**Experiment no 6:** **Singly Linked List**

**Aim:** Implementation of Singly Linked List

**Objective :** It is used to implement stacks and queue which are linked needs throughout computer science .To prevent the Collision between the data in the Hash map.we use a singly Linked list

**Theory :** Linked List can be defined as collection of objects called nodes that are randomly stored in the memory.A node contains two fields i.e. data stored at that particular address and the pointer which contains the address of the next node in the memory.The last node of the list contains pointer to the null.The list is not required to be contiguously present in the memory. The node can reside any where in the memory and linked together to make a list. This achieves optimized utilization of space. List size is limited to the memory size and doesn't need to be declared in advance. Empty node can not be present in the linked list. We can store values of primitive types or objects in the singly linked list.

**Algorithm :\**

Traversing

Step 1: [INITIALIZE] SET PTR = HEAD

Step 2: Repeat Steps 3 and 4 while PTR != NULL

Step 3: Apply process to PTR -> DATA

Step 4: SET PTR = PTR->NEXT

[END OF LOOP]

Step 5: EXIT

Inserting a node at the beginning

Step 1: SET NEW\_NODE = PTR

Step 2: SET PTR = PTR → NEXT

Step 3: SET NEW\_NODE → DATA = VAL

Step 4: SET NEW\_NODE → NEXT = HEAD

Step 5: SET HEAD = NEW\_NODE

Step 6: EXIT

Deleting a node at the beginning

Step 1: IF HEAD = NULL

Write UNDERFLOW

Go to Step 5

[END OF IF]

Step 2: SET PTR = HEAD

Step 3: SET HEAD = HEAD -> NEXT

Step 4: FREE PTR

Step 5: EXIT

**Code :**

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

struct node

{

int data;

struct node \*next;

};

struct node \*head,\*newNode,\*temp;

void create()

{

head = 0;

int choice;

while(choice)

{

newNode = (struct node\*)malloc(sizeof(struct node));

printf("\nEnter Data:");

scanf("%d",&newNode->data);

newNode->next = 0;

if(head==0)

{

head = temp = newNode;

}

else

{

temp->next = newNode;

temp = newNode;

}

printf("Do you want to continue(0/1)?\n");

scanf("\n%d",&choice);

printf("%d",&newNode->data);

}

}

void insertStart()

{

newNode = (struct node \*)malloc(sizeof (struct node));

printf("Enter data:");

scanf("%d",&newNode->data);

newNode->next = head;

head = newNode;

}

void deleteStart()

{

temp = head;

head = head->next;

free (temp);

}

void display()

{

temp = head;

printf("Linked List: ");

while(temp!=NULL){

printf("\t%d",temp->data);

temp = temp->next;

}

printf(" \n");

}

void main()

{

create();

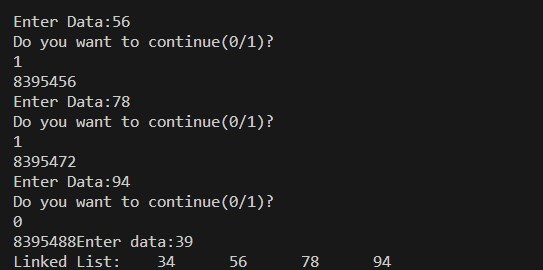
insertStart();

deleteStart();

display();

}

**Output :**

****

**Conclusion :** Unlike arrays, where random access is available, a linked list requires sequential traversal to reach its nodes. A singly linked list is a type of linked list that is unidirectional, that is, it can be traversed in only one direction from head to the last node (tail).