Experiment No 4
Problem Statements

1.Write a program to implement queue using array.

```
//1.Write a program to
implement queue using array.
/*Name:- Anuj Rajendra Mane
ROll No:- 65
Div:-A
Subject:- Data Structures*/
#include<stdio.h>
#include<stdlib.h>
int queue[5],f=-1,r=-1;
void rear();
void front();
void show();
int main() {
   int ch;
   printf("1.Rear\n");
   printf("2.Front\n");
   printf("3.Show\n");
   printf("4.Exit\n");
   while(1)
   {
      printf("Enter Choice:\n");
      scanf("%d",&ch);
      switch(ch)
      {
         case 1: rear();
         break;
         case 2: front();
         break;
         case 3: show();
         break;
         case 4: exit(0);
         break;
         default :
         printf("Invalid Choice\n");
      }
   }
}
void rear() {
   int item;
   if(r==5-1) {
      printf("Queue is full\n");
   }
   else
   {
      if(f==-1)
      {
         f=0;
      }
```

```
        printf("Insert Element in Queue:\n");
        scanf("%d",&item);
        r=r+1;
        queue[r] = item;
    }
}
void front() {
    if(f==-1)
    {
        printf("Queue is empty\n");
    }
    else
    {
        printf("Deleted: %d\n",queue[f]);
        f=f+1;
    }
}
void show() {
    int i;
    if(f==-1)
    {
        printf("Queue is empty\n");
    }
    else
    {
        printf("Queue Elements Are:\n");
        for(i=f;i<=r;i++)
        {
            printf("%d\n",queue[i]);
        }
    }
}
```

2. Write a program to implement a circular queue using arrays.

```
//2.Write a program to implement a circular queue using arrays.

/*Name:- Anuj Rajendra Mane
ROll No:- 65
Div:-A
Subject:- Data Structures*/

#include <stdio.h>
#include <stdlib.h>
#define size 5
void addition();
void deletion();
void display();
int queue[size];
```

```
int front = -1, rear = -1;
void main()
{
   int choice;
   while (1)
   {
      printf("\nIMPLEMENTATION
OF CIRCULAR QUEUE");
      printf("\n----------------------------
-----");
      printf("\n1. Insert");
      printf("\n2. Delete");
      printf("\n3. Display");
      printf("\n4. Exit");
      printf("\n----------------------------
-----");
      printf("\n\nEnter your choice
[1/2/3/4] : ");
      scanf("%d", &choice);
      switch (choice)
      {
      case 1:
         addition();
         break;
      case 2:
         deletion();
         break;
      case 3:
         display();
         break;
      case 4:
         exit(0);
      default:
         printf("\nInvalid choice");
      }
   }
}
// Function to insert element in
the Circular Queue
void addition()
{
   int num;
   if (front == (rear + 1) % size) //
If(front is 0 and 0 the  queue is
full)
   {
      printf("\nQueue is Full");
      return;
   }
   printf("\nEnter the element to
be inserted in circular queue : ");
   scanf("%d", &num);
   if (front == -1)
   {
      front = rear = 0;
   }
```

```
    else
    {
        rear = (rear + 1) %
size;//0+1%5==1
        queue[rear] =
num;//inserting num at
queue[rear]
    }
}
// Function to delete element
from the circular queue
void deletion()
{
    int num;
    if (front == -1)
    {
        printf("\nQueue is empty");
        return;
    }
    num = queue[front];
    printf("\nDeleted element from
circular queue is : %d", num);
    if (front == rear)
        front = rear = -1;
    else
        front = (front + 1) % size;
}
// Function to display circular
queue
void display()
{
    int i;
    if (front == -1)
    {
        printf("\nQueue is empty");
        return;
    }
    printf("\n\nCircular Queue
elements are : \n");
    for (i = front; i <= rear;
i++)//going front to rear
        printf("\nqueue[%d] : %d", i,
queue[i]);
    if (front > rear)
    {
        for (i = 0; i <= rear; i++)
            printf("queue[%d] : %d\n", i,
queue[i]);
        for (i = front; i < size; i++)
            printf("queue[%d] : %d\n", i,
queue[i]);
    }
}
```

3. Write a program to implement double ended queue (dequeue) using arrays.

```
// 3.Write a program to implement double ended queue (dequeue) using arrays.
/*Name:- Anuj Rajendra Mane
ROll No:- 65
Div:-A
Subject:- Data Structures*/

#include <stdio.h>
#define MAX 10
int deque[MAX];
int left = -1, right = -1;
void insert_right()
{
   int val;
   printf("\nEnter the value to be added ");
   scanf("%d", &val);
   if ((left == 0 && right == MAX - 1) || (left == right + 1))
   {
      printf("\nOVERFLOW");
   }
   if (left == -1) // if queue is initially empty
   {
      left = 0;
      right = 0;
   }
   else
   {
      if (right == MAX - 1)
         right = 0;
      else
         right = right + 1;
   }
   deque[right] = val;
}
void insert_left()
{
   int val;
   printf("\nEnter the value to be added ");
   scanf("%d", &val);
   if ((left == 0 && right == MAX - 1) || (left == right + 1))
   {
      printf("\nOVERFLOW");
   }
```

```
    if (left == -1) // if queue is
initially empty
    {
        left = 0;
        right = 0;
    }
    else
    {
        if (left == 0)
            left = MAX - 1;
        else
            left = left - 1;
    }
    deque[left] = val;
}
//-------DELETE FROM RIGHT-------
void delete_right()
{
    if (left == -1)
    {
        printf("\nUNDERFLOW");
        return;
    }
    printf("\nThe deleted element
is %d\n", deque[right]);
    if (left == right) // Queue has
only one element
    {
        left = -1;
        right = -1;
    }
    else
    {
        if (right == 0)
            right = MAX - 1;
        else
            right = right - 1;
    }
}
//-------DELETE FROM LEFT-------
void delete_left()
{
    if (left == -1)
    {
        printf("\nUNDERFLOW");
        return;
    }
    printf("\nThe deleted element
is %d\n", deque[left]);
    if (left == right) // Queue has
only one element
    {
        left = -1;
        right = -1;
```

```
    }
    else
    {
        if (left == MAX - 1)
            left = 0;
        else
            left = left + 1;
    }
}
//-------DISPLAY-------
void display()
{
    int front = left, rear = right;
    if (front == -1)
    {
        printf("\nQueue is Empty\n");
        return;
    }
    printf("\nThe elements in the
queue are: ");
    if (front <= rear)
    {
        while (front <= rear)
        {
            printf("%d\t", deque[front]);
            front++;
        }
    }
    else
    {
        while (front <= MAX - 1)
        {
            printf("%d\t", deque[front]);
            front++;
        }
        front = 0;
        while (front <= rear)
        {
            printf("%d\t", deque[front]);
            front++;
        }
    }
    printf("\n");
}
int main()
{
    int choice;
    do
    {
        printf("\n1.Insert at right ");
        printf("\n2.Insert at left ");
        printf("\n3.Delete from right
");
        printf("\n4.Delete from left ");
```

```
        printf("\n5.Display ");
        printf("\n6.Exit");
        printf("\n\nEnter your choice
");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
           insert_right();
           break;
        case 2:
           insert_left();
           break;
        case 3:
           delete_right();
           break;
        case 4:
           delete_left();
           break;
        case 5:
           display();
           break;
        }
    } while (choice != 6);
    return 0;
}
```