# 551 Student Attendance Record I

2018年4月14日 16:42

You are given a string representing an attendance record for a student. The record only contains the following three characters:

1. **'A'** : Absent.
2. **'L'** : Late.
3. **'P'** : Present.

A student could be rewarded if his attendance record doesn't contain **more than one 'A' (absent)** or **more than two continuous 'L' (late)**.

You need to return whether the student could be rewarded according to his attendance record.

**Example 1:**

**Input:** "PPALLP"

**Output:** True

**Example 2:**

**Input:** "PPALLL"

**Output:** False

来自 <https://leetcode.com/problems/student-attendance-record-i/description/>

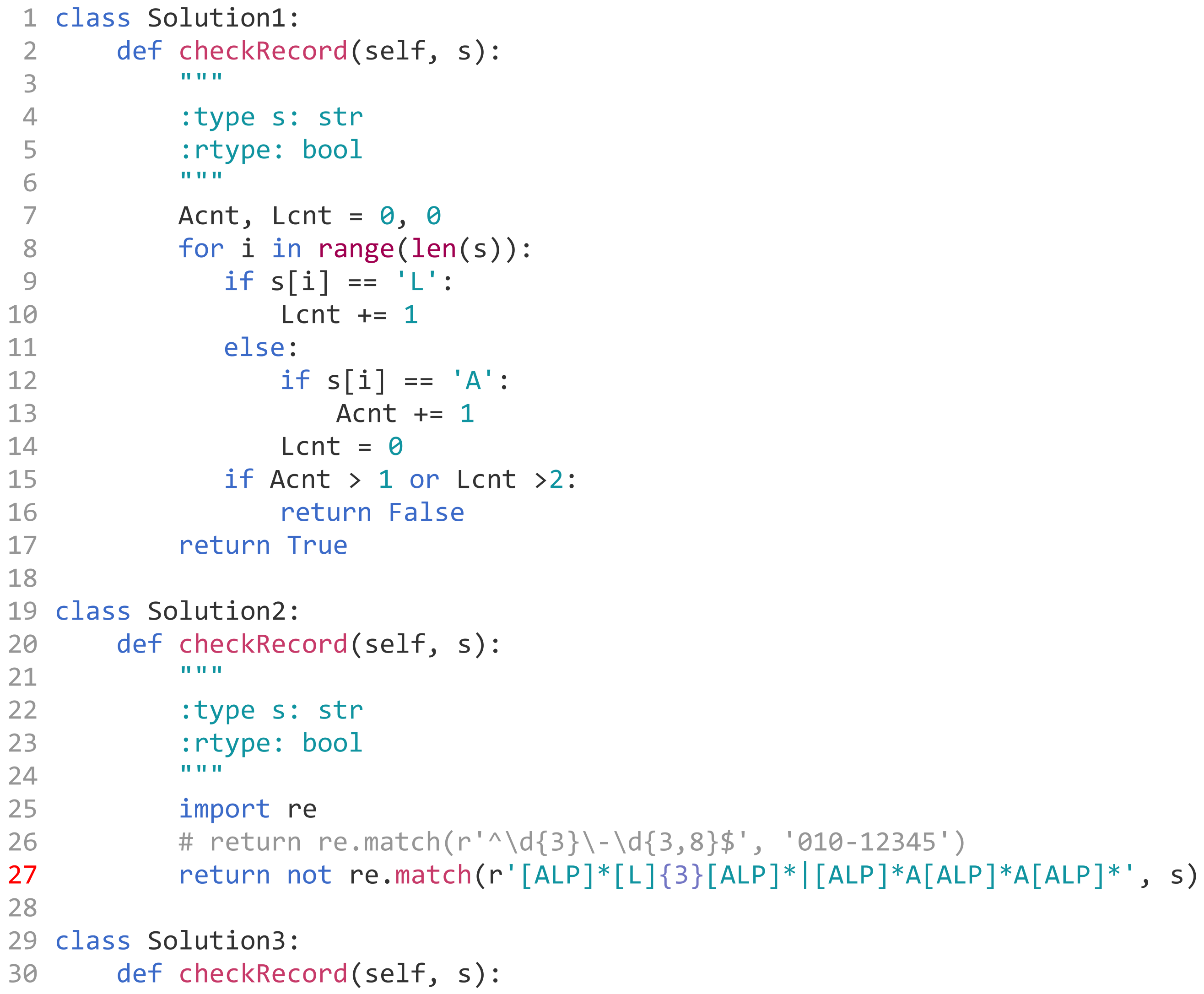
给定一个字符串来代表一个学生的出勤纪录，这个纪录仅包含以下三个字符：

1. **'A'** : Absent，缺勤
2. **'L'** : Late，迟到
3. **'P'** : Present，到场

如果一个学生的出勤纪录中不**超过一个'A'(缺勤)**并且**不超过两个连续的'L'(迟到)**,那么这个学生会被奖赏。

你需要根据这个学生的出勤纪录判断他是否会被奖赏。

## Solution for Python3:



```python
class Solution1:
    def checkRecord(self, s):
        """
        :type s: str
        :rtype: bool
        """
        Acnt, Lcnt = 0, 0
        for i in range(len(s)):
           if s[i] == 'L':
               Lcnt += 1
           else:
               if s[i] == 'A':
                   Acnt += 1
               Lcnt = 0
           if Acnt > 1 or Lcnt >2:
               return False
        return True

class Solution2:
    def checkRecord(self, s):
        """
        :type s: str
        :rtype: bool
        """
        import re
        # return re.match(r'^\d{3}\-\d{3,8}$', '010-12345')
        return not re.match(r'[ALP]*[L]{3}[ALP]*|[ALP]*A[ALP]*A[ALP]*', s)

class Solution3:
    def checkRecord(self, s):
        """
        :type s: str
        :rtype: bool
        """
        import re
        # return not re.match(r'.*LLL.*|.*A.*A.*', s)
        return not re.match(r'.*(LLL|A.*A).*', s)

class Solution4:
    def checkRecord(self, s):
        """
        :type s: str
        :rtype: bool
        """
        import re
        return not re.search('A.*A|LLL', s)


class Solution5:
    def checkRecord(self, s):
        """
        :type s: str
        :rtype: bool
        """
        return s.find('A') == s.rfind('A') and s.find('LLL') == -1
        #find不存在时返回-1，index不存在报错
class Solution6:
    def checkRecord(self, s):
        """
        :type s: str
        :rtype: bool
        """
        return not (s.count('A') > 1) or 'LLL' in s)
```

## Solution for C++:

```
class Solution1 {
public:
    bool checkRecord(string s) {
        int Acnt = 0, Lcnt = 0;
        for (int i = 0; i < s.length(); i++) {
            if (s[i] == 'L') {
                if (++Lcnt == 3)
                    return false;
            } else {
                if (s[i] == 'A') {
                    if (++Acnt == 2)
                        return false;
                }
                Lcnt = 0;
            }
        }
        return true;
    }
```

```cpp
};


class Solution2 {
public:
    bool checkRecord(string s) {
        return !regex_match(s, regex(".*(LLL|A.*A).*"));
        // return !regex_match(s, regex(".*LLL.*|.*A.*A.*"));
    }
};

class Solution3 {
public:
    bool checkRecord(string s) {
        return !regex_search(s, regex("A.*A|LLL"));
    }
};

class Solution4 {
public:
    bool checkRecord(string s) {
        return s.find_first_of('A') == s.find_last_of('A') && s.find("LLL")
== -1;
    }
};
```