CS240

Nathan Sprague

February 20, 2013

Stack ADT

Stack ADT

- *Stack()* - Creates a new empty stack.
- `isEmpty()` - Returns a boolean value.
- *length()* - Returns number of items in the stack.
- `pop()` - Removes and returns the top item on the stack.
- `peek()` - Retruns a reference to the top item without removing it.
- `push(item)` - Adds the item to the top of the stack.

Exercise

```
1  s = Stack()
2  s.push("A")
3  s.push("B")
4  s.push("C")
5  s.pop()
6  s.push("D")
7
8  while not s.isEmpty():
9      print(s.pop())
```

1: A B C D
2: D B A
3: A B D

Exercise

```
1  s1 = Stack()
2  s2 = Stack()
3  s1.push("A")
4  s1.push("B")
5  s1.push("C")
6
7  while not s1.isEmpty():
8        s2.push(s1.pop())
9
10 while not s2.isEmpty():
11       print(s2.pop())
```

1: A B C
2: C B A
3: CRASH!

Implementation...

Python List Stack Worst-Case Performance

1

Stack()	O(1)
isEmpty()	O(1)
length()	O(1)
pop()	O(1)
peek()	O(1)
push(item)	O(n)

2

Stack()	O(1)
isEmpty()	O(1)
length()	O(1)
pop()	O(1)
peek()	O(1)
push(item)	O(1)

3

Stack()	O(1)
isEmpty()	O(1)
length()	O(1)
pop()	$O(n^2)$
peek()	O(1)
push(item)	$O(n^2)$

Linked List Stack Worst-Case Performance

1

Stack()	O(1)
`isEmpty()`	O(1)
length()	O(1)
`pop()`	O(1)
`peek()`	O(1)
`push(item)`	O(n)

2

Stack()	O(1)
`isEmpty()`	O(1)
length()	O(1)
`pop()`	O(1)
`peek()`	O(1)
`push(item)`	O(1)

3

Stack()	O(1)
`isEmpty()`	O(1)
length()	O(1)
`pop()`	$O(n^2)$
`peek()`	O(1)
`push(item)`	$O(n^2)$

Infix, Postfix and Stacks

Exercise

- Consider this infix expression:
  $2 * 4/8 + 7 * 3$
- Convert to postfix
- Hand evaluate using a stack.
- What maximum depth does the stack reach?
  - 2, 3, 4