ACADEMY OF TECHNOLOGY

NAME : PRIYADARSHAN GHOSH

COLLEGE ROLL NO: 72

UNIVERSITY ROLL NO: 16900319072

DEPARTMENT: ECE-1(Y)

SEMESTER:3rd

PAPER CODE : ES-CS391

➢ Laboratory *Assignment #13*

A) DEFINE FOLLOWING FUNCTIONS TO REPRESENTATION STACK USING LINKED LIST

I) PUSH()

II) POP()

III) DISPLAY()

Ans:

```
#include <stdio.h>
#include <stdlib.h>
void push();
int pop();
void display();

struct node
{
int value;
struct node *next;
};
struct node *head;


void push (int value)
{
    struct node *ptr = (struct node*)malloc(sizeof(struct node));
    if(ptr == NULL)
    {
        printf("\n Overflow Condition.");
    }
    else
    {
        if(head==NULL)
        {
            ptr->value = value;
            ptr -> next = NULL;
            head=ptr;
        }
        else
        {
            ptr->value = value;
            ptr->next = head;
            head=ptr;

        }

    }
}

int pop()
{
    int item;
    struct node *ptr;
    if (head == NULL)
    {
        printf("\n Underflow Condition.");
    }
    else
    {
        item = head->value;
        ptr = head;
        head = head->next;
        free(ptr);
        return item;
    }
}
void display()
{
    int i;
    struct node *ptr;
    ptr=head;
    if(ptr == NULL)
    {
        printf("\n Stack is empty.\n");
    }
    else
    {
        printf("\n The Stack elements are: \n");
        while(ptr!=NULL)
        {
            printf(" %d\n",ptr->value);
            ptr = ptr->next;
        }
    }
}

main ()
{
    int ch=0;
    printf("\n Stack operations using linked list -->\n");
    while(ch != 4)
    {
        printf("\n Press 1 for Push an element: ");
                printf("\n Press 2 for Pop an element: ");
                printf("\n Press 3 for display: ");
                printf("\n Press 4 for exit: ");
        printf("\n Enter your choice: ");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1:
            {
            int value;
            printf("\n Enter the element for push operation: ");
        scanf("%d",&value);
```

```
                push(value);
                break;
            }
            case 2:
            {
            int item;
            item = pop();
            printf("\n The element which is popped : %d", item);
                break;
            }
            case 3:
            {
                display();
                break;
            }
            case 4:
            {
                break;
            }
            default:
            {
                printf("\n Your Choice is Wrong.");
            }
    };
}
}
```

OUTPUT =>

```
Stack operations using linked list -->


Press 1 for Push an element:
Press 2 for Pop an element:
Press 3 for display:
Press 4 for exit:
Enter your choice: 1


Enter the element for push operation: 10


Press 1 for Push an element:
Press 2 for Pop an element:
Press 3 for display:
Press 4 for exit:
Enter your choice: 1


Enter the element for push operation: 20


Press 1 for Push an element:
Press 2 for Pop an element:
Press 3 for display:
Press 4 for exit:
```

```
Enter your choice: 1


Enter the element for push operation: 30


Press 1 for Push an element:

Press 2 for Pop an element:

Press 3 for display:

Press 4 for exit:

Enter your choice: 1


Enter the element for push operation: 40


Press 1 for Push an element:

Press 2 for Pop an element:

Press 3 for display:

Press 4 for exit:

Enter your choice: 1


Enter the element for push operation: 50


Press 1 for Push an element:

Press 2 for Pop an element:

Press 3 for display:

Press 4 for exit:
```

```
Enter your choice: 1


Enter the element for push operation: 60


Press 1 for Push an element:

Press 2 for Pop an element:

Press 3 for display:

Press 4 for exit:

Enter your choice: 3


The Stack elements are:

60

50

40

30

20

10


Press 1 for Push an element:

Press 2 for Pop an element:

Press 3 for display:

Press 4 for exit:

Enter your choice: 2
```

```
The element which is popped : 60

Press 1 for Push an element:

Press 2 for Pop an element:

Press 3 for display:

Press 4 for exit:

Enter your choice: 3


The Stack elements are:

50

40

30

20

10


Press 1 for Push an element:

Press 2 for Pop an element:

Press 3 for display:

Press 4 for exit:

Enter your choice: 2


The element which is popped : 50

Press 1 for Push an element:

Press 2 for Pop an element:

Press 3 for display:
```

```
Press 4 for exit:

Enter your choice: 2


The element which is popped : 40

Press 1 for Push an element:

Press 2 for Pop an element:

Press 3 for display:

Press 4 for exit:

Enter your choice: 2


The element which is popped : 30

Press 1 for Push an element:

Press 2 for Pop an element:

Press 3 for display:

Press 4 for exit:

Enter your choice: 2


The element which is popped : 20

Press 1 for Push an element:

Press 2 for Pop an element:

Press 3 for display:

Press 4 for exit:

Enter your choice: 2
```

```
The element which is popped : 10
Press 1 for Push an element:
Press 2 for Pop an element:
Press 3 for display:
Press 4 for exit:
Enter your choice: 2

Underflow Condition.
The element which is popped : 22
Press 1 for Push an element:
Press 2 for Pop an element:
Press 3 for display:
Press 4 for exit:
Enter your choice: 3

Stack is empty.

Press 1 for Push an element:
Press 2 for Pop an element:
Press 3 for display:
Press 4 for exit:
Enter your choice: 4


-------------------------------
```

```
Process exited after 69.54 seconds with return value 0

Press any key to continue . . .
```

B)DEFINE FOLLOWING FUNCTIONS TO REPRESENTATION LINEAR QUEUE USING LINKED LIST

I) ENQUEUE()

II) DEQUEUE()

III) DISPLAY()

Ans:

```
#include <stdio.h>
#include <stdlib.h>

struct node
{
    int info;
    struct node *ptr;
}*front,*rear,*temp,*front1;

int frontelement();
void enq(int data);
void deq();
void empty();
void display();
void create();
void queuesize();

int count = 0;

main()
{
    int no, ch, e;

    printf("\n 1 - Enque");
    printf("\n 2 - Deque");
    printf("\n 3 - Front element");
    printf("\n 4 - Empty");
    printf("\n 5 - Exit");
    printf("\n 6 - Display");
    printf("\n 7 - Queue size");
    create();
    while (1)
    {
        printf("\n Enter choice : ");
        scanf("%d", &ch);
        switch (ch)
        {
        case 1:
            printf("Enter data : ");
            scanf("%d", &no);
            enq(no);
            break;
        case 2:
            deq();
            break;
        case 3:
            e = frontelement();
            if (e != 0)
                printf("Front element : %d", e);
            else
                printf("\n No front element in Queue as queue is empty");
            break;
        case 4:
            empty();
            break;
        case 5:
            exit(0);
        case 6:
            display();
            break;
        case 7:
            queuesize();
            break;
        default:
            printf("Wrong choice, Please enter correct choice  ");
            break;
```

```
        }
    }
}


void create()
{
    front = rear = NULL;
}


void queuesize()
{
    printf("\n Queue size : %d", count);
}


void enq(int data)
{
    if (rear == NULL)
    {
        rear = (struct node *)malloc(1*sizeof(struct node));
        rear->ptr = NULL;
        rear->info = data;
        front = rear;
    }
    else
    {
        temp=(struct node *)malloc(1*sizeof(struct node));
        rear->ptr = temp;
        temp->info = data;
        temp->ptr = NULL;

        rear = temp;
    }
    count++;
}

void display()
{
    front1 = front;

    if ((front1 == NULL) && (rear == NULL))
    {
        printf("Queue is empty");
        return;
    }
    while (front1 != rear)
    {
        printf("%d ", front1->info);
        front1 = front1->ptr;
    }
    if (front1 == rear)
        printf("%d", front1->info);
}



void deq()
{
    front1 = front;

    if (front1 == NULL)
    {
        printf("\n Error: Trying to display elements from empty queue");
        return;
    }
    else
        if (front1->ptr != NULL)
        {
            front1 = front1->ptr;
            printf("\n Dequed value : %d", front->info);
            free(front);
            front = front1;
        }
        else
        {
            printf("\n Dequed value : %d", front->info);
            free(front);
            front = NULL;
            rear = NULL;
        }
        count--;
}


int frontelement()
{
    if ((front != NULL) && (rear != NULL))
        return(front->info);
    else
        return 0;
}


void empty()
{
     if ((front == NULL) && (rear == NULL))
        printf("\n Queue empty");
    else
       printf("Queue not empty");
}
```

***OUTPUT* =>**

```
 1 - Enque
 2 - Deque
 3 - Front element
 4 - Empty
 5 - Exit
 6 - Display
 7 - Queue size
 Enter choice : 1
Enter data : 20

 Enter choice : 1
Enter data : 30

 Enter choice : 1
Enter data : 40

 Enter choice : 1
Enter data : 50

 Enter choice : 1
Enter data : 60

 Enter choice : 1
Enter data : 70

 Enter choice : 1
Enter data : 80

 Enter choice : 1
Enter data : 90

 Enter choice : 6
20 30 40 50 60 70 80 90
 Enter choice : 3
Front element : 20
 Enter choice : 4
Queue not empty
 Enter choice : 7

 Queue size : 8
 Enter choice : 6
20 30 40 50 60 70 80 90
 Enter choice : 2

 Dequed value : 20
 Enter choice : 2

 Dequed value : 30
 Enter choice : 2

 Dequed value : 40
 Enter choice : 2
 Dequed value : 50
 Enter choice : 2

 Dequed value : 60
 Enter choice : 2

 Dequed value : 70
 Enter choice : 2

 Dequed value : 80
 Enter choice : 2

 Dequed value : 90
 Enter choice : 2

 Error: Trying to display elements from empty queue
 Enter choice : 6
Queue is empty
 Enter choice : 7

 Queue size : 0
 Enter choice : 5
--------------------------------
Process exited after 109.7 seconds with return value 0
Press any key to continue . . .
```