Kyla Wilson
Feb 25, 2020
J00813814
Assignment 4

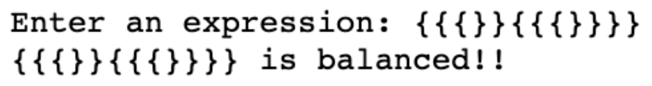
1.) My algorithm is very simple. Whenever an open parenthesis is found, it will increment the depth counter. If a closed parenthesis is found, I will decrement the depth counter by one.

Pseudo code:

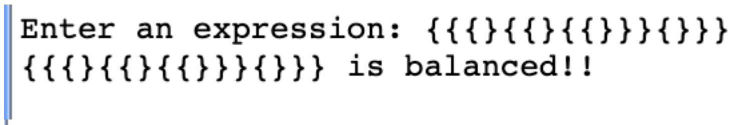
```
While( something in stack ) {
    if( expression == "{" ) {
            depth += 1
    } else if( expression == "}" ) {
            if( stack.pop == "}" or stack.pop == "}" ) {
                    depth -= 1
            } else {
                    return
            }
    }
}
```

Because this algorithm has to transverse through the stack, the worst case will be O(n).
Best case: O(1) (The expression is 2 characters long , empty or the top symbol is not open parenthesis, that way it is safe to say the stack isn't balanced.)
Worst case: O(n) (The expression is balanced)

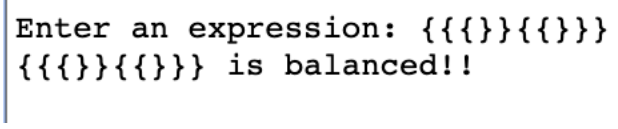
2.)



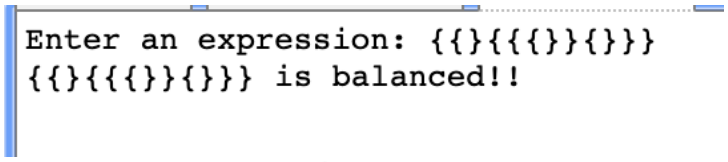
```
Enter an expression: {{{}}{{{}}}}
{{{}}{{{}}}} is balanced!!
```



```
Enter an expression: {{{}{{}{{}}}{}}}
{{{}{{}{{}}}{}}} is balanced!!
```



```
Enter an expression: {{{}}{{}}}
{{{}}{{}}} is balanced!!
```



```
Enter an expression: {{}{{{}}{}}}
{{}{{{}}{}}} is balanced!!
```

```
Enter an expression: {}{{}{}
Expression not balanced!!
Depth: 1
```