```
#include <iostream>
#include <fstream>
#include <string>
#include <cstring>
using namespace std;
template <class t1>
struct node
{
    t1 info;
    node* next;
};

template <class t1>
class queues
{
    node<t1>* front;
    node<t1>* rear;
public:
    queues();
    void enqueue(t1);
    t1 delqueue();
    bool isfull();
    bool isempty();
    void print(ofstream&);
};

template<class t1>
queues<t1>::queues()
{
    front = NULL;
    rear = NULL;
}

template<class t1>
void queues<t1>::enqueue(t1 a)
{
    node<t1>* temp;
    temp = new node<t1>;
    temp->info = a;
    temp->next = NULL;
    if (rear == NULL)
        front = temp;
    else
        rear->next = temp;

        rear = temp;
}

template<class t1>
```

```
t1 queues<t1>::delqueue()
{
    t1 value;
    node<t1>* temp;
    value = front->info;
    temp = front;
    front = front->next;
    temp->next = NULL;
    delete temp;
    return value;
}

template<class t1>
bool queues<t1>::isfull()
{
    node<t1>* ptr;
    ptr = new node<t1>;
    if (ptr == NULL)
    {
        return true;
    }
    else
    {
        delete ptr;
        return false;
    }
}

template<class t1>
bool queues<t1>::isempty()
{
    if (front == NULL)
        return true;
    else
        return false;
}

template<class t1>
void queues<t1>::print(ofstream& fout)
{
    node<t1>* temp;
    temp = front;
    while (temp != NULL)
    {
        fout << temp->info;
        temp = temp->next;
    }
}

```