1+ Given a string s containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid.

In [1]:

```
def validString(s):
    open1="[{("
    close="]})"
    stack=[]
    for i in s:
        if i in open1:
            stack.append(i)
        elif i in close:
            index = close.index(i)
            if ((len(stack)>0) and (open1[index]== stack[len(stack)-1])):
                stack.pop()

            else:
                return "False"
    if len(stack) == 0:
        return "True"
    else:
        return "False"


str1="{{}[][]}"
"""str1="(]"""
print(validString(str1))

```

```
True
```

2 Given two strings s and t, return true if they are equal when both are typed into empty text editors. '#' means a backspace character.

In [34]:

```python
def backspaceCompare(s, t) -> bool:
    stack1=[]
    stack2=[]
    str1=""
    str2=""
    for i in range(len(s)):
        if (s[i]>'a' and s[i]<'z'):
            stack1.append(s[i])
        elif(s[i]=='#' and len(stack1)>0):
            stack1.pop()
    for i in range(len(t)):
        if (t[i]>'a' and t[i]<'z'):
            stack2.append(t[i])
        elif(t[i]=='#' and len(stack2)>0):
            stack2.pop()
    if(len(stack1)!=len(stack2)):
        return False
    elif(len(stack1)==0 and len(stack2)==0):
        return True
    else:
        for i in range(len(stack1)-1,-1,-1):
            str1+=stack1[i]
            str2+=stack2[i]
            if(str1==str2):
                return True
            else:
                return False
s = "a#c"
t = "b"
print(backspaceCompare(s,t))
```

```
False
```

Given a string ‘s’ of lower and upper case English letters.

A good string is a string which doesn't have two adjacent characters s[i] and s[i + 1] where:

In [10]:

```
def makeGood(s) -> str:
    stack=[]
    i=0
    while(i<len(s)):
        def check(str1):
            if(str1.isupper()):
                return str1.lower()
            else:
                return False
        def check1(str1):
            if(str1.isupper()):
                return False
            else:
                return str1.upper()
        if(len(stack)>0 and check(s[i])== stack[len(stack)-1]):
            stack.pop()
            i=i+1
        elif(len(stack)>0 and check1(s[i])== stack[len(stack)-1]):
            stack.pop()
            i=i+1
        else:
            stack.append(s[i])
            i=i+1
    return "".join(stack)
s = "leEeetcode"
"""s = "abBAcC"
s = "s"""
print(makeGood(s))
```

```
leetcode
```

In [ ]: