order that the data items are stored in memory or on disk, allowing the list of items to be traversed in a different order. A linked list is a self-referential data type because it contains a pointer or link to another datum of the same type. Linked lists permit insertion and removal of nodes at any point in the list in constant time but do not allow random access. Several different types of linked list exist: singly-linked lists, doubly-linked lists, and circularly-linked lists.

Singly-linked list

The simplest kind of linked list is a singly-linked list which has one link per node. This link points to the next node in the list, or to a null value or empty list if it is the final node.

A singly-linked list containing two parts: the data values of the current node and a link to the next node. A singly linked list travels one way.

Write programs to implement the following using singly linked list:

1) Create a linked list with n elements by adding elements at the end.
2) Given a node data, insert a new node after it.
3) Given a node data, insert a new node before it.
4) Insert a new node in the given position.
5) Delete a node, given the key data value.
6) Delete a node given the position.
7) Delete the smallest element from the list.
8) Reverse a list.
9) Search for a given element and print it's position.
10) Create a list in sorted order.

9. Implement a stack using singly linked list (this can be considered as a part of the previous assignment on stack)