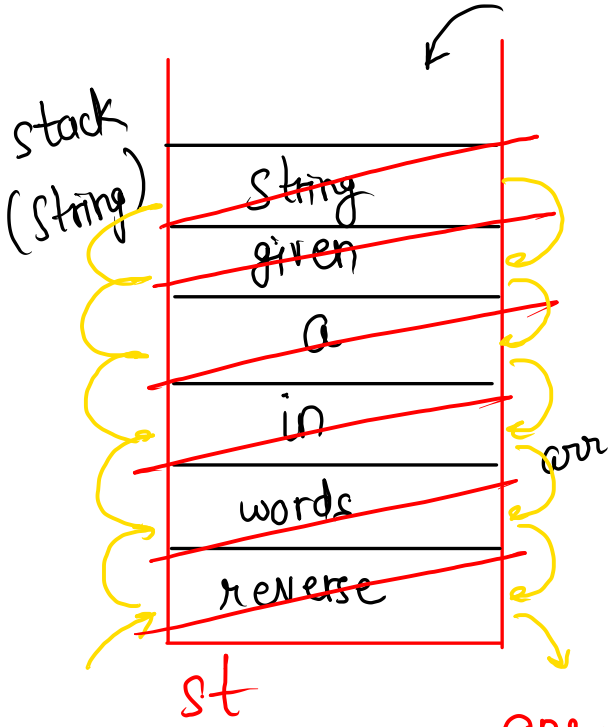
Reverse Words in a Given String

str = "reverse words in a given string"



Inbuilt function

String[] arr = str.split(" ");

reverse	words	in	a	given	string
0	1	2	3	4	5

ans += st.pop() + " "

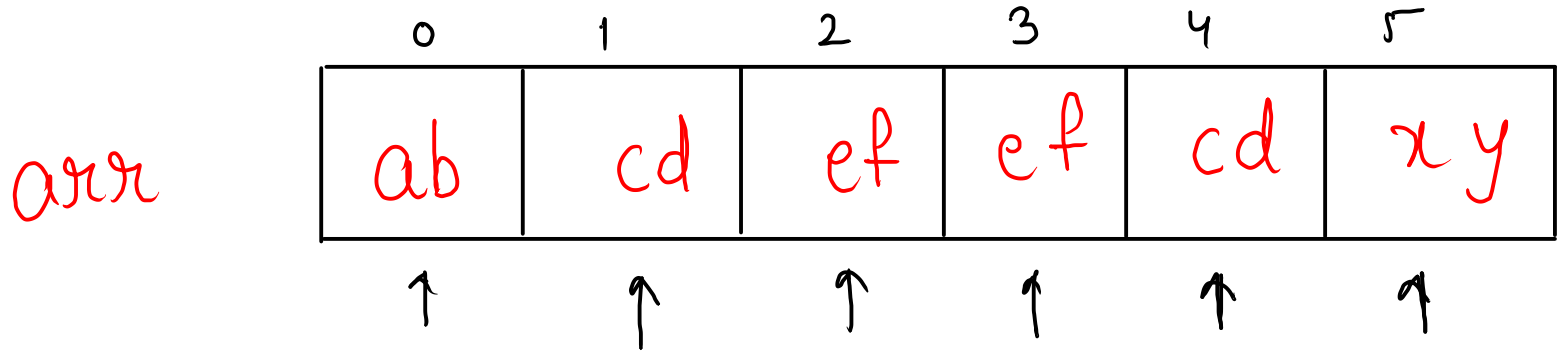
code

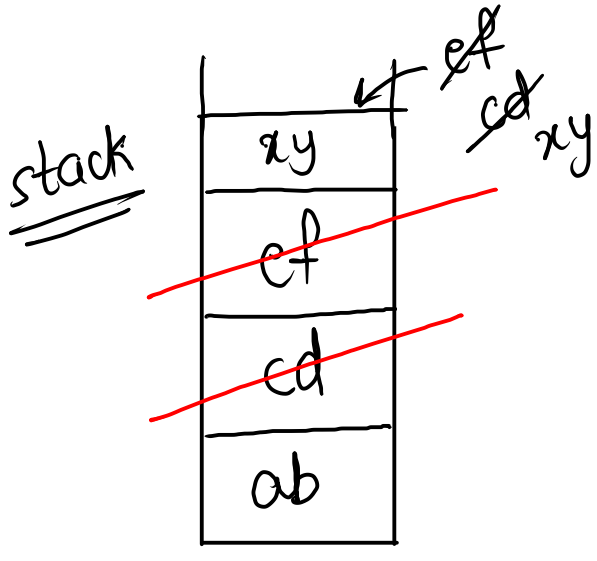
```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    String str = scn.nextLine();
    reverseWords(str);
}
public static void reverseWords(String str) {
    String[] arr = str.split(" ");
    Stack<String> st = new Stack<>();

    for (int i = 0; i < arr.length; i++) {
        st.push( arr[i] );
    }

    String ans = "";
    while ( st.size() > 0 ) {
        ans += (st.peek() + " ");
        st.pop();
    }

    System.out.println(ans);
}
```

T.C = O(N)
(length of stack)

S.C = O(N)

Delete consecutive





return st.size()

psudo code

1) traverse in arr

1.1) check for empty stack && top != curr

push(curr)

1.2) pop()

return size();

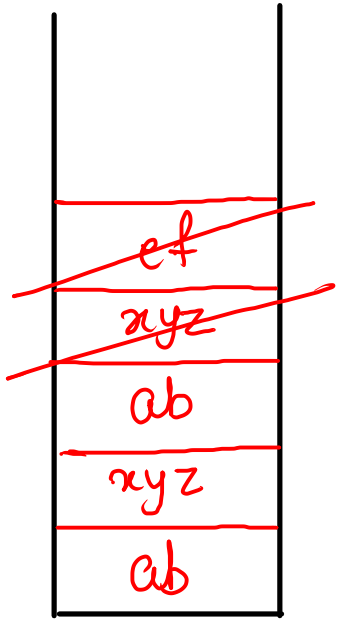
Code

```
public static int deleteConsecutive(String[] arr, int n) {
    Stack<String> st = new Stack<>();
    for (int i = 0; i < arr.length; i++) {
        if ( !st.isEmpty() && st.peek().equals(arr[i]) ) {
            st.pop();
        } else {
            st.push( arr[i] );
        }
    }
    return st.size();
}
```

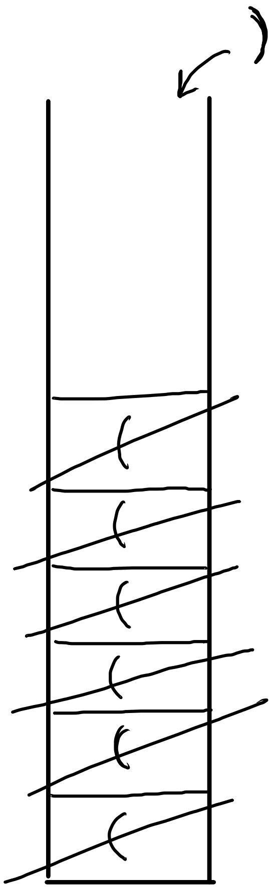
arr [ab , xyz , ab , xyz , ef , ef , xyz]

i

st



valid parentheses 10



str = "((()(()))())"

0 1 2 3 4 5 6 7 8 9 10 11

top == '(' && curr == ')'

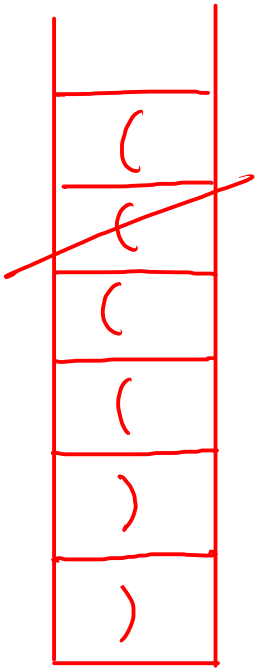
faith :- stack will only contain invalid sequence

true

str = (( )) ((( ) (

0 1 2 3 4 5 6

))(((invalid



if curr == ')'
    then check for '(' at top

else
    push

size > 0 ⟶ false

code

H.W ([{ }])

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    String str = scn.nextLine();
    System.out.println(validParanthesis(str));
}
public static boolean validParanthesis(String str) {
    Stack<Character> st = new Stack<>();
    for (int i = 0; i < str.length(); i++) {
        if ( !st.isEmpty() && str.charAt(i) == ')' && st.peek() == '(' ) {
            st.pop();
        } else {
            st.push( str.charAt(i) );
        }
    }
    return st.size() == 0;
}
```

true
false