dsa\queue.cpp

```cpp
#include<iostream>
#include<queue>
using namespace std;

struct node
{
    int data ;
    node* next ;
};
// insertion is at tail and deletion is at head
struct node* head = NULL ;
struct node* tail = NULL ;

void enqueue(int x)
{
        struct node* newnode= (struct node*)malloc(sizeof(struct node)) ;
        newnode->data = x ;
        newnode->next = NULL ;
    if(head == NULL && tail == NULL)
    {
        head = tail= newnode  ;
    }
    else{
         tail->next = newnode ;
         tail = newnode ;
    }
}

void display() // based on FIFO principle
{
    if(head == NULL && tail == NULL)
    {
        cout<<"queue is empty"<<endl;
        return ;
    }
    struct node* temp = head ;
    while(temp != NULL)
    {
        cout<<temp->data<<" ";
        temp = temp->next ;
    }
    cout<<endl ;
}

void peek()
{
    cout<<"front/peek element is= "<<head->data<<endl ;
}

void dequeue()
{
```

```
52      struct node* temp = head ;
53      cout<<"deleted element is = "<<head->data<<endl;
54      head = head->next ;
55      free(temp) ;
56  }
57
58  int main()
59  {
60      enqueue(10) ; // 10
61      enqueue(-3) ; // 10 -3
62      enqueue(4) ; // 10 -3 4
63      display() ;
64      dequeue() ; // 10
65      peek() ; // -3
66      display() ; // -3 4
67
68      return 0;
69  }
```