Eric Blasko

CSE 330 Data Structures

Winter 2018

Lab 6 – Stacks and Queues

* **Status**

100% complete

* **Complexity Analaysis**

**Stack**

If using List as container, all functions are O(1) as they are constant in time

* bool empty() const
* unsigned int size() const
* void push(const T & x)
* void push(T && x)
* void pop()
* T & top()

If using vector as container, the following are O(1) as they are constant

* bool empty() const
* unsigned int size() const
* void pop()
* T & top()

The following functions for vector containers are O(n), where n is the size of the vector

* void push(const T & x)
* void push(T && x)

**Queue**

The following functions are O(1) as they are constant in time. Vectors are not used as they do not have a pop\_front() function

* bool empty()
* unsigned int size()
* T & front()
* T & back()
* void push(const T & x)
* void push(T && x)
* void pop()
* **Source Code**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Eric Blasko

\* Stack.h

\* 02/21/2018

\* This class contains functions silmilar to that of the STL library. The class can use

\* containers of type vector and list, but list is the default container.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef STACK\_H

#define STACK\_H

#include "../lab5/List.h"

using namespace std;

template <class T, template <class S> class Container = List>

class Stack

{

public:

bool empty() const {return container.empty();}; //return if empty

unsigned int size() const {return container.size();}; //return size

void push(const T & x) {container.push\_back(x);}; //push to top lvalue

void push(T && x){container.push\_back(move(x));}; //push to top rvalue

void pop() {container.pop\_back();}; //pop off top

T & top() {return container.back();}; //return top value

private:

Container<T> container;

};

#endif

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Eric Blasko

\* Stack\_test1.cpp

\* 02/21/2018

\* This program test the push and pop functions of Stack.h. values printed to

\* console are values on top of stack

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include "../lab4/Vector.h"

#include "Stack.h"

using namespace std;

//Main function which pushs and pops value to stack

int main()

{

Stack<int> s;

s.push(5);

s.push(6);

cout << s.top() << endl;

Stack<double, Vector> v;

v.push(1.5);

v.push(2.3);

v.pop();

cout << v.top() << endl;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Eric Blasko

\* Stack\_test2.cpp

\* 02/21/2018

\* This program test the functions of the class Stack.h. assert will verifiy that each

\* function is working as intended

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <cassert>

#include "Stack.h"

#include "../lab3/String.h"

#include "../lab4/Vector.h"

#include "../lab5/List.h"

using namespace std;

//Main function that test if Stack.h is working properly

int main()

{

Stack<int, Vector> s1;

assert(s1.size() == 0);

assert(s1.empty());

s1.push(16);

assert(s1.size() == 1);

assert(s1.top() == 16);

s1.pop();

assert(s1.size() == 0);

s1.push(11);

assert(s1.size() == 1);

assert(s1.top() == 11);

s1.push(22);

assert(s1.size() == 2);

assert(s1.top() == 22);

s1.push(33);

assert(s1.size() == 3);

assert(s1.top() == 33);

s1.pop();

assert(s1.size() == 2);

assert(s1.top() == 22);

Stack<String, List> s2;

s2.push("abc");

s2.push("de");

s2.pop();

assert(s2.top() == "abc");

cout << "SUCCESS\n";

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Eric Blasko

\* Queue.h

\* 2/21/2018

\* This class contains functions similar to the STL library. As vectors are not efficient

\* with pop\_front(), list is the default container for this class. Data in this class is

\* follows First in First out

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef QUEUE\_H

#define QUEUE\_H

#include "../lab5/List.h"

using namespace std;

template <class T>

class Queue

{

public:

bool empty() { return container.empty(); }; //check if empty

unsigned int size() { return container.size(); }; //return size

T & front() { return container.front(); }; //return front value

T & back() { return container.back(); }; //return back value

void push(const T & x) { container.push\_back(x); }; //push to back lvalue

void push(T && x) { container.push\_back(move(x)); }; //push to back rvalue

void pop() {container.pop\_front(); }; //pop front value

private:

List<T> container;

};

#endif

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Eric Blasko

\* Queue\_test1.cpp

\* 02/21/2018

\* This program test the functions in Queue.h. assert will verify that each function is

\* acting as intended

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <cassert>

#include "../lab3/String.h"

#include "Queue.h"

using namespace std;

//Main function that test functions of Queue.h

int main()

{

Queue<int> s;

s.push(5);

s.push(6);

cout << s.back() << endl;

cout << s.front() << endl;

Queue<double> v;

v.push(1.5);

v.push(2.3);

v.pop();

cout << v.back() << endl;

cout << v.front() << endl;

Queue<int> s1;

assert(s1.size() == 0);

assert(s1.empty());

s1.push(16);

assert(s1.size() == 1);

assert(s1.front() == 16);

assert(s1.back() == 16);

s1.pop();

assert(s1.size() == 0);

s1.push(11);

assert(s1.size() == 1);

assert(s1.front() == 11);

assert(s1.back() == 11);

s1.push(22);

assert(s1.size() == 2);

assert(s1.front() == 11);

assert(s1.back() == 22);

s1.push(33);

assert(s1.size() == 3);

assert(s1.front() == 11);

assert(s1.back() == 33);

s1.pop();

assert(s1.size() == 2);

assert(s1.front() == 22);

assert(s1.back() == 33);

Queue<string> s2;

s2.push("abc");

s2.push("de");

assert(s2.front() == "abc");

assert(s2.back() == "de");

s2.pop();

assert(s2.back() == "de");

cout << "SUCCESS\n";

}

* **Sample Runs**

**Stack test 1**

Script started on 2018-02-18 19:05:51-0800

]0;005670557@csusb.edu@csevnc:~/cse330/lab6[005670557@csusb.edu@csevnc lab6]$ g\_[K++ 0c[K[K-c Stack\_tst1.[K[K[K[Kest1.cpp

]0;005670557@csusb.edu@csevnc:~/cse330/lab6[005670557@csusb.edu@csevnc lab6]$ g++ Stack)[K\_test1.o

]0;005670557@csusb.edu@csevnc:~/cse330/lab6[005670557@csusb.edu@csevnc lab6]$ ./a.out

6

1.5

]0;005670557@csusb.edu@csevnc:~/cse330/lab6[005670557@csusb.edu@csevnc lab6]$ exit

Script done on 2018-02-18 19:06:28-0800

**Stack test 2**

Script started on 2018-02-18 19:07:03-0800

]0;005670557@csusb.edu@csevnc:~/cse330/lab6[005670557@csusb.edu@csevnc lab6]$ g++ 0c [K[K[K-c Stack\_test2.cpp

]0;005670557@csusb.edu@csevnc:~/cse330/lab6[005670557@csusb.edu@csevnc lab6]$ g++ Stack\_test2.o ../lab3/String.o

]0;005670557@csusb.edu@csevnc:~/cse330/lab6[005670557@csusb.edu@csevnc lab6]$ ./a.out

SUCCESS

]0;005670557@csusb.edu@csevnc:~/cse330/lab6[005670557@csusb.edu@csevnc lab6]$ exit

Script done on 2018-02-18 19:07:36-0800

**Queue test**

Script started on 2018-02-18 19:04:53-0800

]0;005670557@csusb.edu@csevnc:~/cse330/lab6[005670557@csusb.edu@csevnc lab6]$ g++ -c Queue\_test1.cpp

]0;005670557@csusb.edu@csevnc:~/cse330/lab6[005670557@csusb.edu@csevnc lab6]$ g\_\_[K[K++ Queue([K)t[K[K\_test1 [K.o ../lab3/String.o

]0;005670557@csusb.edu@csevnc:~/cse330/lab6[005670557@csusb.edu@csevnc lab6]$ ./a.out

6

5

2.3

2.3

SUCCESS

]0;005670557@csusb.edu@csevnc:~/cse330/lab6[005670557@csusb.edu@csevnc lab6]$ ^C

]0;005670557@csusb.edu@csevnc:~/cse330/lab6[005670557@csusb.edu@csevnc lab6]$ exit

Script done on 2018-02-18 19:05:36-0800