Universidad de Concepción
Facultad de Ingeniería
Departamento de Ing. Informática y Cs
de la Computación

Profesor
Diego Seco

# Laboratorio 4:

Estructuras de Datos (503220)

**Estudiante:** Nicolás Araya (2018448613)

1.

ADTQueue.h
```h
#ifndef ADTQueue_H
#define ADTQueue_H

class ADTQueue{
    public:
        virtual void push(int x) = 0;
        virtual void pop() = 0;
        virtual int front() = 0;
        virtual int size() = 0;
        virtual bool empty() = 0;
};
#endif
```

2.

StacksQueue.h
```h
#include "ADTQueue.h"
#include "Iterador.h"
#include <vector>

using namespace std;

class StacksQueue : public ADTQueue {
    private:
        vector<int> st1, st2;
    public:
        StacksQueue();
        ~StacksQueue();
        void push(int x);
        void pop();
        int front();
        int size();
        bool empty();
        Iterador begin();

};
```

StacksQueue.cpp

StacksQueue.cpp > StacksQueue::pop()

```cpp
#include "StacksQueue.h"
#include <iostream>

using namespace std;

//push_back pop_back back size empty

StacksQueue::StacksQueue(){
}

StacksQueue::~StacksQueue(){
    st1.clear();
    st2.clear();
}
void StacksQueue::push(int x){
    if(st2.empty()){
        st1.push_back(x);
    }
    else{
        while(!st2.empty()){
            st1.push_back(st2.back());
            st2.pop_back();
        }
        st1.push_back(x);
    }
}
void StacksQueue::pop(){
    if(st2.empty()){
        while(!st1.empty()){
            st2.push_back(st1.back());
            st1.pop_back();
        }
        if(!st2.empty()) st2.pop_back();
    }
    else{
        st2.pop_back();
    }
}
int StacksQueue::front(){
    if(st2.empty()){
        while(!st1.empty()){
            st2.push_back(st1.back());
            st1.pop_back();
        }
        if(!st2.empty()) return st2.back();
    }
    else{
        return st2.back();
    }
}
```

```cpp
int StacksQueue::size(){
    return st1.size()+st2.size();
}
bool StacksQueue::empty(){
    if(size()==0) return true;
    else return false;
}

Iterador StacksQueue::begin(){
    if(st2.empty()){
        while(!st1.empty()){
            st2.push_back(st1.back());
            st1.pop_back();
        }
    }
    vector<int>* stack = &st2;
    Iterador i(stack);
    return i;
}
```

3.

Iterador.h

Iterador.h > Iterador

```cpp
#include <vector>
#include <iostream>

using namespace std;

class Iterador{
    private:
        vector<int>::iterator it;
        vector<int> st;
    public:
        Iterador(vector<int> *stck);
        ~Iterador();
        bool hasNext();
        int next();
};
```

Iterador.cpp

Iterador.cpp > Iterador::hasNext()

```cpp
#include "Iterador.h"


Iterador::Iterador(vector<int>* stck){
    st=*stck;
    it = st.begin();
}
Iterador::~Iterador(){
    st.clear();
}

bool Iterador::hasNext(){
    if( it == st.end()){
        return false;
    }
    else return true;
}

int Iterador::next(){
    int x = *it;
    it++;
    return x;
}
```