**Aim:** Implement Stack ADT using Linked List.

#include <stdio.h>

#include <stdlib.h>

typedef struct Node {

int data;

struct Node\* next;

} node;

node\* createNode(int data)

{

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

if (newNode == NULL)

return NULL;

newNode->data = data; newNode->next = NULL; return newNode;

}

int insertBeforeHead(node\*\* head, int data)

{

node\* newNode = createNode(data);

if (!newNode)

return-1;

if (\*head == NULL) {

\*head = newNode; return 0;

}

newNode->next = \*head;

\*head = newNode; return 0;

}

int deleteHead(node\*\* head)

{

node\* temp = \*head;

\*head = (\*head)->next; free(temp);

return 0;

}

int isEmpty(node\*\* stack) { return \*stack == NULL; }

void push(node\*\* stack, int data)

{

if (insertBeforeHead(stack, data)) { printf("Stack Overflow!\n");

}

}

int pop(node\*\* stack)

{

if (isEmpty(stack)) { printf("Stack Underflow\n"); return-1;

}

deleteHead(stack);

}

int peek(node\*\* stack)

{

if (!isEmpty(stack)) return (\*stack)->data; else

return-1;

}

void printStack(node\*\* stack)

{

node\* temp = \*stack; while (temp != NULL) {

printf("%d-> ", temp->data); temp = temp->next;

}

printf("\n");

}

int main()

{

node\* stack = NULL;

push(&stack, 10);

push(&stack, 20);

push(&stack, 30);

push(&stack, 40);

push(&stack, 50);

printf("Stack: "); printStack(&stack);

pop(&stack);

pop(&stack);

printf("\nStack: ");

printStack(&stack); return 0;

}