Assignment - 03

Name	:	B.Lakesh
Reg No	:	192365037
Dept	:	CSE (Cyber Security)
Course Code	:	CSA0389
Course Name	:	Data Structures
Faculty Name	:	Dr. Ashok Kumar
Assignment No	:	03
Date of Submission	:	21-Aug-2024
No.of pages	:	04

Illustrate the queue operation using following function calls of size=5. Enqueue (25), Enqueue (37), Enqueue (90), Dequeue (), Enqueue (15), Enqueue(40), Enqueue(12), Dequeue (), Dequeue(), Dequeue().

Sol. Let's assume the queue has a size of 5

Initialise state:

Queue : [-, -, -, -, -] (Empty)
front : -1
Rear : -1

1) Enqueue(25):

Insert 25 at the rear
Queue after operation : [25, -, -, -, -]
front: 0 [moved from -1 to 0]
Rear: 0 [moved from -1 to 0]

2) Enqueue (37):

Insert 37 at the rear

Queue after operation [25, 37, -, -, -]
Front: 0
Rear: 1

3) Enqueue (90):

Insert 37 at the rear

Queue after operation : [25, 37, 90, -, -]
Front = 0,
Rear = 2,

4) Dequeue ()

Remove element from the front
Queue after operation : [-, 37, 90, -, -]
front = 0, Rear = 2

5) Enqueue (15):

Insert is at rear

Queue after operation: [-, 37, 90, 15, -]

front = 1

rear = 3

6) Enqueue (40):

Insert 40 at rear

Queue after operation: [-, 37, 90, 15, 40]

front = 1

rear = 0

7) Dequeue ():

Remove element from the front

Queue after operation [12, -, 90, 15, 40]

front = 2

rear = 0

8) Dequeue ():

Remove the element from the front (i.e., 90)

Queue after operation [12, -, -, 15, 40]

front = 3

rear = 0

10) Deque ():

remove the element

Queue after operation [12, -, -, -, 40]

front = 4, rear = 0

11) Dequeue ():

remove the element

Queue after operation [12, -, -, -, -]

front = 0, rear = 0

Write a C program to implement Queue Operations such as EnQueue, Dequeue & display.

```
#include<stdio.h>
#define size 5;
  struct Queue {
        int item [size];
        int front, rear;
  };
void initialize (struct Queue *q) {
        q -> front = q -> rear = -1;
  }
int is-full (struct Queue *q) {
      return (q->rear+1)% size == q-> front;
}
 int isEmpty (struct Queue *q) {
            return q-> front == -1;
  }
 void enqueue (struct Queue *q) {
   if (is empty(q)){printf ("Queue Underflow"), return;
   int element = q -> items [q - front];
   if (q-> front == q->rear) q-> front = q -> rear=-1;
   else {
        q -> front = (q -> front+1) %size;
        return element; }
void display (struct * q) {
        int i = q -> front;
       while (i != q->rear) {
        printf ("%d , q -> item [i]);
```

```
    i = (i+1) % size; }

int main() {
    struct Queue q; initialize(&q);
    enqueue(q,25); enqueue(q,37);
    dequeue(q);
    return 0;
}
```