Practical No.

Date :

Aim :- Write a C Program to implement the concept of Queue ? Write a menu driven Program to do the following operations :

i> Insertion (Push)

ii> Deletion (Pop)

iii> Display

Objective :- To study the concept of linear data structure i.e (QUEUE).

Practical No.

Date :-

Aim :- Write a C Program to implement the concept of Queue ? Write a Menu driven Program to do the following operations :-

i) Insertion (Push)

ii) Deletion (Pop)

iii) Display

Objective:- To study the concept of linear data structure i.e (Queue).

Theory :- Linear Data Structures are data structures in which elements are arranged in a linear or sequential Manner.

A Queue is Important & a very useful data structure that follows the principle of First in first out (FIFO), meaning the first element added is the first one to be Removed.

There are various operations that are performed on stack i.e

i) Push (Insertion of an element)

ii) Pop (Deletion of an element)

iii) Top (Inspecting the element at Peek of the Queue without removing it.

iv) Underflow (Is_Empty conduction).

```
    front = front + 1;
  }
 }

void display ()
{
  int i;
 if(front == -1)
  {
   printf("\n Underflow");
  }
  else
  {
  printf("Queue is \n");
  for(i=front; i<=rear; i++)
   {
    printf("%d", queue_
    array[i]);
    printf("  ");
    printf("\n");
   }
  }
}
```

```
void main ()
{
 printf("Perform operations
 on Queue \n");
 printf("--------");
 printf("\t Menu:");
 printf("---------");
 printf("1. Insert element\n");
 printf("2. Delete element\n");
 printf("3. Display Queue\n");
 printf("4. Exit \n");
 printf("--------");

 int ch;
 while (1)
 {
  printf("choose Operations:\n");
  scanf("%d", &ch);
  switch (ch)
  {
   case 1:
        insert();
        break;
```

```
case 2:
        delete();
        break;

case 3:
        display();
        break;

case 4:
        exit(0);

default :
        printf("Invalid
        operation\n");
    }
}

    return 0;
}
```

Output :- Perform operations on Queue

Menu

---

1. Insert element.
2. Delete element.
3. Display queue.
4. Exit.

Choose operation : 1

Enter element : 10

10 is Inserted in the queue

Choose operation : 2

Element deleted from the Queue is : 10

Choose operation :

Conclusion :- Hence according to given theory, the code & it's corresponding output are valid.

Algorithm:- Algorithm for insertion in a Linear Queue.

Step 1. if REAR = (MAX_SIZE - 1) then
    print(" Overflow");
    return 0;
Step 2: Read NUM To be inserted in Queue.
Step 3: SET REAR = REAR + 1;
Step 4: SET QUEUE[REAR] = NUM
Step 5: if FRONT = -1 then
    SET FRONT = 0
Step 6: Exit

Algorithm for deletion in a Linear Queue.

Step 1: if FRONT = -1 then
    print("Underflow");
    return 0;
Step 2: set NUM = ~~FRONT + 1~~ QUEUE[FRONT]
Step 3: print("Deleted element is")
    NUM
Step 4: Set FRONT = FRONT + 1;
Step 5: if FRONT > REAR then
    Set FRONT = REAR - 1
Step 6: EXIT

Conclusion:- Hence according to given theory, the code & it's corresponding output are valid.