KATHMANDU UNIVERSITY

Dhulikhel,Kavrepalanchok

Estd. 1991

COMP-202

LAB 1

Submitted by:

Sudip Subedi

Roll no:52

Group:C.E

Year/sem: II/I

Submitted to:

Dr. Rajani Chulyadyo PhD

Department of Computer

OUTPUT:

```
The Linked list is empty:
True
New Node with data 2 added to empty Linked List
Linked List: 2
A newNode having data 1 added to Head
Linked List: 1 2
newNode with data 3 added to TAIL
Linked List: 1 2 3
newNode with data 6 added to TAIL
Linked List: 1 2 3 6
3 found at index: 3
newNode having data 4 added after predecessor Node with 3
Linked List: 1 2 3 4 6
4 found at index: 4
newNode having data 5 added after predecessor Node with 4
Linked List: 1 2 3 4 5 6
First Node deleted
Linked List: 2 3 4 5 6
4 deleted.
Linked List: 2 3 5 6
3 found return True.
4 not found return False.
The Linked list is empty:
False
```

Once the file is run, the terminal makes a call to compile the source code and header files- main.cpp, linkedList.cpp and linkedList.h. The program gets called through the main function of main.cpp file where different sets of operation takes place. Firstly, it checks if the list is empty and prints True if empty. A temp node pointing to head is created, if the temp equals to null it shows true else false.

A new node having certain data 2 is added to the empty list. A newNode having data 1 is added to head. Traverse is done to retrieve the data in the linked list for multiple instances. A newNode with certain data also added to the list at the tail with the use of tail pointer. We use **`search(int data)`** to search for certain data in the linked list and returns the output accordingly. We can also add data at certain position using predecessor. We can also delete a node unless the list is empty.