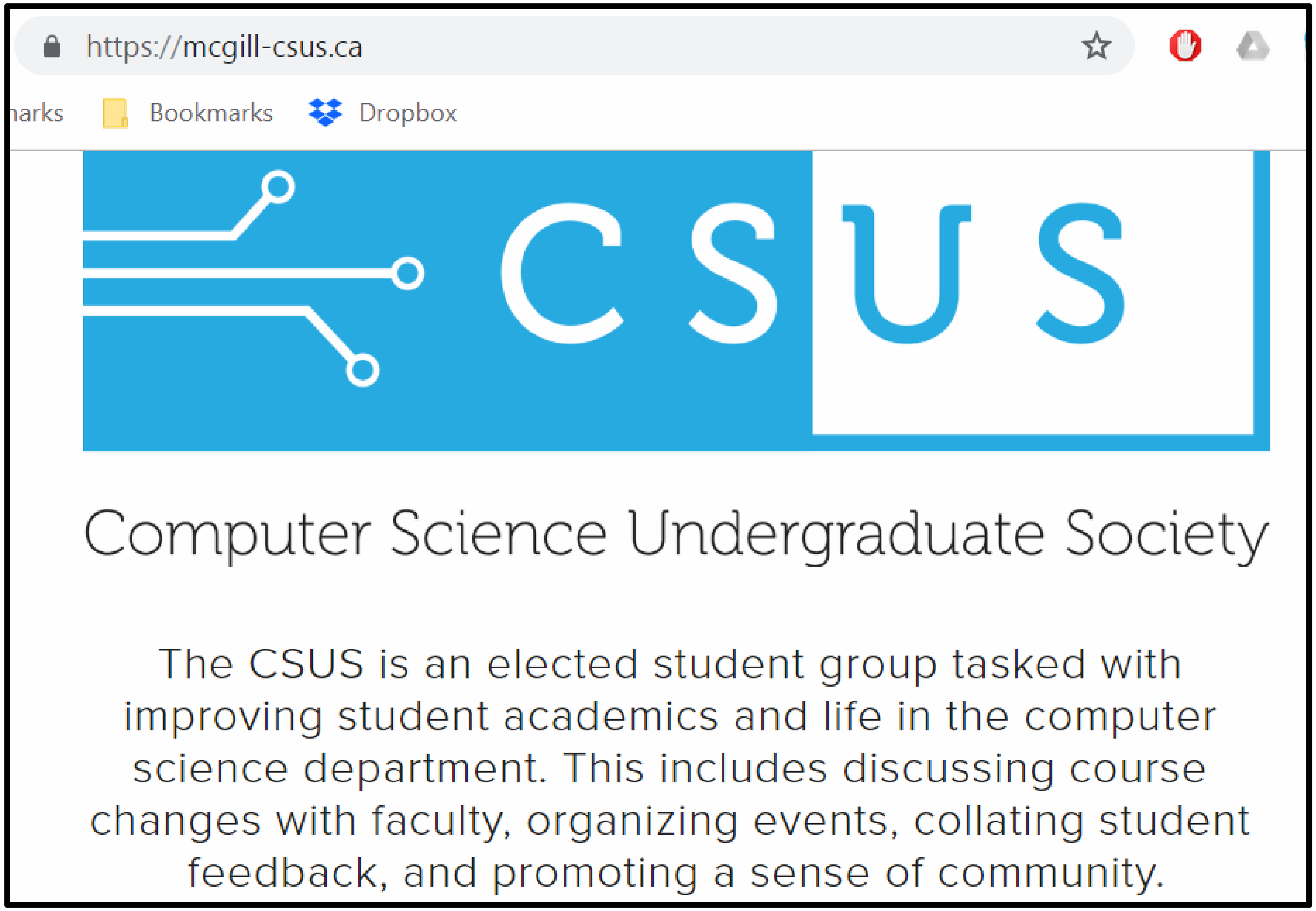
COMP 250

Lecture 13

stack

Oct. 10, 2018



Computer Science Undergraduate Society

The CSUS is an elected student group tasked with improving student academics and life in the computer science department. This includes discussing course changes with faculty, organizing events, collating student feedback, and promoting a sense of community.

“Do you have ideas to help us improve your experience in CS? If yes, then please send us your feedback (fill out survey below). It will be relayed to the Director of the School of Computer Science & to the Dean of Science. It takes 2 minutes.“

Survey https://goo.gl/forms/dUinZb986FeA1C4W2

Recall: List operations

```
get(i)         //  Returns the i-th element (but doesn't remove it)
set(i,e)       //  Replaces the i-th element with e
add(i,e)       //  Inserts element e into the i-th position
remove(i)      //  Removes the i-th element from list
remove(e)      //  Removes first occurrence of element e
               //   from the list (if it is there)
clear()        //  Empties the list.
isEmpty()      //  Returns true if empty, false if not empty.
size()         //  Returns number of elements in the list
:
```

This operations can be defined abstractly, without specifying the implementation details of the data structure (arraylist vs. linked list).

Abstract data type (ADT)

“ADT” defines a data type by the *values of the data* and *operations on the data.*

It is defined from the point of view of the *user*.

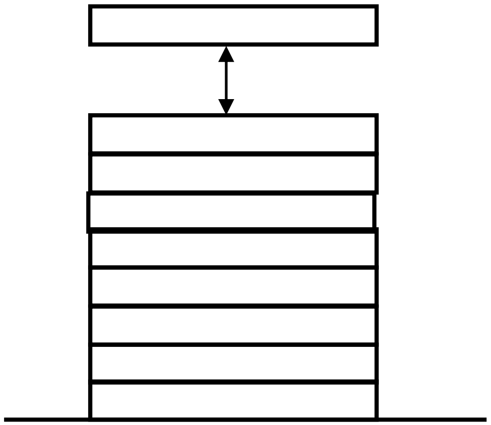
It ignores the details of the implementation.

An ADT is more abstract than a data structure.

Stack ADT

push(element)

pop( )



isEmpty( )

peek( )

A stack is a list. However, it typically does not have operations to access the list element *i* directly. Instead one accesses only one end of the list.

How to implement a stack?

	push(e)	pop ()
array list		
singly linked list		
doubly linked list		

How to implement a stack?

	push(e)	pop ()
array list*	addLast(e)	removeLast()

*Java ArrayList class doesn’t have addLast and removeLast methods.

How to implement a stack?

	push(e)	pop ()
array list*	addLast(e)	removeLast()
singly linked list	addFirst(e)	removeFirst ()

*Java ArrayList class doesn’t have addLast and removeLast methods.

How to implement a stack?

	push(e)	pop ()
array list*	addLast(e)	removeLast()
singly linked list	addFirst(e)	removeFirst ()
doubly linked list	either row above	

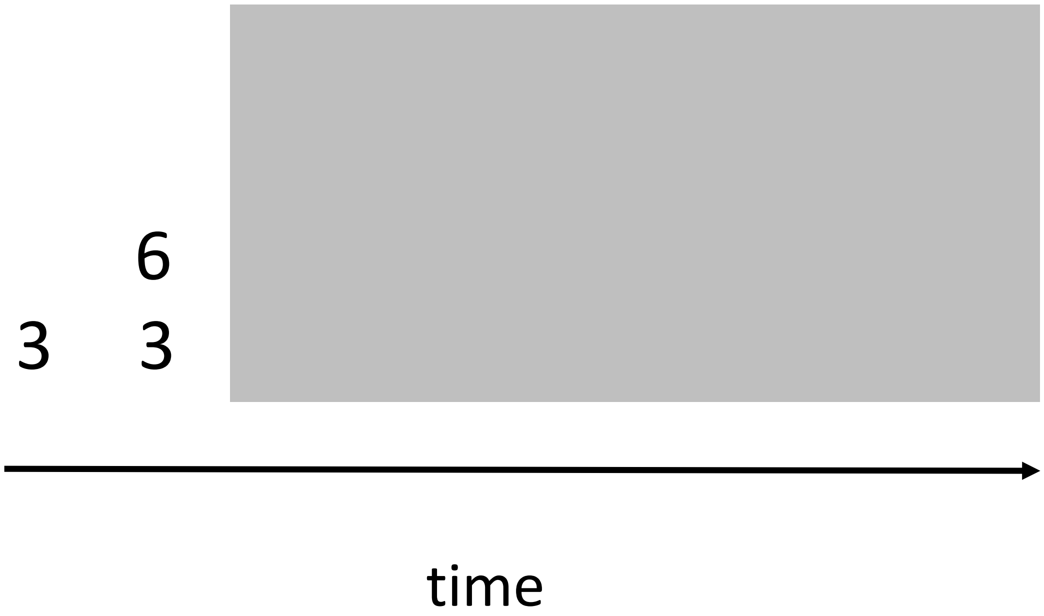
*Java ArrayList class doesn’t have addLast and removeLast methods.

Example 1: stack of int

push(3)
push(6)
push(4)
push(1)
pop()
push(5)
pop()
pop()

Example 1: stack of int

push(3)
push(6)
push(4)
push(1)
pop()
push(5)
pop()
pop()



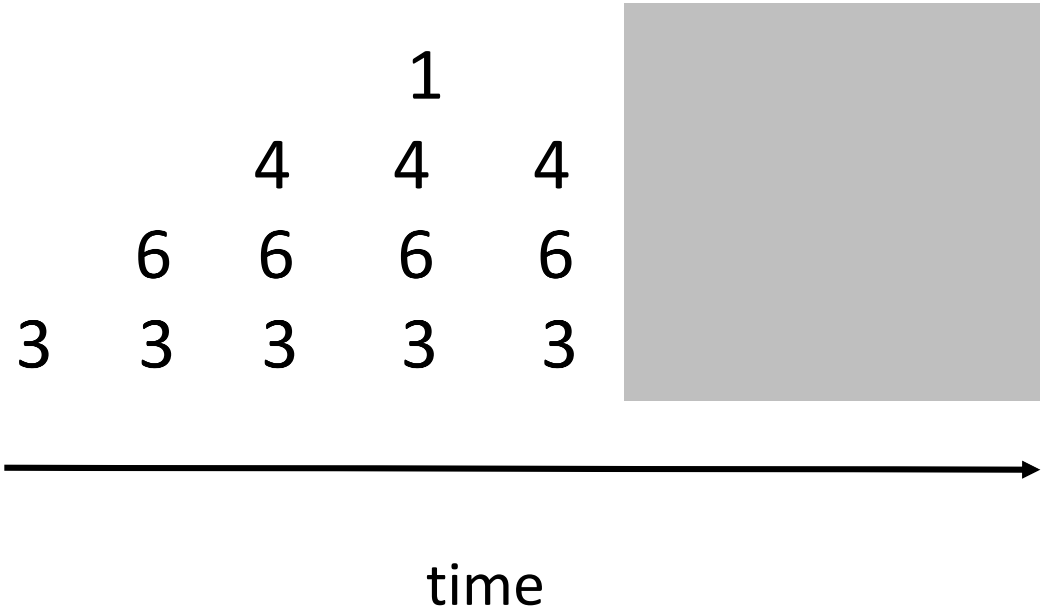
6

3 3

time

Example 1: stack of int

push(3)
push(6)
push(4)
push(1)
pop()
push(5)
pop()
pop()

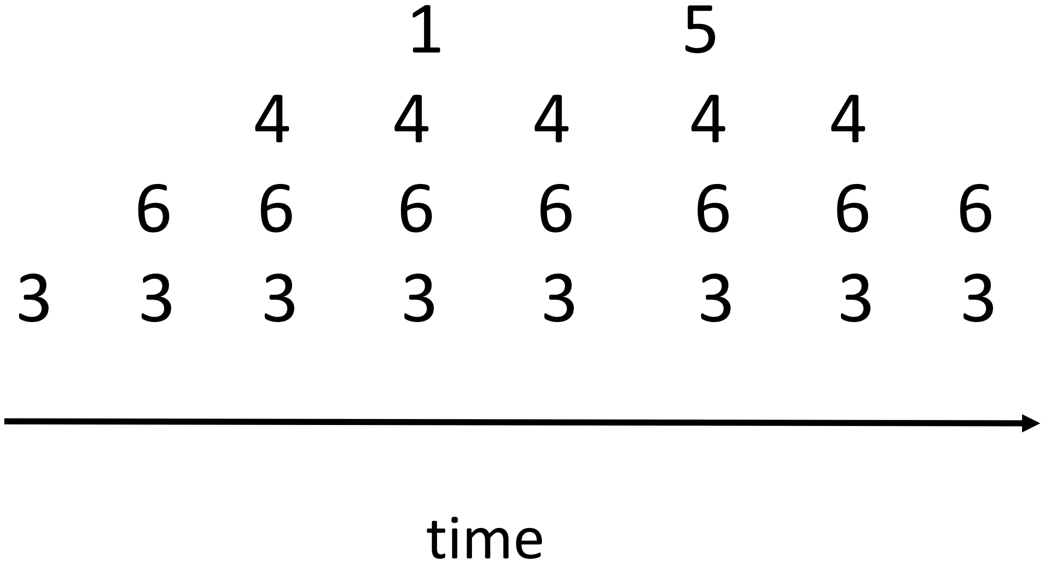


1
4 4 4
6 6 6 6
3 3 3 3 3

time

Example 1: stack of int

push(3)
push(6)
push(4)
push(1)
pop()
push(5)
pop()
pop()



Example 2 - balancing parentheses

e.g. (([ ])) [ ] { [ ] }

To ensure proper nesting, we traverse the list and use a stack.

How?

Example 2 - balancing parentheses

e.g. (([ ])) [ ] { [ ] }

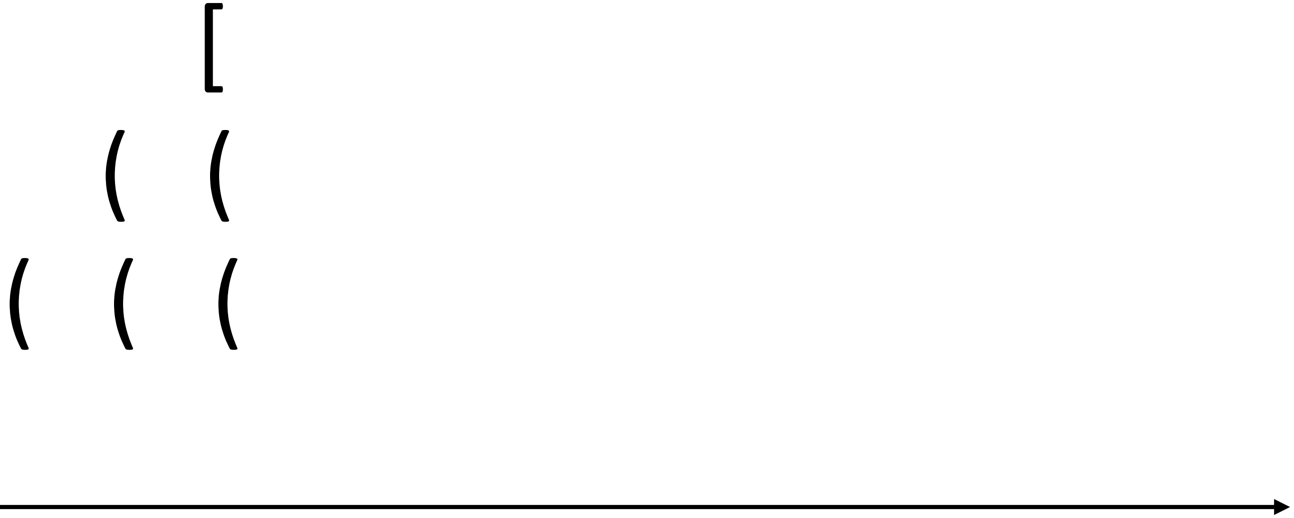
To ensure proper nesting, we traverse the list and use a stack.

When we reach a left parenthesis, we push it onto the stack.

When we reach a right parenthesis, we compare it to top of the stack. It it matches, then we pop.

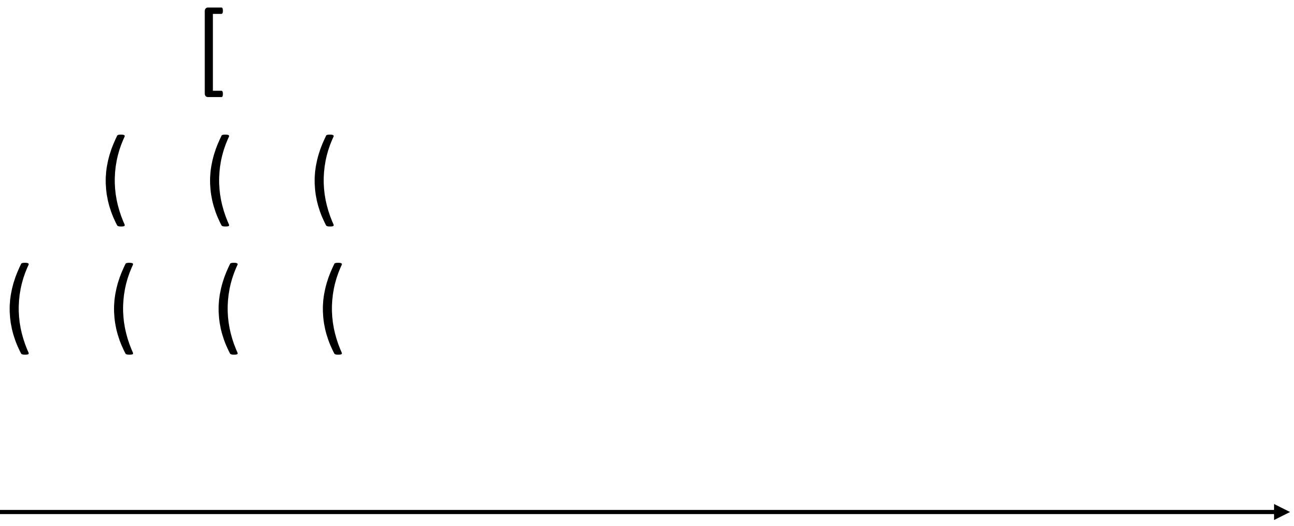
Example 2 - balancing parentheses

e.g. (([ ])) [ ] { [ ] }



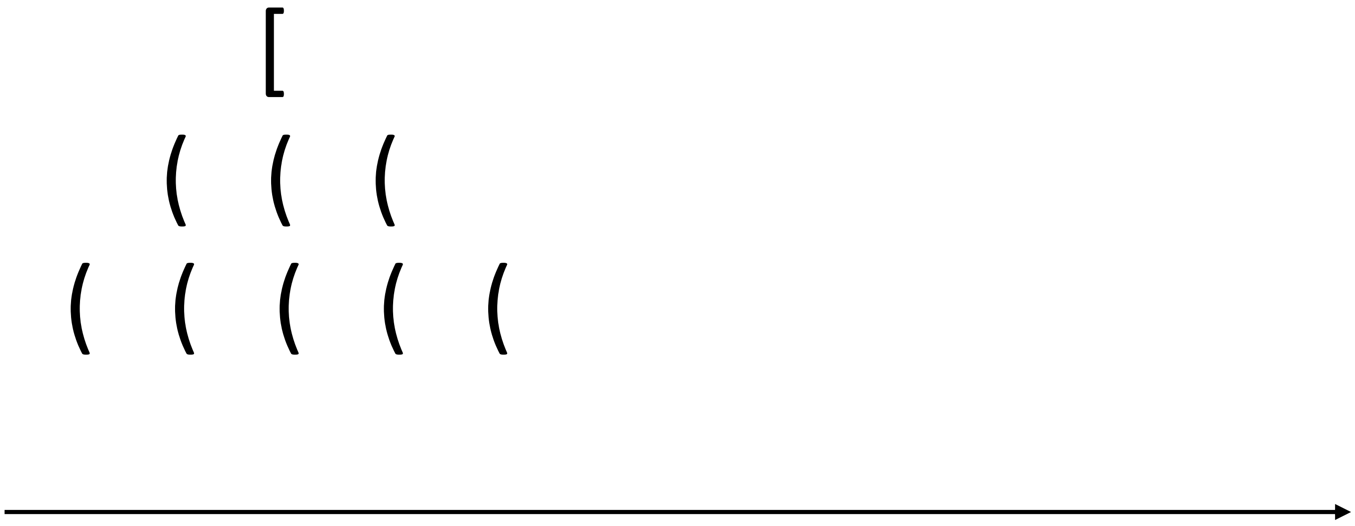
Example 2 - balancing parentheses

e.g. (([ ])) [ ] { [ ] }



Example 2 - balancing parentheses

e.g. (([ ])) [ ] { [ ] }



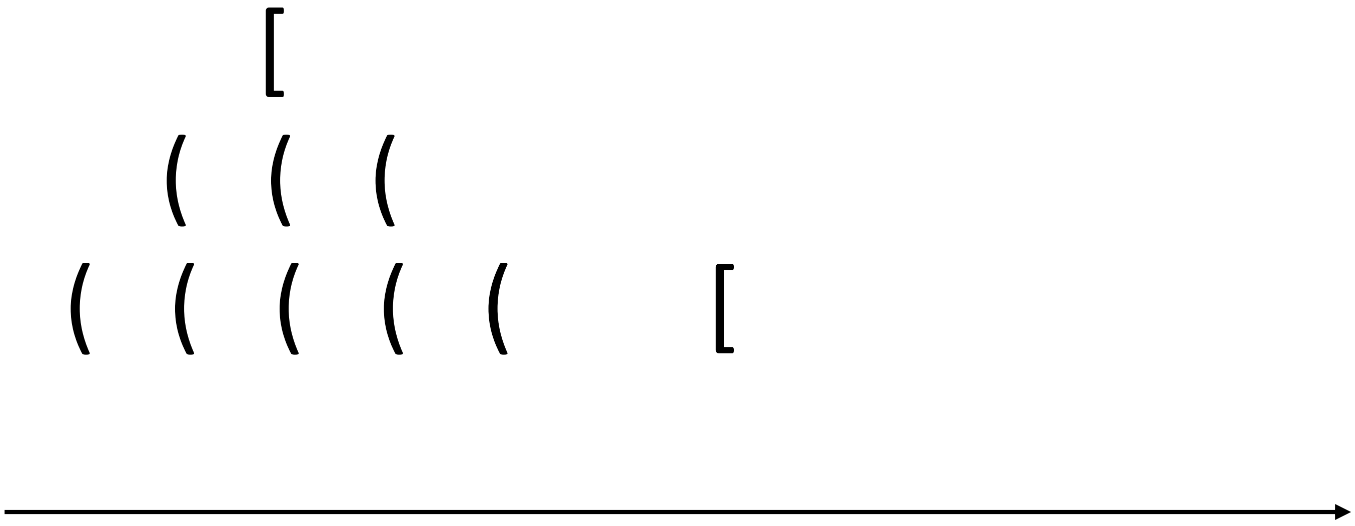
[

(((

(((((

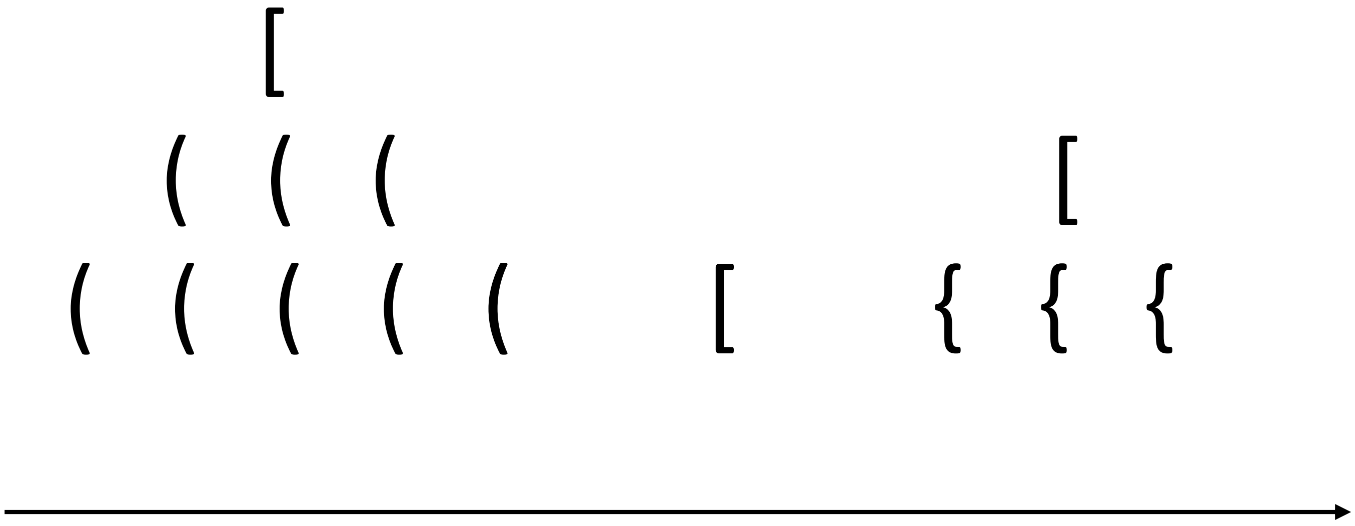
Example 2 - balancing parentheses

e.g. (([ ])) [ ] { [ ] }



Example 2 - balancing parentheses

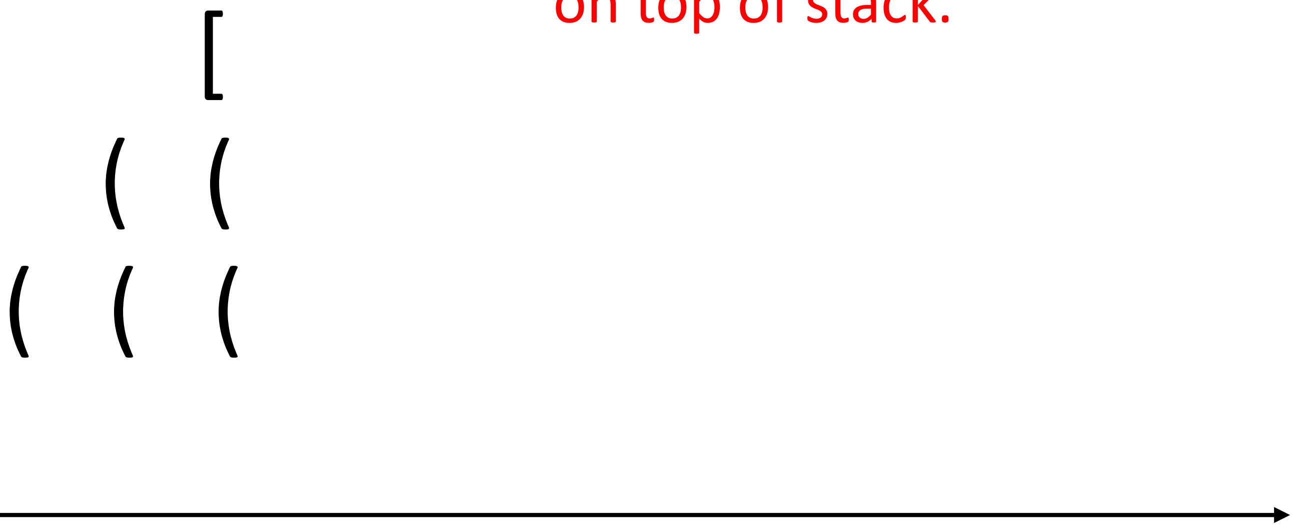
e.g. (([ ])) [ ] { [ ] }



Example 2 - balancing parentheses

e.g. (([ ))] { [ ] }

Does not match left bracket on top of stack.



//   We refer to brackets as “tokens”.   This is the more general term using in
//   string parsing.

Algorithm: decide if parentheses are matched.

```
while (there are more tokens) {
   token  =  get next token
   if token is a left parenthesis
        push(token)
   else {                                          //  token is a right parenthesis
              if stack is empty
                 return false
              else {
                  pop left parenthesis from stack
                  if  popped left parenthesis doesn’t match the right parenthesis
                       return false
               }
         }
}
return  stack.empty   //   true if stack is empty,  false if not.
```

Example 3:   HTML tags

Suppose you want:

I am bold.   *I am italic.*

In html,   you would write:

```
<b>  I am bold.  </b>  < i > I am italic. < /i >
```

HTML Elements

An HTML *element* starts with a start tag.
An HTML *element* ends with an end tag.
These tags can be thought of as brackets.

HTML documents consist of nested HTML *elements*.

```
<html>
<body>
<b>  I am bold </b>
<i>   I am italic </i>
</body>
</html>
```

Suppose you want:

I am bold. ***I am bold and italic.*** *I am italic.*

What if you were to write the following ?

```
<b>  I am bold.  <i>  I am bold and italic. </b>  I am italic. </i>
```

Suppose you want:

I am bold. ***I am bold and italic.*** *I am italic.*

What if you were to write the following ?

<b> I am bold. <i> I am bold and italic. </b> I am italic. </i>

This is *officially* incorrect, because elements are not nested.

```
                < i >
____    <b>     <b>     Error:   mismatch between <i>  </b>
```

Most web browsers will interpret it correctly, however.

I am bold. ***I am bold and italic.*** *I am italic.*

The correct way to write it is:

<b> I am bold. <i> I am bold and italic. </i> </b> <i> I am italic. </i>

< i >

____ < b > < b > < b > ____ < i > ____

What problems can arise if you write it incorrectly?

Suppose you are editing a html document that contains the following:

```
Hello.    <b>  I am bold.

<i>       I am bold and italic. </b>   I  am italic.        < /i >

Bla bla bla  ……
```

Q: What happens if you delete the middle line ?

What problems can arise if you write it incorrectly?

Suppose you are editing a html document that contains the following:

Hello. <b> I am bold.

<i> I am bold and italic. </b> I am italic. < /i >

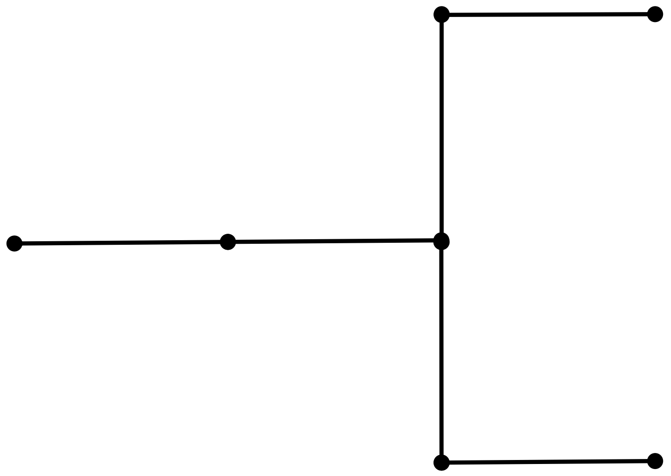
Bla bla bla ……

Q: What happens if you delete the middle line ?

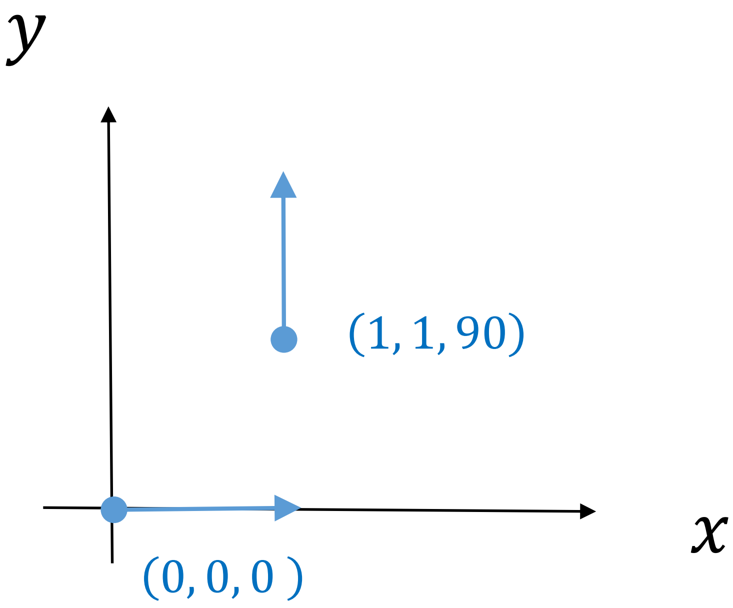
A: … Hello. **I am bold. Bla bla bla ……**

Example 4:   Stacks in Graphics

Define a ‘programming language’ for drawing simple figures like this:



Define a pen position and direction (x, y, θ) where θ is clockwise degrees from x axis.



The initial state of the pen is (0, 0, 0).

Let instructions be symbols :

D - draw unit length line in direction θ (changes (x, y))

R - turn right 90 degrees (changes θ)

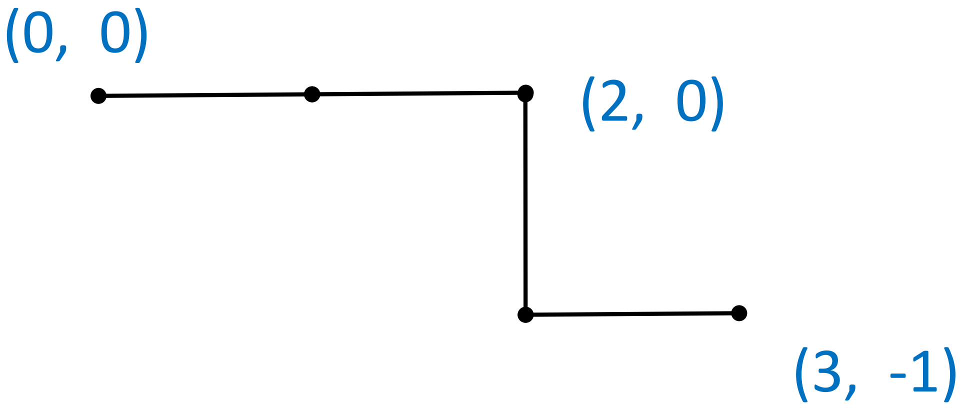
L - turn left 90 degrees (changes θ)

[- push state (x, y, θ)

] - pop state, and go to that state

The initial state of the pen is (0, 0, 0).

D D R D L D



D - draw

R - turn right 90 deg

L - turn left 90 deg

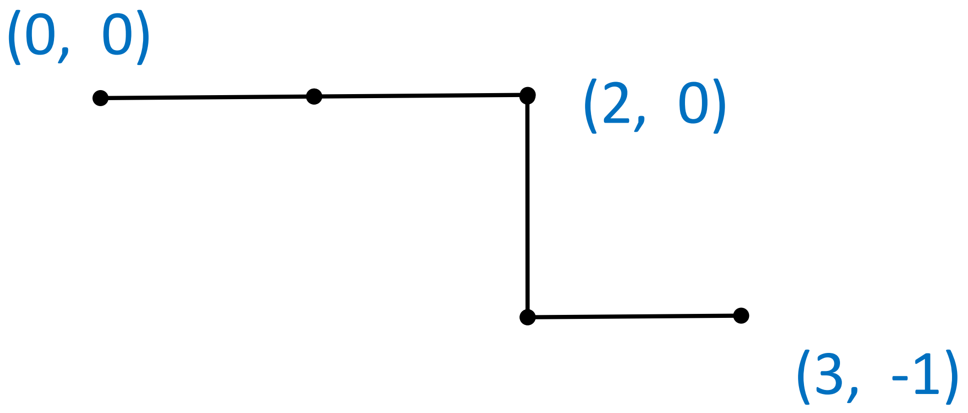
[- push state

] - pop state

The final pen state is (3, -1, 0).

The initial state of the pen is (0, 0, 0).

D D [R D L D]



D - draw

R - turn right 90 deg

L - turn left 90 deg

[- push state

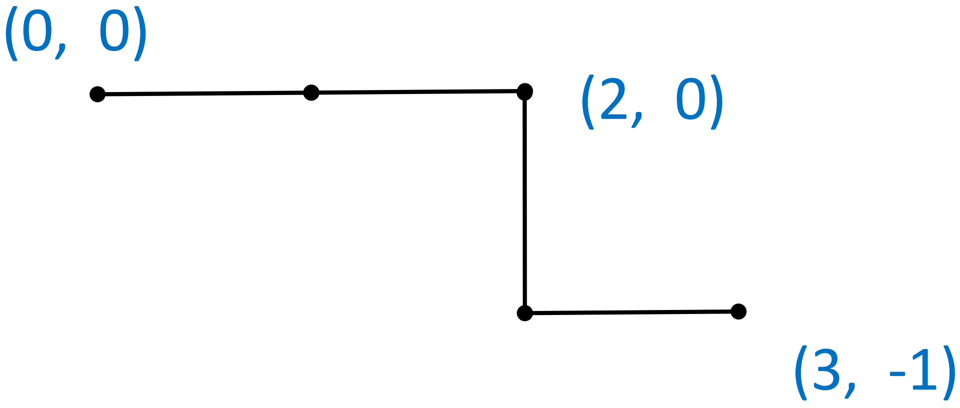
] - pop state

Q: What will be the final pen state ?

A:

The initial state of the pen is (0, 0, 0).

D D [R D L D]



D - draw

R - turn right 90 deg

L - turn left 90 deg

[- push state

] - pop state

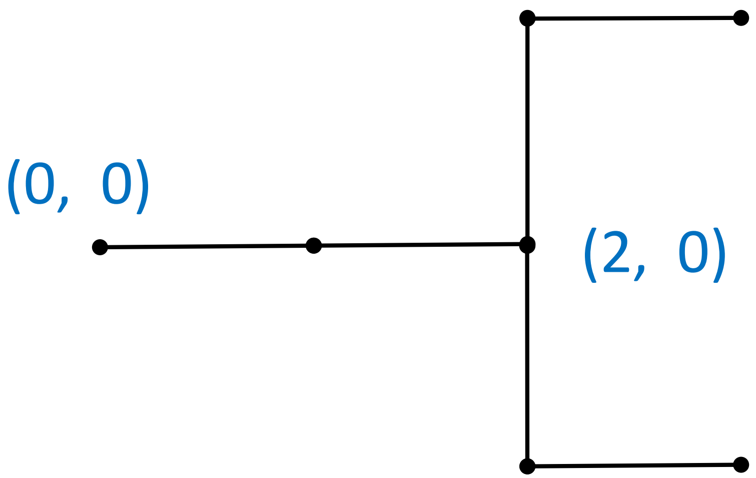
Q: What will be the final pen state ?

A: (2, 0, 0)

The initial state of the pen is (0, 0, 0).

D D [R D L D] L D R D

______ (2, 0, 0) ______



D - draw
R - turn right 90 deg
L - turn left 90 deg
[- push state
] - pop state

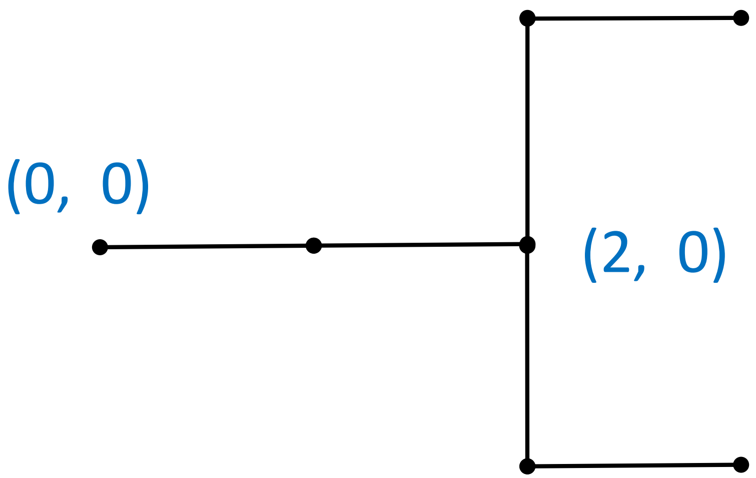
Q: What will be the final pen state ?

A:

The initial state of the pen is (0, 0, 0).

D D [R D L D] L D R D

_____ (2, 0, 0) ______



D - draw

R - turn right 90 deg

L - turn left 90 deg

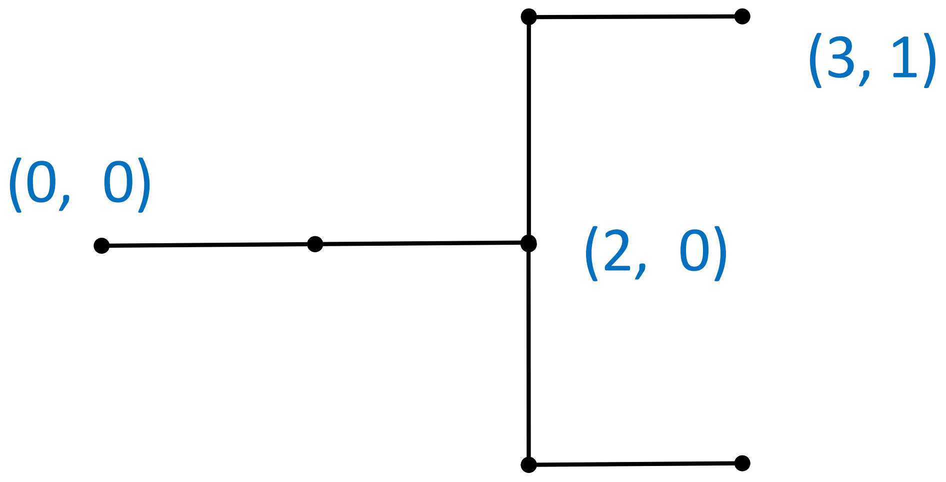
[- push state

] - pop state

Q: What will be the final pen state ?

A: (3, 1, 0)

The initial state of the pen is (0, 0, 0).



D - draw

R - turn right 90 deg

L - turn left 90 deg

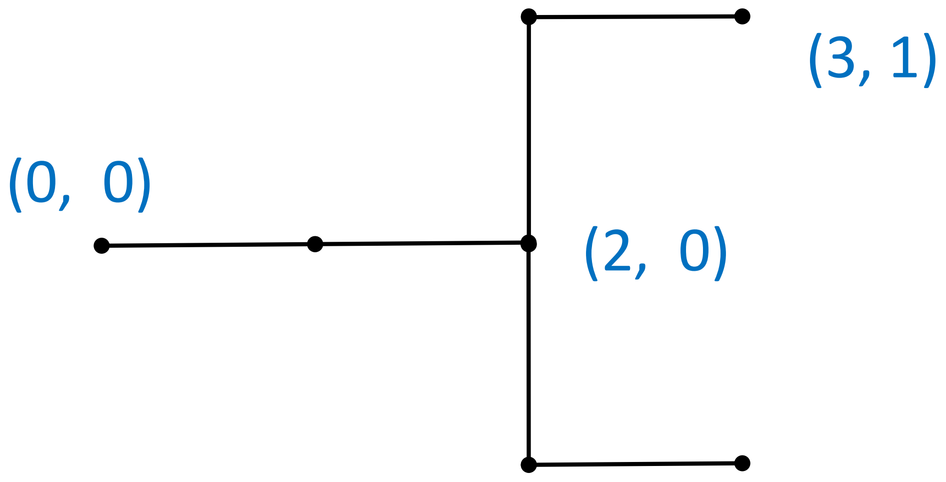
[- push state

] - pop state

Q: What if we add brackets at beginning and ending ?

[D D [R D L D] L D R D]

The initial state of the pen is (0, 0, 0).



D - draw

R - turn right 90 deg

L - turn left 90 deg

[- push state

] - pop state

Q: What if we add brackets at beginning and ending ?

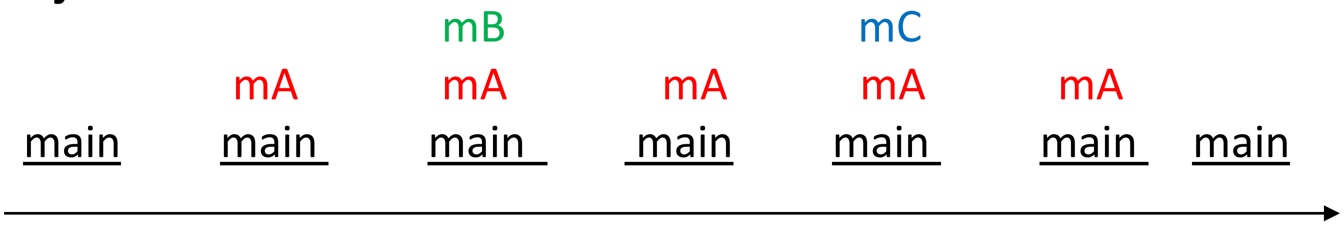
[D D [R D L D] L D R D]

A: The pen state will return to (0, 0, 0).

Example 5 :   “Call Stack”

```
class   Demo {
     void   mA( ) {
                 mB( );
                 mC( );
          }
     void  mB( ) { … }
     void  mC( ) { … }

     void  main( ){
               mA(  );
        }
}
```

```
class   Demo {
    void   mA( ) {
                mB( );
                mC( );
            }
    void  mB( ) { … }
    void  mC( ) { … }

    void  main( ){
                mA(  );
          }
}
```



main

mA
main

mB
mA
main

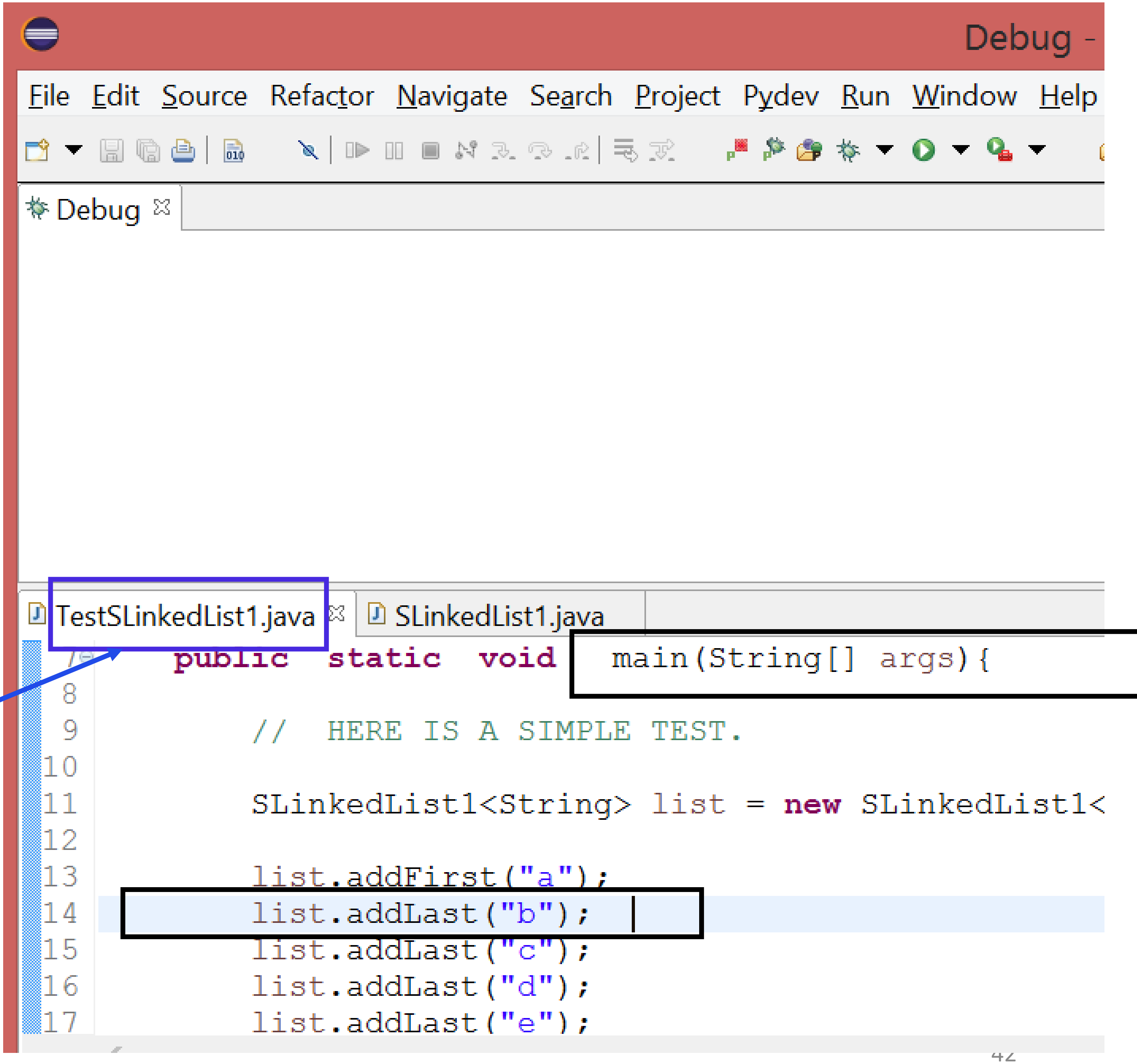
mA
main

mC
mA
main

mA
main

main

Eclipse debug mode



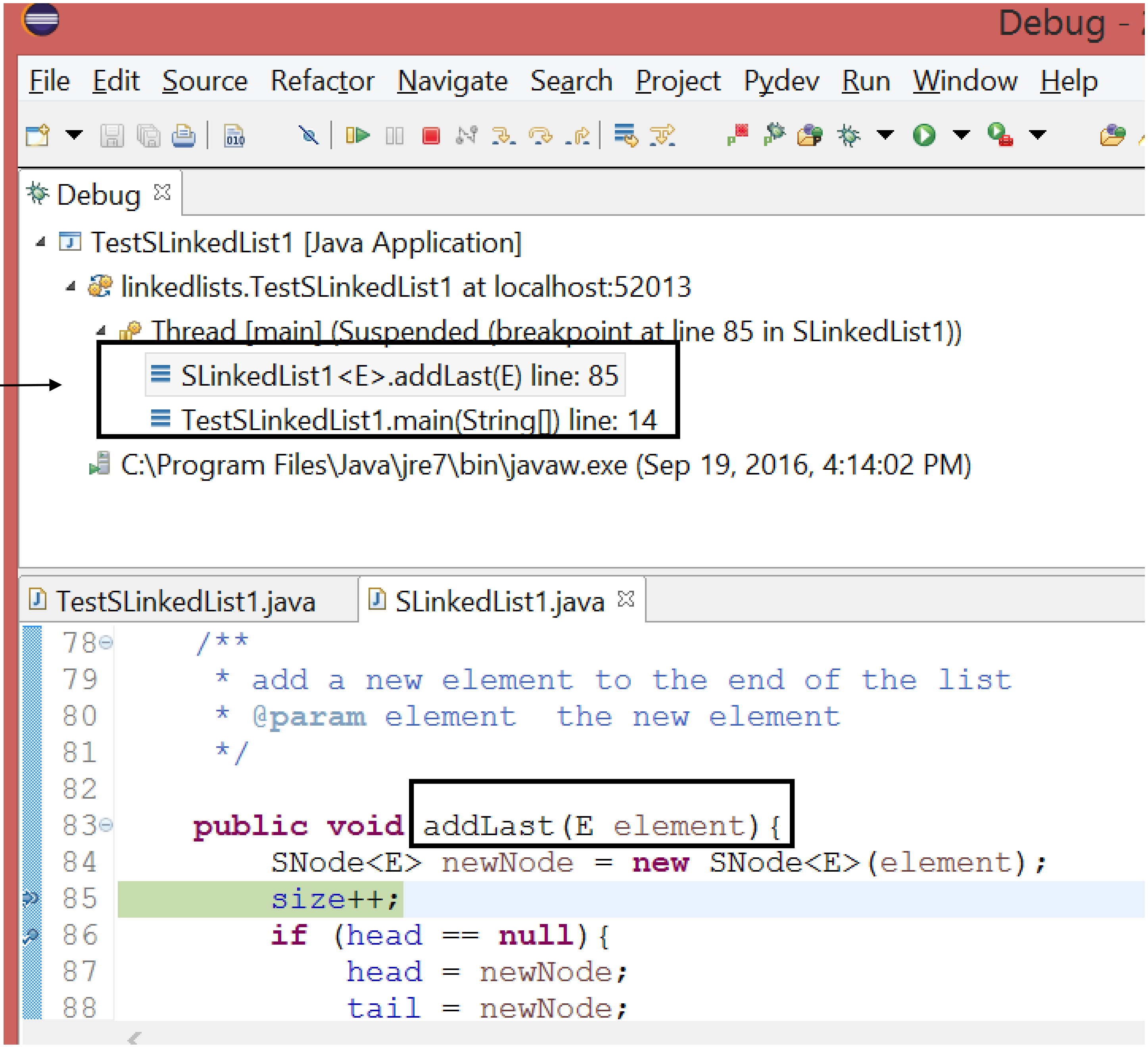
```
    public static void main(String[] args){

        //  HERE IS A SIMPLE TEST.

        SLinkedList1<String> list = new SLinkedList1<

        list.addFirst("a");
        list.addLast("b");
        list.addLast("c");
        list.addLast("d");
        list.addLast("e");
```

TestSLinkedList1's main() method calls addLast() method of SLinkedList class.

Eclipse debug mode

call stack

Breakpoint in the SLinkedList1. addLast method


Debug -
File Edit Source Refactor Navigate Search Project Pydev Run Window Help
Debug
TestSLinkedList1.java
SLinkedList1.java

```
TestSLinkedList1 [Java Application]
  linkedlists.TestSLinkedList1 at localhost:52013
    Thread [main] (Suspended (breakpoint at line 85 in SLinkedList1))
      SLinkedList1<E>.addLast(E) line: 85
      TestSLinkedList1.main(String[]) line: 14
  C:\Program Files\Java\jre7\bin\javaw.exe (Sep 19, 2016, 4:14:02 PM)
```

```java
    /**
     * add a new element to the end of the list
     * @param element  the new element
     */

    public void addLast(E element){
        SNode<E> newNode = new SNode<E>(element);
        size++;
        if (head == null){
            head = newNode;
            tail = newNode;
```