```c
#include <stdio.h>
#define size 5
int queue[size],front=-1,rear=-1;
void enqueue(int a){
  if(rear==size-1){
    printf("Queue is full/overflow\n");
    return;
  }
  else if(front==-1 && rear==-1){
    front=0;
    rear=0;
  }
  else{
    rear=rear+1;
  }
  queue[rear]=a;
}
int dequeue(){
  int a;
  if((front==-1 && rear==-1)||front>rear){
    printf("Queue is empty/underflow\n");
  }
  else{
    a=queue[front];
    front++;
  }
  return a;
}
void display(){
  if((front==-1 && rear==-1)||front>rear){
    printf("Queue is empty/underflow\n");
  }
  else{
    for(int i=front;i<=rear;i++){
      printf("%d\t",queue[i]);
    }
  }
```

```
    front++;
  }
  return a;
}
void display(){
  if((front==-1 && rear==-1)||front>rear){
    printf("Queue is empty/underflow\n");
  }
  else{
    for(int i=front;i<=rear;i++){
      printf("%d\t",queue[i]);
    }
  }
}
void main(){
  int op,n;
  while(1){
    printf("\nEnter 1.Enqueue\n2.Dequeue\n3.Display\n4.-1 to stop exicution\n");
    scanf("%d",&op);
    if(op==-1){
      break;
    }
    switch(op){
      case 1:printf("Enter no\n");
             scanf("%d",&n);
             enqueue(n);
             break;
      case 2:n=dequeue();
             printf("%d is Dequeued\n",n);
             break;
      case 3:display();
             break;
      default:printf("Invalid choice\n");
    }
  }
}
```

Output

```
/tmp/h8G6brZRUk.o
1.INSERT 2.DELETE 3.DISPLAY 4.EXIT:
1
Enter a value:1
1.INSERT 2.DELETE 3.DISPLAY 4.EXIT:
1
Enter a value:2
1.INSERT 2.DELETE 3.DISPLAY 4.EXIT:
3
12
1.INSERT 2.DELETE 3.DISPLAY 4.EXIT:
1
Enter a value:4
1.INSERT 2.DELETE 3.DISPLAY 4.EXIT:
2
1 deleted
1.INSERT 2.DELETE 3.DISPLAY 4.EXIT:
3
24
1.INSERT 2.DELETE 3.DISPLAY 4.EXIT:
4
```

08/01/2024

SURYA Gold
Date Page

11

1. Write a program to simulate the working of the queue of integers using an array. Provide the following operations: insert, delete, display. The program should print appropriate message for overflow and underflow condition

```
#include <stdio.h>
#include <string.h>
#define N = 5
int front=-1, rear=-1;
int queue[N]

void enqueue(int x)
{
    if (front == -1 && rear == -1)
    {
        printf("Underflow")
        front = rear = 0;
        queue[rear] = x;
        rear++;
    }
    else if ( rear = n-1)
    {
        printf("Overflow");
    }
    else{
        queue[rear] = x;
        rear++;
    }
```

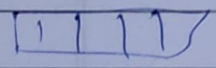
```
void dequeue{
    if (front==-1 && rear==-1)
    {
      printf("underflow");
    }
    else if (front==rear)
    {
      printf(
        front = rear = -1
      printf("Element Deleted");
    }
    else{
      front = front+1;
      printf("Element Deleted");
    }
}
void display{
    if (front==-1 && rear==-1)
    {
      printf("Queue is empty");
    }
    else{
    for(int i=front; i<=n-1; i++)
    {
      queue[i]
      printf("%d in queue[i]);
      i=i+1
    }
}
void main()
{
    char str[10], sym;
    int sym;
```

```
printf("Enter 1. for insertion, 2 for
        deletion 3. for display & exit")
printf( %d, char);
scanf(%d, &ch);
switch (ch)
{
    case 1: printf("Enter the element");
            scanf(%d, &sym);
            enqueue(sym);
    case 2: printf("Deletion");
            dequeue();
    case 3: printf("Display");
            display();
    case 4: Exit(0);
}
}
```

Output

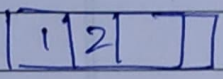
Enter 1. for insertion, 2. for deletion
3. for display & exit.

1. Enter the element
1



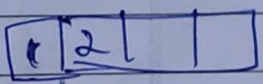
~~1. Deletion~~

1. Enter the element
2



2. Deletion
Element Deleted

3. Display



4 ~~exit~~ =>// exit

```c
#include<stdio.h>
#include<stdlib.h>
#define n 3
int front=-1, rear=-1, queue[n];

void enqueue(int a)
{
    if(rear==-1 && front== -1)
    {
     rear=front=0;
     queue[rear]=a;
    }
    else if((rear+1)%n==front)
    {
       printf("Queue is full \n");

    }
    else{
            rear=(rear+1)%n;
            queue[rear] = a;
    }

}
void dequeue()
{
    int a;
    if(front==-1 && rear == -1)
    {
        printf("underflow \n");

    }
    else if(front==rear)
    {
        front=rear=-1;
    }
    else{
        printf("%d is popped \n",queue[front]);
        front=(front+1)%n;
    }

}

void display()
{
    int i=front;
    if(front==-1)
    {
        printf("no elements \t");
    }
    else{
     while(i!=rear)
        {
            printf("%d \n",queue[i]);
             i=(i+1)%n;
        }
        break;
```

```
    }
    else if(front==rear)
    {
        front=rear=-1;
    }
    else{
        printf("%d is popped \n",queue[front]);
        front=(front+1)%n;
    }

}

void display()
{
    int i=front;
    if(front==-1)
    {
        printf("no elements \t");
    }
    else{
     while(i!=rear)
        {
            printf("%d \n",queue[i]);
             i=(i+1)%n;
        }
        break;

    }
}
void main()
{
    int a,choice;
    printf("Enter 1 enqueue, 2 dequeue ;, 3 Display 4 Exit \n");
    while(1)
    {
        printf("Enter choice \t");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: printf("Enter number to be inserted \n");
                    scanf("%d",&a);
                    enqueue(a);
                break;
                case 2: dequeue();
                        break;
                case 3: display();
                        break;
                case 4: exit(0);
        }

        }
    }
```

C:\Users\vigne\OneDrive\Des

```
1
Enter a value:20

1.insert 2.delete 3.DISPLAY 4.EXIT:
1
Enter a value:30

1.insert 2.delete 3.DISPLAY 4.EXIT:
1
Enter a value:40

1.insert 2.delete 3.DISPLAY 4.EXIT:
2
20 deleted
1.insert 2.delete 3.DISPLAY 4.EXIT:
2
30 deleted
1.insert 2.delete 3.DISPLAY 4.EXIT:
1
Enter a value:30

1.insert 2.delete 3.DISPLAY 4.EXIT:
3
40      30
1.insert 2.delete 3.DISPLAY 4.EXIT:
4

--------------------------------
Process exited after 51.07 seconds with return value 0
Press any key to continue . . .
```

SURYA Gold
Date
Page

Write a program to simulate working of a circular queue using an array. Provide the following operations: insert, delete & display. The program should print appropriate conditions.

```
#include<stdio.h>
#include<string.h>
#define N 10
int queue[N];

void enqueue(int x)
{
  if(front==-1 && rear==-1)
  {
    front=rear=0;
    queue[rear]=x;
  }
  else if((rear+1)%N)==front)
  {
    printf("queue is full");
  }
  else
  {
    rear=(rear+1)%N;
    queue[rear]=x;
  }
}
```

SURYA Gold
Date Page

```
void dequeue()
{
  if (front == -1 && rear == -1)
  {
  printf("Queue is empty");
  }
  else if (front == rear)
      {
        front = rear = -1;
      }
  else
  {
    printf("%d", queue[front]);
     front = (front + 1) % N;
  }
}
void display
{
  if (front == -1 && rear == -1)
    {
      printf("Queue is empty");
    }
  else
  { printf("Queue is: ");
    while(i != rear)
      {
        printf("%d, queue[i]);
          i = (i + 1) % N;
      }
  }
```

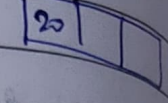
SURYA Gold
Date Page

Output

Enter 1. insert 2. delete 3. display 4. exit.

1

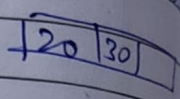
Enter an element 20



Enter 1. insert 2. delete 3. display 4. exit

1

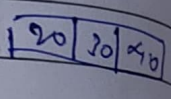
Enter an element 30



Enter 1. insert 2. delete 3. display 4. exit

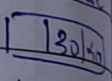
1

Enter an element 40



Enter 1. insert 2. delete 3. display 4. exit

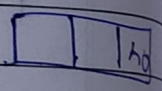
2



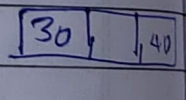
Enter 1. insert 2. delete 3. display 4. exit

2

Enter 1. insert 2. delete 3. display 4. exit 1



Enter an element 30



Enter 1. insert 2. delete 3. display 4. exit 3

40
30

08/01/24