Name - Pranjali P. Sirambat
Roll No. - 71

Experiment No. 2

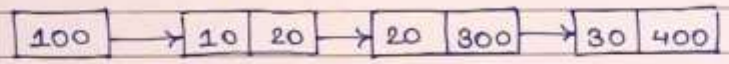
Aim: Implementation of queue using array for real world application.

Objectives:
To introduce the concept of data structures & analysis procedure.
To conceptualize linear data structures and its implementation for various real-world applications.

Theory -

Introduction to linear & non-linear data structure
Linear Data Structure organize data elements in linear fashion and each element is attached one after other.

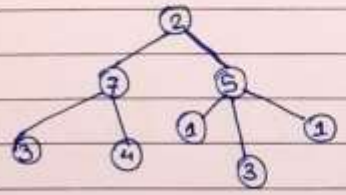
- Contiguous memory locations allocation



- Examples - Array, stack, Queue, lists

Non-Linear Data Structures - Organization is not in a sequential fashion and each element is attached one after other several data elements multiple relationships among them.

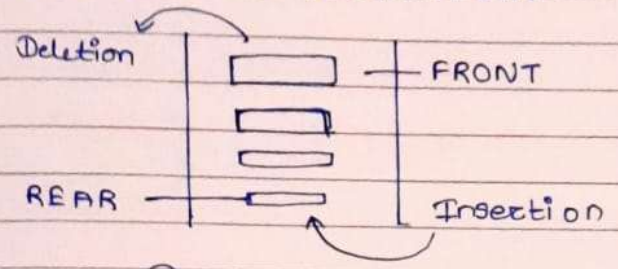
- Examples - Graphs, Tree



Introduction to Queue -

Queue is a linear structure which follows a particular order in which the operations are performed the order is First In First Out (FIFO).

In a queue, new elements one added to queue from one end called REAR end & elements are always removed from other end called FRONT end.



Operation in Queue -

Enqueue - Adds an item in queue.
Dequeue - Removes an item in queue.
Front - get the front item from queue.
rear - gets the rear item from queue.

Algorithm -

- QINSERT (Q, F, R, N, V). Given F & R, pointers to front & rear elements of queue Q having N elements V insertion in queue Q.

1. If $R \geq N$
   then write ('overflow')
   Return
2. [Increment rear pointer] $R \leftarrow R + 1$
3. [Insert element] $Q[R] \leftarrow y$

4. [Is front pointer properly set?]
   If F = 0
   then F ← 1
   Return

QDELETE (Q, F, R) Given F & R pointers to front & rear elements of queue Q, element y is to be deleted.

1. if F = 0
   then write ('Underflow')
   Return (0)
2. [Delete element] y ← Q [F]
3. [Queue empty]
   if F = R
   then F ← R ← 0
   else F ← F + 1 (increment front pointer)
4. [Return element] Return [V]

Example - Best example is the people standing in a railway reservation row for tickets. As each new person comes and stands at end of row and person after their reservation confirmation get out of row from front end.

Conclusion - Learned how to implement queue using array And Queue is used when things don't have to be processed immediately, but have to processed in first in first out.

outcome - Apply the concepts of queues to real world application

PROGRAM FOR QUEUE- QUEUE.C :-

DOSBox 0.74, Cpu speed: max 100% cycles, Fra...

File Edit Search Run Compile Debug Project Options Window Help

QUEUE.C

```c
/*********************************
IMPLEMENTATION OF QUEUE USING ARRAY
*********************************/
#include<stdio.h>
int Q[100],FRONT=-1,REAR=-1,i,n,x,choice;
void insert();
void delete();
void display();

void main()
{
  printf("\t WELCOME TO IMPLEMENTATION OF QUEUE USING ARRAY!!\n");
  printf("Enter the size of queue (Maximum size=100):");
  scanf("%d",&n);
  do
  {
   printf("\n QUEUE OPERATION AVAILABLE:\n");
   printf("\t1.Insert \t2.Delete \t3.Display \t4.Exit\n");
   scanf("%d", &choice);
   switch(choice)
   {
```

1:1

F1 Help F2 Save F3 Open Alt-F9 Compile F9 Make F10 Menu

DOSBox 0.74, Cpu speed: max 100% cycles, Fra...

File Edit Search Run Compile Debug Project Options Window Help

QUEUE.C

```c
   switch(choice)
   {
   case 1:
   insert();
   break;
  case 2:
   delete();
   break;
  case 3:
   display();
   break;
  case 4:
   printf("Exit:Progrsm Finished!!");
   break;
  default:
   printf("Please enter a valid choice 1,2,3,4\n");
   break;
   }
  }while(choice!=4);
}
//Function to INSERT element
```

40:9

F1 Help F2 Save F3 Open Alt-F9 Compile F9 Make F10 Menu

DOSBox 0.74, Cpu speed: max 100% cycles, Fra...

File Edit Search Run Compile Debug Project Options Window Help

QUEUE.C

```c
  }while(choice!=4);
}
//Function to INSERT element
void insert()
{
 if(REAR>=n-1)
 {
 printf("Queue Overflow!\n");
 }
else
{
 printf("Enter the element to insert:");
 scanf("%d",&x);
 REAR++;
 Q[REAR]=x;
 if(FRONT==-1)
 {
  FRONT=0;
 }
}
}
```

58:9

F1 Help F2 Save F3 Open Alt-F9 Compile F9 Make F10 Menu

DOSBox 0.74, Cpu speed: max 100% cycles, Fra...

File Edit Search Run Compile Debug Project Options Window Help

QUEUE.C

```c
//FUNCTION to DELETE element
void delete()
{
 if(FRONT==-1)
{
 printf("Queue Underflow!\n");
}
else
{
 printf("The deleted element is:%d\n",Q[FRONT]);
 if(FRONT==REAR)
 FRONT=REAR=-1;
 else
  FRONT++;
 }
}
//Function to DISPLAY Queue
void display()
{
 if(REAR<0)
 {
```

79:9

F1 Help F2 Save F3 Open Alt-F9 Compile F9 Make F10 Menu

```
DOSBox 0.74, Cpu speed: max 100% cycles, Fra...
≡ File Edit Search Run Compile Debug Project Options Window Help
[■]═══ QUEUE.C ═══1=[■]
 }
}
//Function to DISPLAY Queue
void display()
{
 if(REAR<0)
 {
  printf("Queue is empty!\n");
 }
 else
 {
 printf("The elements in the Queue are:\n");
 for(i=FRONT;i<n;i++)
 {
  printf("%d",Q[i]);
 }
 printf("\n");
 }
}
93:9
F1 Help F2 Save F3 Open Alt-F9 Compile F9 Make F10 Menu
```

OUTPUT

1.ENQUEUE(INSERT THE ELEMENT)

```
C:\TURBOC3\BIN>TC
         WELCOME TO IMPLEMENTATION OF QUEUE USING ARRAY!!
Enter the size of queue (Maximum size=100):5

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
1
Enter the element to insert:1

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
1
Enter the element to insert:2

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
1
Enter the element to insert:3

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
1_
```

SHOWING OVERFLOW MESSAGE :-

```
DOSBox 0.74, Cpu speed: max 100% cycles, Fra...
Enter the element to insert:3

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
1
Enter the element to insert:4

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
1
Enter the element to insert:5

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
1
Queue Overflow!

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
1
Queue Overflow!

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
```

2.DISPLAY THE ELEMENT :-

```
1
Queue Overflow!

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
3
The elements in the Queue are:
12345

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
```

3.DEQUEUE (DELETE THE ELEMENT)

```
DOSBox 0.74, Cpu speed: max 100% cycles, Fra...
 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
3
The elements in the Queue are:
12345

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
2
The deleted element is:1

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
2
The deleted element is:2

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
3
The elements in the Queue are:
345

 QUEUE OPERATION AVAILABLE:
        1.Insert        2.Delete        3.Display       4.Exit
```