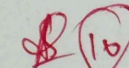
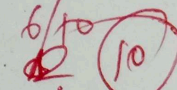


INDEX

SI No.	Date	Title	Page No	Teacher Sign
1.	29/09/25	Push, pop, peek	1.	↓ (10)
2.	6/10/25	Infix to postfix	2	 (10)
3.	13/10/25	Insert, delete, display	3 ↳	6/10  (10) 13/10

date: 13/10/25

LAB PROGRAM-3

WAP to simulate the working of queue of integers using an array. provide the following operations insert, delete, display. the program, the program should print appropriate messages for queue empty and queue overflow conditions.

Algorithm:

1. Initialise front to -1, rear to -1 and

2. Define array queue [N].

3. Insert

if $\text{rear} == \text{N} - 1$

Print "Queue overflow",

Exit from function

if $\text{front} == -1$;

Set front to 0,

Set $\text{rear} = \text{rear} + 1$

Set $\text{queue}[\text{rear}] = x$;

Print x inserted

4. Delete

if $\text{front} == -1$ & $\text{rear} == -1$

Print ("Queue is empty")

Exit from function

let $x = \text{queue}[\text{front}]$

~~$\text{front} = \text{front} + 1$~~

if ($\text{front} == \text{rear}$)

$\text{front} = \text{rear} = -1$;

Doct

else

~~front = front + 1;~~
~~if front > rear;~~

for

for

print queue[front]

front = front + 1;

4. Display:

for (i = front; i < rear + 1; i++)

print (queue[i]);

5. Inside ~~main~~ Main function,
repeat until user exits

6. Print menu insert, delete, display.

7. run ~~switch~~ for all cases .. and call the function,

Code:

#include

#include

#define

int

int

void

Code :

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define N 5
```

```
int front = -1, rear = -1;
```

```
int queue[N];
```

```
void enqueue(int x){
```

```
    if (rear == N-1){
```

```
        printf("queue overflow\n");
```

```
    }
```

```
    else if (front == -1 && rear == -1){
```

```
        front = rear = 0;
```

```
        queue[rear] = x;
```

```
    }
```

```
    else {
```

```
        rear++;
```

```
        queue[rear] = x;
```

```
    }
```

```
}
```

```
void dequeue(){
```

```
    if (front == -1 && rear == -1){
```

```
        printf("queue is empty\n");
```

```
    }
```

```
    else if (front == rear){
```

```
        front = rear = -1;
```

```
    }
```

```
    else {
```

```
        printf("deleted element = %d\n", queue[front]);
```

```
        front++;
```

```
    }
```

```
}
```



```

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

```

```

int main() {
    int ch;
    printf("1. Insert\n 2. Delete\n 3. Display\n 4. Exit\n");
    while (1) {
        int x;
        printf("Enter choice\n");
        scanf("%d", &ch);
        switch (ch) {
            case 1:
                printf("Enter the number you want to insert: ");
                scanf("%d", &x);
                enqueue(x);
                break;
            case 2:
                dequeue();
                break;
            case 3:
                display();
                break;
            case 4:
                return 0;
            default:
                printf("Invalid choice.");
        }
    }
}

```


}
return 0;

}

O/I:

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

Enter choice : 2

Queue ~~is~~ ^{is} Empty.

Enter choice : 1

Enter the number you want to Insert: 23

Element Inserted.

Enter choice : 1

Enter the number you want to Insert: 56

~~Element Inserted~~

Enter choice : 1

Enter the number you want to Insert: 89

Element Inserted

Enter choice : 1

Enter the number you want to Insert: 74

Element Inserted

Enter choice : 1

Enter the number you want to Insert: 12

Element Inserted

Enter choice : 1

Enter the number: 58

Queue Overflow.

Enter choice : 3

8 , 23
56
89
74
12

Enter choice: 2

Element deleted : 23

Enter choice : 3

8
56
89
74
12

~~Enter choice: 4.~~

~~13/10~~