Joshua Kegley

10/01/2014

Project 1

PROJECT 1

NAMESPACE COP3530

SSLL – Simple Singly Linked List

*Template Class*

SSLL is a simple data structure that implements a Linked List using Nodes. This class contains many useful methods.

Constructors/Destructors

SSLL() – *Takes no argument, starts the list off at 0*

SSLL(const SSLL& src) – *Copy Constructor*

~SSLL() – *Destructor*

Overloaded Operators

operator=(const SSLL& src) – takes an SSLL and copies it to this, returns a

reference to an SSLL

operator[](int position) – Returns a reference to a position in the structure

const operator[](int position) – Returns a const reference to a position in the structure

Public Methods

clear() - Clears the structure of all elements and then creates a new head node.

contains(const T& element, bool equals(const T& a, const T& b)) const – takes in a way to

compare two elements, and then checks for the element.

insert(const T& element, int position) – insert takes an element and places it into the position.

If less than 0 or greater than amount, throws exception.

is\_empty() const – checks to see if the structure is empty by checking size()

item\_at(int position) const – returns the item at the specified position. If no element exist,

throws exception.

pop\_back() – uses remove(amount) to remove the last element. If no element exist, throws

exception.

pop\_front() – uses remove(0) to remove the first element

print(std::ostream& out) const – accepts an std::cout to out put the structure

push\_back(const T& element) – uses insert(amount) to add element to the end. If no element

exist, throws exception.

push\_front(const T& element) – uses insert(0) to add element to the front. If no element exist,

throws exception.

remove(int position) – removes the element at the specified position and then returns it. If no

element exist, throws exception.

replace(const T& element, int position) – replaces the element at the position, returns the

original element

size() const – returns the size of structure, type size\_t

Private Methods

Node\* get\_new\_Node() – Tries to allocate a new node, once it does it returns it.

PSLL – Simple Singly Linked List

*Template Class*

PSLL is a simple data structure that implements a Linked List using Nodes and a Pool. This class contains many useful methods.

Constructors/Destructors

PSLL() – *Takes no argument, starts the list off at 0*

PSLL(const SSLL& src) – *Copy Constructor*

~PSLL() – *Destructor*

Overloaded Operators

operator=(const SSLL& src) – takes an SSLL and copies it to this, returns a

reference to an SSLL

operator[](int position) – Returns a reference to a position in the structure

const operator[](int position) – Returns a const reference to a position in the structure

Public Methods

clear() - Clears the structure of all elements and then creates a new head node.

contains(const T& element, bool equals(const T& a, const T& b)) const – takes in a way to

compare two elements, and then checks for the element.

insert(const T& element, int position) – insert takes an element and places it into the position.

If less than 0 or greater than amount, throws exception.

is\_empty() const – checks to see if the structure is empty by checking size()

item\_at(int position) const – returns the item at the specified position. If no element exist,

throws exception.

pop\_back() – uses remove(amount) to remove the last element. If no element exist, throws

exception.

pop\_front() – uses remove(0) to remove the first element

print(std::ostream& out) const – accepts an std::cout to out put the structure

push\_back(const T& element) – uses insert(amount) to add element to the end. If no element

exist, throws exception.

push\_front(const T& element) – uses insert(0) to add element to the front. If no element exist,

throws exception.

remove(int position) – removes the element at the specified position and then returns it. If no

element exist, throws exception.

replace(const T& element, int position) – replaces the element at the position, returns the

original element

size() const – returns the size of structure, type size\_t

Private Methods

Node\* get\_new\_Node() – Tries to allocate a new node, once it does it returns it.