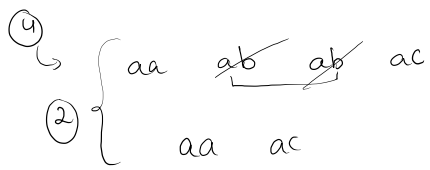


Delete consecutive

Given a sequence of N strings, the task is to check if any two similar words come together then they destroy each other than *print* the number of words left in the sequence after this *pairwise* destruction.

```
Sample Input 0
4
aa ab ab ac

Sample Output 0
2
```



eg2. aa ~~ab~~ ~~ab~~ ab ac
ans = ? \Rightarrow 3

eg3. ~~ab~~ ~~ab~~ ~~ab~~ ~~ab~~ ac
ans = ? = 1

eg4. ~~aa~~ ~~ab~~ ~~ab~~ ~~ab~~ ac
ans = ? = 1

~~aa~~ ~~aa~~ ac
ans = 1

eg5. ab ac ab ac
 \Rightarrow 4

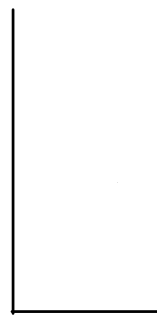
eg. ab ac ac aa } ②

eg.2. aa ab ab ab ac } ③

eg.3. ab ab ab ab ac } ①

eg.4. ab ac ab ac } ④

{ first or st.empty → push
if curr_ele == peek → pop
else push.



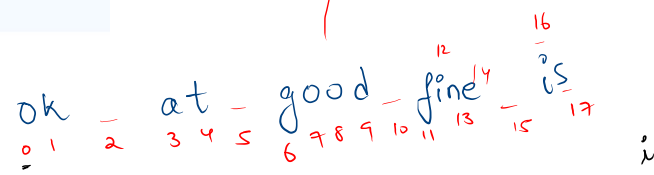
```
1 import java.io.*;
2 import java.util.*;
3
4 public class Solution {
5
6     public static void main(String[] args) {
7         Scanner scn = new Scanner(System.in);
8         int n = scn.nextInt();
9         Stack<String> st = new Stack<>();
10
11         for(int i = 0; i < n; i++){
12             String s = scn.next();
13             if(st.size() != 0 && s.equals(st.peek())){
14                 st.pop();
15             }else{
16                 st.push(s);
17             }
18         }
19         System.out.println(st.size());
20     }
21 }
```

Reverse Words in a Given String

Sample Input 0

```
reverse words in a given string
```

is fine good at ok
②✓



tmp → is

~~fine~~
~~good~~
~~at~~
~~ok~~

if $s[i] \neq ' '$
 $tmp += s[i]$

$s[i] == ' '$
 $st.push(tmp)$
 $tmp = ''$

tmp → "is_fine_good_at_ok"

while $(st.size() != 0)$
 $tmp += " " + st.pop()$

```

1 import java.io.*;
2 import java.util.*;
3
4 public class Solution {
5
6     public static void main(String[] args) {
7         Scanner scn = new Scanner(System.in);
8         String s = scn.nextLine();
9         Stack<String> st = new Stack<>();
10
11         String tmp = "";
12         for(int i = 0; i < s.length(); i++){
13             char ch = s.charAt(i);
14             if(ch != ' '){
15                 tmp += ch;
16             }else{
17                 st.push(tmp);
18                 tmp = "";
19             }
20         }
21         //remove
22         while(st.size() != 0){
23             tmp += " " + st.pop();
24         }
25
26         System.out.println(tmp);
27     }
28 }
29 }

```

game - - - is - - - on - - -

s → game - is - on
0 1 2 3 4 5 6 7 8 9

is
game

tmp → on - is - game

on - is - game

Longest Valid Parentheses 4

Problem

Submissions

Leaderboard

Discussions

Given a string containing just the characters '(' and ')', return the length of the longest valid (well-formed) parentheses substring.

()
)(
))) () () ()
))) () () ()

() () () → 6

(()) → 4

(()) → 2

) (→ 0

(() ()) → 6

) () () → 4

() (())) → 8

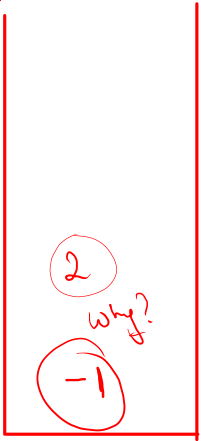
8 \rightarrow $\begin{pmatrix} (&) & (&) & (&) \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 \end{pmatrix}$

ans = ~~ϕ~~ ~~2~~ ~~4~~

if $\rightarrow s[i] == '(' \rightarrow \text{push}(i)$

i

idx



len = $i - \text{peek}$
 $6 - 2 = 4$

$\hookrightarrow ') '$ $\rightarrow \text{st.pop}()$ } to calc. length
find len

A hand-drawn diagram of a U-shaped container, possibly representing a water body or a basin. The container is drawn with two vertical lines and a horizontal base line. Above the right side of the container, there is a curved arrow pointing upwards and to the right.

$s[i] == 'c' \rightarrow \text{push}$

$$\text{len} = i - \text{peek}$$

len = 2
len = 4
len = 2

```
1 class Solution {
2     public int longestValidParentheses(String s) {
3         Stack<Integer> st = new Stack<>();
4         st.push(-1);
5         int ans = 0;
6         int n = s.length();
7         for(int i = 0; i < n; i++){
8             char ch = s.charAt(i);
9             if(ch == '('){
10                 st.push(i);
11             }else{
12                 //remove and find len
13                 st.pop();
14                 if(st.size() == 0){
15                     st.push(i);
16                 }else{
17                     int len = i - st.peek();
18                     ans = Math.max(ans, len);
19                 }
20             }
21         }
22         return ans;
23     }
24 }
```