1249. Minimum Remove to Make Valid Parentheses

Given a string s of `'('` , `')'` and lowercase English characters.

Your task is to remove the minimum number of parentheses (`'('` or `')'`, in any positions) so that the resulting *parentheses string* is valid and return **any** valid string.

Formally, a *parentheses string* is valid if and only if:

- It is the empty string, contains only lowercase characters, or
- It can be written as `AB` (`A` concatenated with `B`), where `A` and `B` are valid strings, or
- It can be written as `(A)`, where `A` is a valid string.

Example 1:

```
Input: s = "lee(t(c)o)de)"
Output: "lee(t(c)o)de"
Explanation: "lee(t(co)de)" , "lee(t(c)ode)" would also be accepted.
```

Example 2:

```
Input: s = "a)b(c)d"
Output: "ab(c)d"
```

Example 3:

```
Input: s = "))(("
Output: ""
Explanation: An empty string is also valid.
```

Example 4:

```
Input: s = "(a(b(c)d)"
Output: "a(b(c)d)"
```

Constraints:

- `1 <= s.length <= 10<sup>5</sup>`
- `s[i]` is either`'('` , `')'`, or lowercase English letter`.`

```
class Solution:
    def minRemoveToMakeValid(self, s: str) -> str:
```

```
        stack = []
        s = list(s)

        for i in range(len(s)):
            ch = s[i]
            if ch == '(':
                stack.append(i)
            elif ch==')':
                if len(stack)==0 or s[stack[-1]]==')':
                    stack.append(i)
                else:
                    stack.pop()
        # ans = ''
        # for i in range(len(s)):
        #     if i in stack:
        #         continue
        #     else:
        #         ans = ans+s[i]
        # return ans
        for i in stack:
            s[i] = ''
        return ''.join(s)
```