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

// Structure of Node.
struct Node
{
int data;

Node *link;
};

Node *front = NULL;
Node *rear = NULL;

//Function to check if queue is empty or
not
bool isempty()
{
 if(front == NULL && rear == NULL)
 return true;
 else
 return false;
}

//function to enter elements in queue
void enqueue ( int value )
{
 Node *ptr = new Node();
 ptr->data= value;
 ptr->link = NULL;

 //If inserting the first element/node
 if( front == NULL )
 {
  front = ptr;
  rear = ptr;
 }
 else
 {
  rear ->link = ptr;
  rear = ptr;
 }
}

//function to delete/remove element from
queue
void dequeue ( )
{
 if( isempty() )
 cout<<"Queue is empty\n";
 else
 //only one element/node in queue.
 if( front == rear)
 {
  free(front);
  front = rear = NULL;
 }
 else
 {
  Node *ptr = front;
  front = front->link;
  free(ptr);
 }
}

//function to show the element at front
void showfront( )
{
 if( isempty())
 cout<<"Queue is empty\n";
 else
 cout<<"element at front is:"<<front-
>data;
}

//function to display queue
void displayQueue()
{
 if (isempty())
  cout<<"Queue is empty\n";
 else
 {
  Node *ptr = front;
  while( ptr !=NULL)
  {
   cout<<ptr->data<<" ";
   ptr= ptr->link;
  }
 }
}

//Main Function
int main()
{
 int choice, flag=1, value;
 while( flag == 1)
 {
  cout<<"\n1.enqueue 2.dequeue 3.showfront
4.displayQueue 5.exit\n";
  cin>>choice;
  switch (choice)
  {
  case 1: cout<<"Enter Value:\n";
          cin>>value;
          enqueue(value);
          break;
  case 2: dequeue();
          break;
  case 3: showfront();
          break;
  case 4: displayQueue();
          break;
  case 5: flag = 0;
          break;
  }
 }

 return 0;
}
```