Lab program-3a

WAP to stimulate the working of a queue of integers using an array. provide the following operations: Insert, Delete, Display, the program should print appropriate messages for queue empty and queue overflow conditions.
write the pseudocode in your observation.

pseudocode

→ start
→ Define constant N = 5, and declare array Queue[N]
→ Declare front = -1, rear = -1.

function enqueue(x)
→ If rear = N-1, then print "Queue overflow"
else if front = -1 and rear = -1 then front ← 0
rear ← 0
queue[rear] ← x

else rear ← rear + 1
queue[rear] ← x
end if and end function.

function dequeue()
→ If front = -1 and rear = -1 then print "Queue is empty".
else if front = rear then print "Deleted element", Que[front]
front ← -1
rear ← -1

else print "Deleted element", Queue[front]
front ← front + 1
end if
end function

function display() → If front = -1 and rear = -1 then print "Queue is Empty".
else print "Queue elements are:" → for I from front to rear do print Queue[i].
end for
end function.

function peek() → If front = -1 and rear = -1 then print "Queue is empty".
else print "front element:", queue[front]
endif
end function

Main program

→ Declare choice, x and Repeat the following

Repeat
print "Enter your choice":
print "1. Enqueue"
print "2. Dequeue"
print "3. Display"
print "4. peek"
print "5. Exit"
Read choice from user

→ use switch statement

switch(choice)
case 1: print "enter element to insert:" and Read x
call enqueue(x) Break
case 2: call dequeue(), break
case 3: call display(), break
Case 4: call peek() break
case 5: print exiting.. break
default: print "choice out of range"

→ end switch and until choice = 5, end.

MG

Code

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

void enqueue (int x)
   if (rear == N-1)
  { printf(" Queue Overflow\n"); }
   else if (front == -1 && rear == -1)
   { front = rear = 0;
     queue[rear] = x;
     printf ("inserted element = %d", x);
    }
   else
   {
       rear ++;
       queue[rear] = x;
       printf ("insented element = %d", x);
    }
}
void deque()
 if (front == -1 && rear == -1)
 { printf("Queue is empty\n"); }
else if (front == rear)
 { printf ("Deleted element = %d \n", queue[front]);
   front = rear = -1;
 }
 else
 { printf ("Deleted element = %d \n", queue[front]);
   front++;
 }
}
 void display()
  {  if (front == -1 && rear == -1)
     { printf ("Queue is empty\n");
     }
    else
      { printf ("Queue elements: ");
```

```
for (int i = front; i <= rear; i++)
{ printf("%d", queue[i]);
}
printf("\n");
}
}
void peek()
{ if (front == -1 && rear == -1)
{ printf("Queue is empty\n");
}
else
{ printf("Front element = %d\n", queue[front]);
}
}
int main()
{
int ch, x;
while (1)
{ printf("\n Enter your choice : 1. Enqueue 2. deque
3. Display 4. peek 5. Exit \n");
scanf("%d", &ch);
switch (ch)
{
case 1:
printf("Enter the element to insert : ");
scanf("%d", &x);
enque(x);
break;
case 2:
deque();
break;
case 3:
display();
break;
case 4:
peek()
break;
case 5: exit(0);
```

```
    default: printf("choice out of range\n");
  }
}
return 0;
}
```

olp

Enter your choice 1. Enqueue 2. Dequeue 3. Display 4. peek 5 Exit

1

Enter element to insert = 10

inserted element = 10

Enter your choice 1. Enqueue 2 Dequeue 2 Display 4. peek. 5. exit

1

Enter element to insert = 20

inserted element = 20

Enter your choice 1. Enqueue 2. Dequeue 3. Display. 4. peek 5. exit

1.

enter element to insert : 30

inserted element = 30

Enter your choice 1. Enque 2. Dequeue 3. Display 4. peek 5 exit

3

Queue elements : 10 20 30

Enter your choice 1. Enqueue 2. Dequee 3. Display 4. peek 5. exit

4

front element = 10

Enter your choice 1. Enqueue 2. Dequeue 3 Display 4. peek 5. exit

2

front element = 20

Deleted element = 10

Enter your choice 1. Enqueue 2. Dequeue 3. Display 4. peek 5. exit

2

Queue elements : 30

Deleted element = 20

Enter your choice 1. Enqueue 2. Dequeue 2. Display 4. peek 5 exit

2 Deleated element = 30

Enter your choice : 1. Enqueue 2. Dequie 3. Display 4. peek 5 exit

2

Queue is empty

Enter your choice : 1. Enqueue 2. Dequeue 3. Display 4. peek 5 exit

3

Queue is empty

Enter your choice : 1. Enqueue 2. Dequie 3. Display 4. peek 5. exit

5.

Lab program # 3b

Circular Queue

Q.

pseudocode

define size N 5
initialise queue of size N
initialise, front ← -1 and rear ← -1

function enqueue

if front & rear == -1
    front = rear = 0
    queue[rear] = x
else if (rear+1) % N == front → print "Queue is full"
else rear ← (rear+1) % N and queue[rear] ← x

function dequeue

if front and rear = -1 → print Queue is empty
else if front = rear
    print queue[front]
    and assign front = rear = -1
else
    deleted element = queue[front]
    front = (front+1) % N

function display

if front = rear = -1
    print queue is empty
else
    Queue elements are
    for int i = front; i <= rear; (i++) % N
        print queue[i]

function peek

front = rear = -1
queue is empty
else
queue(front)

main function

Variables: int x, choice

while (True)
print ("Enter your choice 1. Enqueue 2. Dequeue 3. Display 4. Peek")
Take input choice from user.
Switch(choice)
Case 1: Take input x from user
enqueue(int x); break;
Case 2: dequeue() break
Case 3: display() break
Case 4: peek() break
Case 5: print "existing" break
default: print Invalid choice

return 0

```
#include<stdio.h>
#define N 5
int queue[N];
int front=-1;
int rear=-1;
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\n");
  }
  else{
    rear=(rear+1)%N;
    queue[rear]=x;
  }
}
void dequeue()
{
  if(front==-1 && rear==-1)
  {
    printf("Queue is empty\n");
  }
  else if(front==rear)
  {
    printf("\nDeque Deleted element is : %d \n",
                                    queue[front]);
    front=rear=-1;
  }
  else{
    printf("Deleted element is : %d \n",queue[front
    front=(front+1)%N;
  }
}
void display()
{
  if(front==-1 && rear==-1)
  {
    printf("Queue is empty\n");
  }
  else{
    printf("Queue elements are:\n");
    for(int i=front; i!=rear; i=(i+1)%N)
    {
      printf("%d",queue[i]);
```

```
printf("\n"); printf("%d", queue[rear])
}
}
void peek()
  if (front == -1 && rear == -1)
    { printf("Queue is empty\n");
    }
  else
  { printf("Front elements are : %d\n", queue[front]);
  }
}
int main()
{  int choice, x;
while(1)
   { printf("\n1. Enqueue 2. Dequeue 3. Display 4. Peek\n");
     printf("Enter your choice");
     scanf("%d", &choice);
   switch(choice)
  {  case 1: printf("Enter element to insert: ");
             scanf("%d", &x);
             enqueue(x);
             break;
     case 2 : dequeue(); break;
     case 3 : display(); break;
      case 4 : peek(); break;
      case 5 : printf("Existing...\n"); break;
      default : printf("Invalid choice\n");
  } }
  return 0
}
```

MG.

o/p

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 1
Enter Element to insert : 10

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 1
Enter element to insert : 20

1. Enqueue 2. Dequeue 3. Display 4. peek.
Enter your choice 1
Enter element to insert : 30

1. Enqueue 2. Dequeue 3. Display. 4. peek
Enter your choice 3
Queue elements are :
10 20 30

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 4
front elements are : 10

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 2
Deleted element is : 10

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 1
Enter element to insert 40

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 1
Enter element to insert 50

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter element to insert 60

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 1
Enter element to insert : 70
Queue is full

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 3
Queue elements are:
20 30 40 50 60

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 5
Existing . . .

MG
3/11/25.

Lab program-3a

WAP to stimulate the working of a queue of integers using an array. provide the following operations: Insert, Delete, Display, the program should print appropriate messages for queue empty and queue overflow conditions.
write the pseudocode in your observation.

pseudocode

→ start
→ Define constant N = 5, and declare array Queue[N]
→ Declare front = -1, rear = -1.

function enqueue(x)
→ If rear = N-1, then print "Queue overflow"
else if front = -1 and rear = -1 then front ← 0
rear ← 0
queue[rear] ← x

else rear ← rear + 1
queue[rear] ← x
end if and end function.

function dequeue()
→ If front = -1 and rear = -1 then print "Queue is empty".
else if front = rear then print "Deleted element", Que[front]
front ← -1
rear ← -1

else print "Deleted element", Queue[front]
front ← front + 1
end if
end function

function display() → If front = -1 and rear = -1 then print "Queue is Empty".
else print "Queue elements are:" → for I from front to rear do print Queue[i].
end for
end function.

function peek() → If front = -1 and rear = -1 then print "Queue is empty".
else print "front element:", queue[front]
endif
end function

Main program

→ Declare choice, x and Repeat the following

Repeat
print "Enter your choice":
print "1. Enqueue"
print "2. Dequeue"
print "3. Display"
print "4. peek"
print "5. Exit"
Read choice from user

→ use switch statement

switch(choice)
case 1: print "enter element to insert:" and Read x
call enqueue(x) Break
case 2: call dequeue(), break
case 3: call display(), break
Case 4: call peek() break
case 5: print exiting.. break
default: print "choice out of range"

→ end switch and until choice = 5, end.

MG

Code

```
#include <stdio.h>
#include <stdlib.h>
#define N 5
int queue[N];
int front = -1;
int rear = -1;
void enqueue (int x)
   if (rear == N-1)
  { printf(" Queue Overflow\n"); }
   else if (front == -1 && rear == -1)
   { front = rear = 0;
     queue[rear] = x;
     printf ("inserted element = %d", x);
    }
   else
   {
       rear ++;
       queue[rear] = x;
       printf ("insented element = %d", x);
    }
}
void deque()
 if (front == -1 && rear == -1)
 { printf("Queue is empty\n");}
else if (front == rear)
 { printf ("Deleted element = %d \n", queue[front]);
   front = rear = -1;
 }
 else
 { printf ("Deleted element = %d \n", queue[front]);
   front++;
 }
}
 void display()
  {  if (front == -1 && rear == -1)
     { printf ("Queue is empty\n");
     }
    else
      { printf ("Queue elements: ");
```

```
for (int i = front; i <= rear; i++)
{ printf("%d", queue[i]);
}
printf("\n");
}
}
void peek()
{ if (front == -1 && rear == -1)
{ printf("Queue is empty\n");
}
else
{ printf("Front element = %d\n", queue[front]);
}
}
int main()
{
int ch, x;
while (1)
{ printf("\n Enter your choice : 1- Enqueue 2- deque
3- Display 4- peek 5- Exit \n");
scanf("%d", &ch);
switch (ch)
{
case 1:
printf("Enter the element to insert : ");
scanf("%d", &x);
enque(x);
break;
case 2:
deque();
break;
case 3:
display();
break;
case 4:
peek()
break;
case 5: exit(0);
```

```
    default: printf("choice out of range\n");
  }
 }
 return 0;
}
```

olp

Enter your choice 1. Enqueue 2. Dequeue 3. Display 4. peek 5 Exit

1

Enter element to insert = 10

inserted element = 10

Enter your choice 1. Enqueue 2 Dequeue 2 Display 4. peek. 5. exit

1

Enter element to insert = 20

inserted element = 20

Enter your choice 1. Enqueue 2. Dequeue 3. Display. 4. peek 5. exit

1.

enter element to insert : 30

inserted element = 30

Enter your choice 1. Enque 2. Dequeue 3. Display 4. peek 5 exit

3

Queue elements : 10 20 30

Enter your choice 1. Enqueue 2. Dequee 3. Display 4. peek 5. exit

4

front element = 10

Enter your choice 1. Enqueue 2. Dequeue 3 Display 4. peek 5. exit

2

front element = 10

Deleted element = 10

Enter your choice 1. Enqueue 2. Dequeue 3. Display 4. peek 5. exit

2

Queue elements : 20 30

Deleted element = 20

Enter your choice 1. Enqueue 2. Dequeue 2. Display 4. peek 5 exit

2 Deleted element = 30

Enter your choice : 1. Enqueue 2. Dequeue 3. Display 4. peek 5 exit

2

Queue is empty

Enter your choice : 1. Enqueue 2. Dequeue 3. Display 4. peek 5 exit

3

Queue is empty

Enter your choice : 1. Enqueue 2. Dequeue 3. Display 4. peek 5. exit

5.

Lab program # 3b

Circular Queue

Q.

pseudocode

define size N 5
initialise queue of size N
initialise, front ← -1 and rear ← -1

function enqueue

if front & rear == -1
    front = rear = 0
    queue[rear] = x
else if (rear+1) % N == front → print "Queue is full"
else rear ← (rear+1) % N and queue[rear] ← x

function dequeue

if front and rear = -1 → print Queue is empty
else if front = rear
    print queue[front]
    and assign front = rear = -1
else
    deleted element = queue[front]
    front = (front+1) % N

function display

if front = rear = -1
    print queue is empty
else
    Queue elements are
    for int i = front; i <= rear; (i++) % N
        print queue[i]

function peek

front = rear = -1
queue is empty
else
queue(front)

main function

Variables: int x, choice

while (True)
print ("Enter your choice 1. Enqueue 2. Dequeue 3. Display 4. Peek")
Take input choice from user.
switch(choice)
case 1: Take input x from user
enqueue(int x); break;
case 2: dequeue() break
case 3: display() break
case 4: peek() break
case 5: print "existing" break
default: print Invalid choice

return 0;

```
#include<stdio.h>
#define N 5
int queue[N];
int front=-1;
int rear=-1;
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\n");
  }
  else{
    rear=(rear+1)%N;
    queue[rear]=x;
  }
}
void dequeue()
{
  if(front==-1 && rear==-1)
  {
    printf("Queue is empty\n");
  }
  else if(front==rear)
  {
    printf("(Deque)Deleted element is : %d\n",
                                queue[front]);
    front=rear=-1;
  }
  else{
    printf("Deleted element is : %d\n",queue[front
    front=(front+1)%N;
  }
}
void display()
{
  if(front==-1 && rear==-1)
  {
    printf("Queue is empty\n");
  }
  else{
    printf("Queue elements are:\n");
    for(int i=front; i!=rear; i=(i+1)%N)
    {
      printf("%d",queue[i]);
```

```
printf("\n"); printf("%d", queue[rear])
}
}
void peek()
if (front == -1 && rear == -1)
{ printf("Queue is empty\n");
}
else
{ printf("Front elements are : %d\n", queue[front]);
}
}
int main()
{ int choice, x;
while(1)
{ printf("\n1. Enqueue 2. Dequeue 3. Display 4. Peek\n");
printf("Enter your choice");
scanf("%d", &choice);
switch(choice)
{ case 1: printf("Enter element to insert: ");
scanf("%d", &x);
enqueue(x);
break;
case 2 : dequeue(); break;
case 3 : display(); break;
case 4 : peek(); break;
case 5 : printf("Existing...\n"); break;
default : printf("Invalid choice\n");
} }
return 0
}
```

MG.

o/p

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 1
Enter Element to insert : 10

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 1
Enter element to insert : 20

1. Enqueue 2. Dequeue 3. Display 4. peek.
Enter your choice 1
Enter element to insert : 30

1. Enqueue 2. Dequeue 3. Display. 4. peek
Enter your choice 3
Queue elements are :
10 20 30

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 4
front elements are : 10

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 2
Deleted element is : 10

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 1
Enter element to insert 40

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 1
Enter element to insert 50

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter element to insert 60

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 1
Enter element to insert : 70
Queue is full

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 3
Queue elements are:
20 30 40 50 60

1. Enqueue 2. Dequeue 3. Display 4. peek
Enter your choice 5
Existing . . .

MG
3/11/25.