|  |  |
| --- | --- |
|  | **DEPARTMENT OF COMPUTER ENGINEERING** |

**Experiment No. 11 B**

|  |  |
| --- | --- |
| Semester | S.E. Semester III – Computer Engineering |
| Subject | Data Structures Lab (CSL301) |
| Subject Professor In-charge | Prof. Swapnil S. Sonawane |
| Assisting Teachers | Prof. Swapnil S. Sonawane |

|  |  |
| --- | --- |
| Student Name | Trisha Shah |
| Roll Number | 20102A0004 |

**Title:**

Implement Linear Queue ADT using Linked List.

**Objective:**

Students will be able to handle various operations like searching, insertion, deletion and traversals on various data structures.

**Explanation:**

The storage requirement of linked representation of a queue with n elements is o(n) while the time requirement for operations is o(1).

In a linked queue, each node of the queue consists of two parts i.e. data part and the link part. Each element of the queue points to its immediate next element in the memory.

In the linked queue, there are two pointers maintained in the memory i.e. front pointer and rear pointer. The front pointer contains the address of the starting element of the queue while the rear pointer contains the address of the last element of the queue.

Insertion and deletions are performed at rear and front end respectively. If front and rear both are NULL, it indicates that the queue is empty.

**Program Code:**

#include<stdio.h>

#include<stdlib.h>

typedef struct node

{

int data;

struct node \*next;

}node;

typedef struct LL

{

node \*start;

}LL;

void display(LL \*l)

{

node \*p;

if(l->start==NULL)

{

printf("\nList is empty...");

}

else

{

p=l->start;

while(p!=NULL)

{

printf("\n%d",p->data);

p=p->next;

}

}

}

void enqueue(LL \*l,int x)

{

node \*newrec,\*p;

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

newrec->data=x;

newrec->next=NULL;

if(l->start==NULL)

{

l->start=newrec;

}

else

{

p=l->start;

while(p->next!=NULL)

{

p=p->next;

}

p->next=newrec;

}

}

void dequeue(LL \*l)

{

node \*p;

if(l->start==NULL)

{

printf("\nDeletion not possible...");

}

else

{

p=l->start;

l->start=l->start->next;

free(p);

}

}

int main()

{

int ch,x;

LL l;

l.start=NULL;

while(1)

{

printf("\nMenu:\n1-ENQUEUE\n2-DEQUEUE\n3-DISPLAY\n4-EXIT\nEnter Choice=");

scanf("%d",&ch);

if(ch==4)

break;

switch(ch)

{

case 1:

{

printf("\nEnter element to be inserted=");

scanf("%d",&x);

enqueue(&l,x);

display(&l);

}

break;

case 2:

{

dequeue(&l);

display(&l);

}

break;

case 3:

{

display(&l);

}

break;

default:

{

printf("\nInvalid Choice...");

}

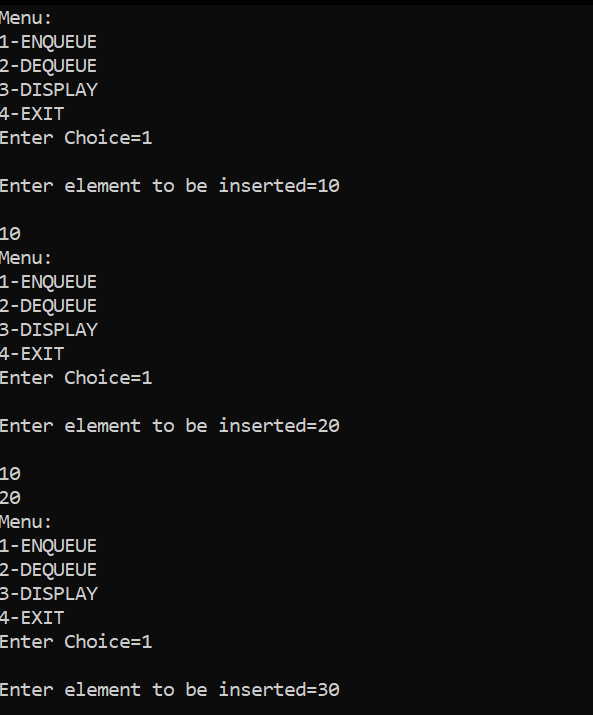
}

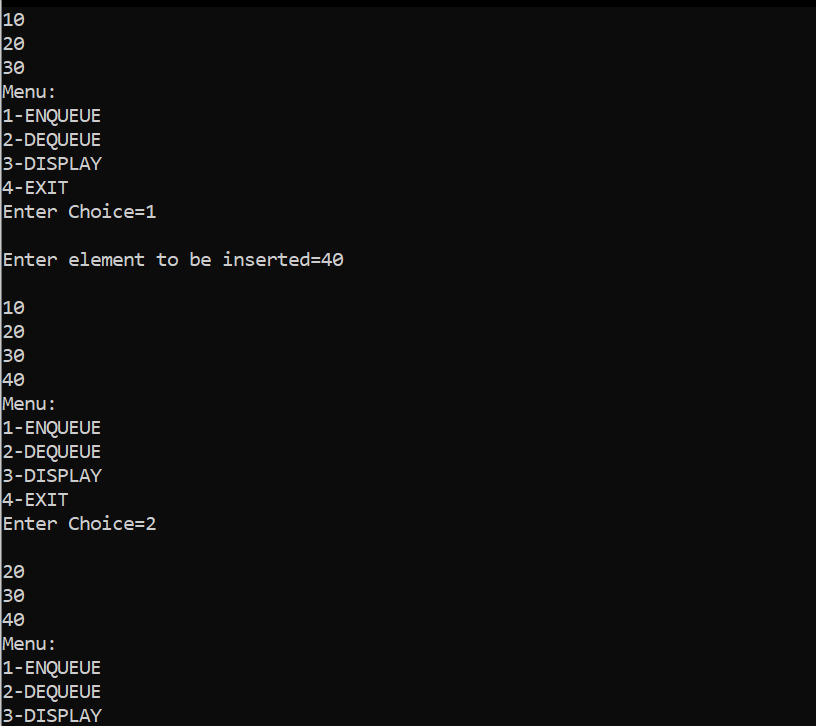
}

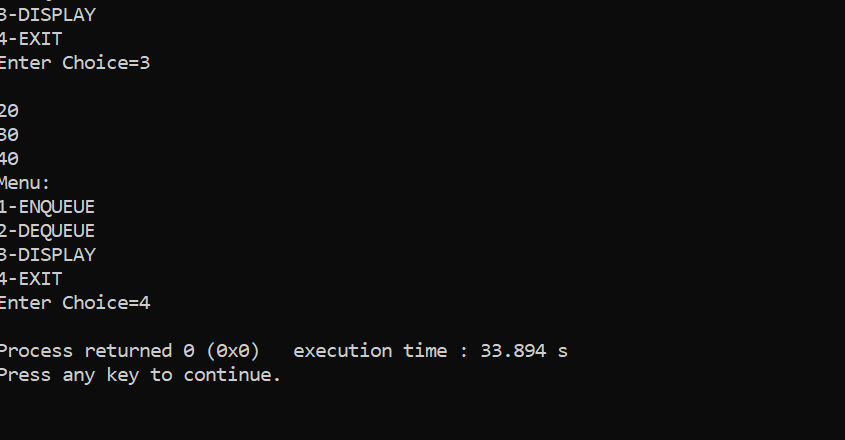
return 0;

}

**Output:**

****

****

****

**Conclusion:**

Through this experiment, students were able to successfully implement a queue using linkedlist.