```
while (sum. inch >= 12.0){
        sum. inch -= 12.0;
        sum. fet++;
    }
pf (" feet sum : %d inch sum : %0.2f", sum.fet,
      sum. inch);
} ~~ot~~
```

o/p ⇒ feet sum : 20 inch sum : 9.2

week - 4 08/01/2024

Implement linear queue using linear Array

* pseudo code

```
int q[size]
 front == -1;
 rare == -1;
enqueue ( x){
  if (isfull())
    pf ("queue is full");
   else if (front ← rear ← 0)
  else
    (rear = rear++)
    q [rear] = x
dequeue (){
 if (isempty())
    pf (" queue is empty");
else if (front == rear)
else (front = front+1)
```

```
isfull (){
   if (rear == size-1){
   ~~pf (" stack is~~
   return (-1);
   }
  }

is empty (){
 if (front == -1 || rear == -1)
    return 1;
else
   return 0;
```

```
#include <stdio.h>
# define MAX 10
 int q [MAX], front = -1 , rear = -1;
void insert (void);
int delete_element();
~~int peek();~~
void display ();

void main (){
  int option
  printf (" 1.Insert \n 2. Delete \n 3.Display ");
  printf (" Enter the choice");
  scanf (" %d ", & option);
  switch (option) {
   case 1 : insert();
        break;
   case 2 : delete_element();
        if (val != -1)
          printf (" \n the deleted number is: %d ", val);
        break;

   case 3: display();
        break;
   Default : printf (" Enter valid input");
  }
}
```

```
void insert()
{
 int num;
 pf("\n Enter the no. to be inserted");
 scanf(" %d",&num);
 if (rear == MAX-1);
        pf("overflow");
 else if (front == -1 && rear == -1)
     {  front = 0;
        rear = 0;
     }
 else
     rear++;
     q[rear] = num;
}
int delete_element()
{
  int val;
  if (front == -1 || front > rear)
    {
      pf ("\n underflow");
      return -1;
    }
else
    { val = q[front];
      front ++;
```

```
  if (front > rear)
    {
      rear = -1;
      front = -1;
    }
  return val;
  }
}
void display ()
 { int i;
   pf ("\n");
   if (front == -1 || front > rear)
       pf (" \n queue is empty");
   else {
       for(i = front ; i <= rear ; i++)
           pf(" \t %d", q[i]);
     }
  }
```

2) Pseudo code.

```
enqueue (x){
if (is full)
    pf (" queue is full");
else if (is empty()){
    front ← rear ← 0
else
  rear ← (rear+1) % N.
  A [rear ] = x }
dequeue ()){
  if (is empty())
    pf (" queue is empty");
else if ( front == rear){
    front = -1; rear = -1;}
```

```
else
    front = (front+1) % N;
isfull (){
    if (rear +1)%N == front)
        return 1 )
    else
     return 0;
}
```

[illegible] 08/01/24

2) Implement Circular queue using Array,

```
# include <stdio.h>.
# include < stdlib.h>
# define SIZE 5
 int items[SIZE] ,rear =-1 ,front =-1;
int isfull()
 { if ((front==rear+1) || (front==0 && rear ==SIZE-1))
        return 1;
    return 0;
int isempty()
{
 if (front==-1)
      return -1;
 return 0;
}
void enqueue (int element).
{ if (isfull())
    {
      printf ("\n queue is full");
   }
   else {
      if (front == -1)
         front =0;
      rear = (rear+1) % SIZE;
      items [rear] =element;
      pf ("%d is inserted", element);
  }
```

```
int dequeue ()
{ int value;
  if (isempty()){
     pf(" \nQueue is empty!!");
         return -1;

   }
   else
     {
     value = items [front];
      if (front ==rear)
      { front =-1;
       rear =-1;
      }
      else
       {
       front =(front+1) % SIZE;
       }
      return (value);
      }
   }
void display ()
{
  int i;
  if (isEmpty()).
       pf (" \n queue is empty");
  else {
      pf(" \n front position = %d \n", front);
      for (i =front; i !=rear; i= (i+1) % SIZE)
    {
      pf (" \t%d\t", items [i] );
    }
      pf(" %d \t", items [i]);
```

```
void main()
{
 int choice, element;
 while(1)
 {
 pf("\n 1. Insert \t 2. Delete \t 3. Display");
 pf(" \n Enter choice");
 sf(" %d", &choice);
 switch (choice)
 {
 case 1 : pf(" \n Enter the element");
          sf("%d", & element);
          enqueue (element);
          break;
 case 2 : element = dequeue ();
          if (element != -1)
              pf("%d element is deleted", element);
          break;
 case 3 : display ();
          break;
 default : pf("\n Invalid input");
 }
}
```

```
o/p : Enter choice 1
        enqueue (10);
        enqueue (20)
        enqueue (30)
        display ()
        dequeue ()
 10 is inserted
 20 is inserted
 30 is inserted.
 10
 20
 30
the deleted element is 10.
```

22/01/24