Valid Paranthesis

```
class Solution:
        def isValid(self, s: str) -> bool:
            stack = []
            open_bracket = {'(', '[', '{'}
            closed_bracket = {')', ']', '}'}

            if s == '':
                return False

            for i in s:
                if i in open_bracket:
                    stack.append(i)
                if i in closed_bracket:
                    if stack != [] and ((i == ')' and stack[-1]
                        stack.pop()
                    else:
                        return False
            return len(stack) == 0
```

Initially I used a variable that stored previous pushes, however this was inefficient and a waste of memory

I stuck with using sets for membership checks as it was O(1) so the operation was efficient and quick