**Experiment No-6B**

**Aim**: - Write a program in C to implement linear queue using linked list.

**Theory**: -

In a [Queue data structure](https://www.geeksforgeeks.org/queue-data-structure/), we maintain two pointers, front and rear. The front points the first item of queue and rear points to last item.

**enQueue(): -** This operation adds a new node after rear and moves rear to the next node.

**deQueue(): -** This operation removes the front node and moves front to the next node.

**CODE:-**

#include<stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node \*next;

};

struct queue

{

struct node \*front,\*rear;

};

struct queue\*createqueue()

{

struct queue \*q=(struct queue \*)malloc(sizeof(struct queue));

q->front=q->rear=NULL;

return q;

}

struct node\*createnode(int ele)

{

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

ptr->data=ele;

ptr->next=NULL;

return ptr;

}

void insert(struct queue \*q,int e)

{

struct node \*ptr=createnode(e);

if(q->front==NULL)

{

q->front=ptr;

q->rear=ptr;

q->front->next=q->rear->next=NULL;

}

else

{

q->rear->next=ptr;

q->rear=ptr;

q->rear->next=NULL;

}

}

void del(struct queue \*q)

{

struct node \*ptr=q->front;

if(ptr==NULL)

printf("Queue is Empty(Underflow Condition)\n");

else

{

q->front=q->front->next;

printf("The deleted element is %d\n",ptr->data);

free(ptr);

}

}

void display(struct queue \*q)

{

struct node \*ptr=q->front;

if(ptr==NULL)

printf("Stack is Empty\n");

else

{

while(ptr!=NULL)

{

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

ptr=ptr->next;

}

}

}

int main(void)

{

int ch,val;

struct queue \*q=createqueue();

do

{

printf("Enter your choice:\n");

printf("1.Insert\n2.Delete\n3.Display\n");

scanf("%d",&ch);

switch(ch)

{

case 1:

printf("Enter the element:\n");

scanf("%d",&val);

insert(q,val);

break;

case 2:

del(q);

break;

case 3:

display(q);

break;

default:

printf("Invalid Input\n");

}

}while(ch<4);

return 0;

}

**// ALTERNATE CODE: -**

#include<stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node \*next;

struct node \*front,\*rear;

};

struct node \*createqueue()

{

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

q->front=q->rear=NULL;

return q;

}

struct node \*createnode(int ele)

{

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

ptr->data=ele;

ptr->next=NULL;

return ptr;

}

void insert(struct node \*q,int e)

{

struct node \*ptr= createnode(e);

if(q->front==NULL)

{

q->front=ptr;

q->rear=ptr;

q->front->next=q->rear->next=NULL;

}

else

{

q->rear->next=ptr;

q->rear=ptr;

q->rear->next=NULL;

}

}

void del(struct node \*q)

{

struct node \*ptr=q->front;

if(ptr==NULL)

printf("Queue is Empty(Underflow Condition)\n");

else

{

q->front=q->front->next;

printf("The deleted element is %d\n",ptr->data);

free(ptr);

}

}

void display(struct node \*q)

{

struct node \*ptr=q->front;

if(ptr==NULL)

printf("Stack is Empty\n");

else

{

while(ptr!=NULL)

{

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

ptr=ptr->next;

}

}

}

int main(void)

{

int ch,val;

struct node \*q= createqueue();

do

{

printf("Enter your choice:\n");

printf("1.Insert\n2.Delete\n3.Display\n");

scanf("%d",&ch);

switch(ch)

{

case 1:

printf("Enter the element:\n");

scanf("%d",&val);

insert(q,val);

break;

case 2:

del(q);

break;

case 3:

display(q);

break;

default:

printf("Invalid Input\n");

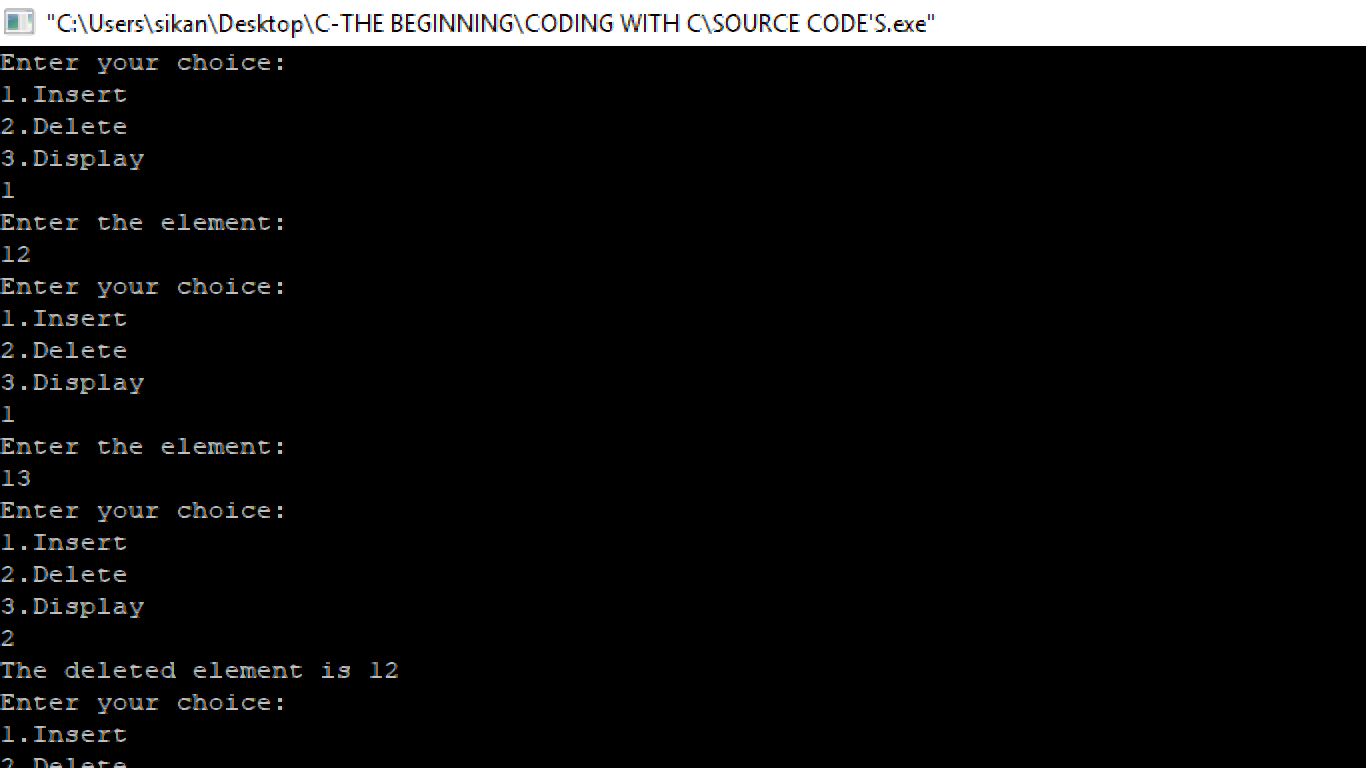
}

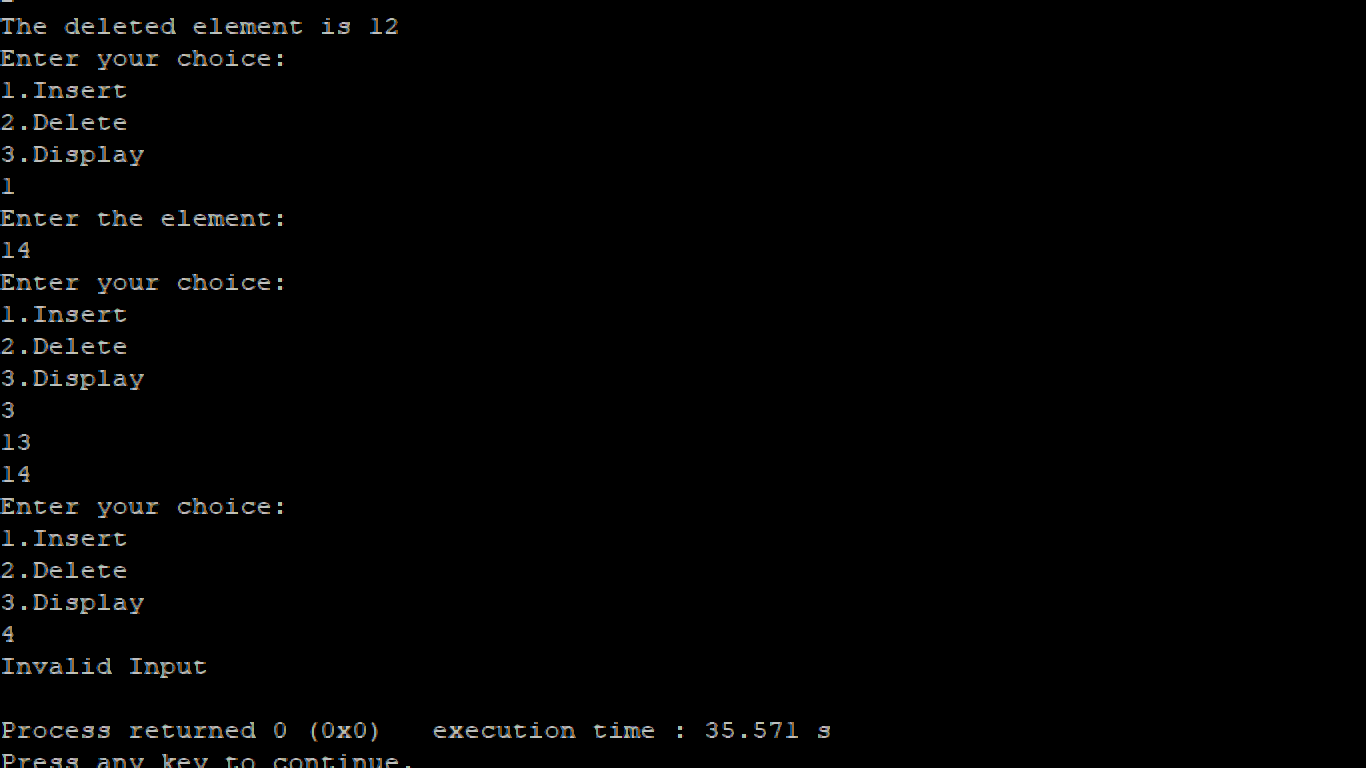
}while(ch<4);

return 0;

}

**OUTPUT: -**

****

****