

NAME: srijan Shankar s
REG NO : 230701341

EX 7 IMPLEMENTATION OF QUEUE USING ARRAY AND LINKED LIST

Queue using stack

```
#include <stdio.h>
#define MAX 5
int Queue[MAX], front = -1, rear = -1;
int IsFull();
int IsEmpty();
void Enqueue(int ele);
void Dequeue();
void Display();
int main()
{
    int ch, e;
    do
    {
        printf("1.ENQUEUE 2.DEQUEUE 3.DISPLAY 4.EXIT");
        printf("\nEnter your choice : ");
        scanf("%d", &ch);
        switch(ch)
        {
            case 1:
                printf("Enter the element : ");
                scanf("%d", &e);
                Enqueue(e);
                break;
            case 2:
                Dequeue();
                break;
            case 3:
                Display();
                break;
        }
    } while(ch <= 3);
    return 0;
}
int IsFull()
{
    if(rear == MAX - 1)
```

```

    return 1;
else
    return 0;
}
int IsEmpty()
{
if(front == -1)
    return 1;
else
    return 0;
}
void Enqueue(int ele)
{
if(IsFull())
    printf("Queue is Overflow...\n");
else
    {
        rear = rear + 1;
        Queue[rear] = ele;
        if(front == -1)
            front = 0;
    }
}
void Dequeue()
{
if(IsEmpty())
    printf("Queue is Underflow...\n");
else
    {
        printf("%d\n", Queue[front]);
        if(front == rear)
            front = rear = -1;
        else
            front = front + 1;
    }
}
void Display()
{
int i;
if(IsEmpty())
    printf("Queue is Underflow...\n");
else
    {
        for(i = front; i <= rear; i++)

```

```

printf("%d\t", Queue[i]);
printf("\n");
}
}

```

Queue using linked list

```

#include <stdio.h>
#include <stdlib.h>
struct node
{
int Element;
struct node *Next;
}*Front = NULL, *Rear = NULL;
typedef struct node Queue;
int IsEmpty(Queue *List);
void Enqueue(int e);
void Dequeue();
void Display();
int main()
{
int ch, e;
do
{
printf("1.ENQUEUE 2.DEQUEUE 3.DISPLAY 4.EXIT");
printf("\nEnter your choice : ");
scanf("%d", &ch);
switch(ch)
{
case 1:
printf("Enter the element : ");
scanf("%d", &e);
Enqueue(e);
break;
case 2:
Dequeue();
break;
case 3:
Display();
break;
}
} while(ch <= 3);

```

```

return 0;
}
int IsEmpty(Queue *List)
{
if(List == NULL)
    return 1;
else
    return 0;
}
void Enqueue(int e)
{
Queue *NewNode = malloc(sizeof(Queue));
    NewNode->Element = e;
    NewNode->Next = NULL;
if(Rear == NULL)
    Front = Rear = NewNode;
else
{
    Rear->Next = NewNode;
    Rear = NewNode;
}
}
void Dequeue()
{
if(IsEmpty(Front))
    printf("Queue is Underflow...\n");
else
{
    Queue *TempNode;
    TempNode = Front;
    if(Front == Rear)
        Front = Rear = NULL;
    else
        Front = Front->Next;
    printf("%d\n", TempNode->Element);
    free(TempNode);
}
}
void Display()
{
if(IsEmpty(Front))
    printf("Queue is Underflow...\n");
else
{

```

```
Queue *Position;  
Position = Front;  
while(Position != NULL)  
{  
    printf("%d\t", Position->Element);  
    Position = Position->Next;  
}  
printf("\n");  
}  
}
```