

Student: Ovidiu Mura
Problem 385 - Mini Parser

Success

Details >

Runtime: 8 ms, faster than 20.80% of Java online submissions for Mini Parser.

Memory Usage: 41.7 MB, less than 30.00% of Java online submissions for Mini Parser.

Next challenges:

Flatten Nested List Iterator

Ternary Expression Parser

Remove Comments

Show off your acceptance:

f

t

in

Time Submitted	Status	Runtime	Memory
a few seconds ago	Accepted	8 ms	41.7 MB
3 minutes ago	Accepted	8 ms	41.6 MB

i Java

Autocomplete

```
29 * class Solution {
30 *     public NestedInteger deserialize(String s) {
31 *         Stack<NestedInteger> stack = new Stack<>();
32 *         StringBuilder value = new StringBuilder();
33 *         for(int i=0; i<s.length(); i++){
34 *             char c = s.charAt(i);
35 *             if(c == '[') {
36 *                 stack.push(new NestedInteger());
37 *             } else if (c == ']') {
38 *                 if (value.length() > 0) {
39 *                     stack.peek().add(new NestedInteger(Integer.parseInt(value.toString())));
40 *                     value = value.delete(0, value.length());
41 *                 }
42 *                 NestedInteger top = stack.pop();
43 *                 if (stack.isEmpty()) {
44 *                     return top;
45 *                 } else {
46 *                     stack.peek().add(top);
47 *                 }
48 *             } else if(c == ',') {
49 *                 if (value.length() > 0) {
50 *                     stack.peek().add(new NestedInteger(Integer.parseInt(value.toString())));
51 *                     value = value.delete(0, value.length());
52 *                 }
53 *             } else {
54 *                 value.append(c);
55 *             }
56 *         }
57 *         if(value.length() > 0){
58 *             return new NestedInteger(Integer.parseInt(value.toString()));
59 *         }
60 *         return null;
61 *     }
62 * }
```

```
public class Solution {
    public NestedInteger deserialize(String s) {
        Stack<NestedInteger> stack = new Stack<>();
        StringBuilder value = new StringBuilder();
        for(int i=0; i<s.length(); i++){
            char c = s.charAt(i);
            if(c == '[') {
                stack.push(new NestedInteger());
            } else if (c == ']') {
                if (value.length() > 0) {
                    stack.peek().add(new NestedInteger(Integer.parseInt(value.toString())));
                    value = value.delete(0, value.length());
                }
                NestedInteger top = stack.pop();
                if (stack.isEmpty()) {
                    return top;
                } else {
                    stack.peek().add(top);
                }
            } else if(c == ',') {
                if (value.length() > 0) {
                    stack.peek().add(new NestedInteger(Integer.parseInt(value.toString())));
                    value = value.delete(0, value.length());
                }
            } else {
                value.append(c);
            }
        }
        if(value.length() > 0){
            return new NestedInteger(Integer.parseInt(value.toString()));
        }
        return null;
    }
}
```

```
        value.append(c);
    }
}
if(value.length()>0){
    return new NestedInteger(Integer.parseInt(value.toString()));
}
return null;
}
}
```