linear queue implementation (pseudo code):-

```
    define   MAX 100
     int queue [MAX]
      int front = -1
      int rear = -1

void enque (int item a) {
    if (rear == MAX-1) {
     printf ( "overflow"); }
  else {
    if (front == -1)
          front = 0;
          rear= rear +1;
          queue [rear]= a;
      }.
void ~~delete~~ deque () {
    if (front == -1 // front > rear) {
        printf ( "underflow"); }

     else {
     printf (queue [front]);
          front = front +1;
       }
}
     void display() {
     for (int i = front ; i <= rear ; i++ ) {
          printf ("%d", queue [i]);
     }
}
```

program 4 : Circular queue (pseudocode

```
Initialize :.
define MAX 5
int queue [MAX]
int front = -1;
int rear = -1;

void enque (int a) {
if (front == -1) {
    front = 0;
    rear = 0; }

else if (rear == MAX-1)
    rear = 0;

 else
    rear = rear+1;
    queue (rear) = a;
    printf ("%d enqued \n", item.


void deque ( ) {
    if (front == -1) {
        printf ("underflow");
    else {
     printf ("% dequed\n", queue [insert]);
        if (front == rear) {
            front = -1;
    rear = -1 ; }
  else if  (front = = MAX-1)
           front = 0;
        else
           front = front + 1 ;

  }

void display () {
if  front == -1 ) {
        printf  ("Empty") ;
    }
        else {
        printf (" Elements : ") ;
           int i = front ;
           while (1)  {
              printf ("%d " queue [i]));
              if  i = = rear)
                  break ;
             i = (i + 1) % MAX  ;
           }

           printf  (" in") ;
           }
    }
```

Circular queue program.

```
# include <stdioh>
# include <stdlib.h>
# define Size 5
int queue [SIZE];
int front = -1 , rear = -1;

int is full () {
  return (front == (rear +1) % SIZE);
     }

int is Empty () {
        return (front = = -1);
   }
void   Enque (int item) {
      if   (is full ()) {
    printf ("Queue overflow! cannot insert %d\n", item);
        return ;
      }
    if (is Empty ()) {
        front = rear = 0;

      } else {
          rear = (rear +1) % SIZE ;
       }
     queue [ rear ] = item;
     printf ("Enqueued : %. d \n", item);

   }
void deque () {
    if (is Empty ()) {
     printf (queue underflow ! nothing to deque . \n");
         return;
  }
printf ( "Dequeued : %d \n", queue [front]);
```

```
if  (front == rear) {
  (front == rear) {
     front = rear = -1 ;
  } else {
     front = (front +1) % SIZE ;
  }

}

  void display () {
     if (isEmpty ()) {
       printf ("Queue is empty. \n");
       return ;
   }
   printf ("queue elements : ");
       int i = front ;
       while (1) {
       printf (" %d", queue [i]);
        if   (i == rear)
              break ;
        i = (i + 1)  % SIZE;
     }
   printf (" \n");
    }
int main () {
   int choice, value;
 while (1) {
  printf ("\n -- circular  queue Menu ---\n");
   printf ("1.  Enque \n");
   printf ("2. Deque \n");
   printf ("3. Display \n");
   printf ("4.  Exit \n");
   printf ( "Enter your  choice : ");
   scanf ("%d, & choice );
switch (choice) {
  case 1 :
   printf ("Enter  value to  enqueue : ");
   scanf ("%d", & value );
   enque  (value);
      break ;
case 2 :
      deque () ;
       break :
case 3 : display()
       break;
                      case 4 :
                        printf (" Exiting....\n");
                        exit(0) ;
                      default :
                        printf (" Invalid choice ! please try agai
                                                              .\n
                      }
                    }
                   } return 0;
```

output of circular queue.

```
---- circular queue menu --
 1.  Enque.
 2   Dequeue
 3.  Display
 4.  Exit
 Enter your choice : 01
     Enter item = 1
 Enter your Choice : 01
     Enter item = 2
 Enter your choice : 01
     Enter item = 3
 Enter your choice : 01
       Enter item = 4
 Enter your choice : 01
         Enter item = 5
 Enter Choice : 3
     1 2 3 4 5
Enter choice 2
   Deleted element 1
Enter Choice 2
   Deleted element 2
Enter Choice : 1
     Enter element 6
Enter Choice : 3

     3 4 5 6
```

program :- 03   Linear queue program.

```
# include <stdio.h>
# include <stdlib.h>
# define SIZE 5

int queue [SIZE];
int front = -1, rear = -1;

int is full () {
    return (front == (rear+1) % SIZE;
}
int is Empty () {
    return ( front == -1 );
}

void enqueue (int item {
  if ( is full ()) {
    printf ("Queue overflow! cannot insert %d \n", item)
    return ;
  }
  if ( is Empty ()) {
      front = rear = 0;
  } else {
    rear = (rear +1) % SIZE;
  }
 queue [rear] = item;
 printf ("Enqueued : %d \n", item).
 }

void deque () {
  if (is Empty ()) {
    printf ("Queue underflow ! nothing to deque .\n" )
    return;

}
printf ( "Queue is empty .\n");
    return;

}
printf ("queue elements : ");
    int i = front ;
while (1)

printf ("Dequed : %d \n". queue [front]);
    if (front == rear){
      front = rear = -1;
    } else {
      front = (front+1) % SIZE;
    }
}
void display () {
  if (is Empty ()) {
     printf (" Queue is empty .\n");
     return;
  }
printf ("queue elements:");
    int i = front;
  while (1) {
  printf ("%d", queue [i]);
    if (i == rear)
        break;
  i = (i+1) % SIZE;
}
```

```
    printf (" /n" );

  }
int main () {
  int choice , value :

  while (1) {
    printf ("\n--- linear queue Menu ~~circular queue Menu~~ --- \n");
    printf ("1. Enqueue\n ");
    printf ("2. Dequeue \n");
    printf ( "3. Display \n" ) ;
    printf ("4. Exit \n" );
    printf ("Enter your Choice : " );
    scanf (" %d , & choice);

    switch (choice) {
     case1 :
       printf ("Enter value to enqueue : ");
       scanf ( "%d" . & value );
       enque (value );
        break ;
     case 2 ;
        deque () ;
         break;
    case 3 ;
       display ();
         break ;
      case 4 ;
       print f  " Exiting ... \n" );
       exit (0) :
```

```
default :
    printf (" Invalid choice ! please try again .\n");
    }
  }
return 0;
}.
```

o/p of -- linear queue Menu ---

```
    1. Enque

    2  Deque.

    3. Display.

    4. Exit

Enter your choice :1
     Enter item : 5

Enter your choice : 1
     Enter item : 10

Enter your choice : 1
     Enter item : 15

Enter your choice 3
     5 10 15

Enter your choice 2
     Dequeued element : 5
Enter your choice : 3
     10 15
Enter your choice 4
     Exiting
```

Nand
04/11