LAB 6

**Question:**

**6b) WAP to Implement Single Link List to simulate Stack &amp; Queue Operations.**

#include <stdio.h>

#include <stdlib.h>

typedef struct node

{

int data;

struct node \*next;

} node;

void push(node \*\*head, int new\_data)

{

node \*new\_node = (node \*)malloc(sizeof(node));

new\_node->data = new\_data;

new\_node->next = NULL;

if (\*head == NULL)

{

\*head = new\_node;

}

else

{

node \*temp = \*head;

while (temp->next != NULL)

{

temp = temp->next;

}

temp->next = new\_node;

}

}

void pop(node \*\*head)

{

if (\*head == NULL)

{

printf("Stack is empty\n");

}

else

{

node \*temp = \*head;

node \*prev = NULL;

while (temp->next != NULL)

{

prev = temp;

temp = temp->next;

}

if (prev == NULL)

{

// Only one element in the list

\*head = NULL;

}

else

{

prev->next = NULL;

}

printf("Popped element: %d\n", temp->data);

free(temp);

}

}

void enqueue(node \*\*front, int new\_data)

{

node \*new\_node = (node \*)malloc(sizeof(node));

new\_node->data = new\_data;

new\_node->next = NULL;

if (\*front == NULL)

{

\*front = new\_node;

}

else

{

node \*temp = \*front;

while (temp->next != NULL)

{

temp = temp->next;

}

temp->next = new\_node;

}

}

void dequeue(node \*\*front)

{

if (\*front == NULL)

{

printf("Queue is empty\n");

}

else

{

node \*temp = \*front;

\*front = temp->next;

printf("Dequeued element: %d\n", temp->data);

free(temp);

}

}

void display(node \*list)

{

node \*current = list;

while (current != NULL)

{

printf("%d ", current->data);

current = current->next;

}

printf("\n");

}

int main()

{

node \*stack = NULL;

node \*queue = NULL;

// Stack operations

push(&stack, 18);

push(&stack, 23);

push(&stack, 98);

push(&stack, 77);

// Display the stack

printf("Stack: ");

display(stack);

// Pop elements from the stack

pop(&stack);

pop(&stack);

pop(&stack);

// Queue operations

enqueue(&queue, 48);

enqueue(&queue, 35);

enqueue(&queue, 96);

// Display the queue

printf("Queue: ");

display(queue);

// Dequeue elements from the queue

dequeue(&queue);

dequeue(&queue);

dequeue(&queue);

return 0;

}

Output:

