

Problem 10: Regular Expression Matching

Problem Information

Difficulty: Hard

Acceptance Rate: 29.98%

Paid Only: No

Tags: String, Dynamic Programming, Recursion

Problem Description

Given an input string `s` and a pattern `p`, implement regular expression matching with support for `.` and `*` where:

`.` Matches any single character. `*` Matches zero or more of the preceding element.

The matching should cover the **entire** input string (not partial).

Example 1:

Input: `s = "aa", p = "a"` **Output:** `false` **Explanation:** `"a"` does not match the entire string `"aa"`.

Example 2:

Input: `s = "aa", p = "a"` **Output:** `true` **Explanation:** `"a"` means zero or more of the preceding element, `'a'`. Therefore, by repeating `'a'` once, it becomes `"aa"`.

Example 3:

Input: `s = "ab", p = ".*"` **Output:** `true` **Explanation:** `.*` means "zero or more (*) of any character (.)".

Constraints:

`s` contains only lowercase English letters. `p` contains only lowercase English letters, `.`, and `*`. It is guaranteed for each

appearance of the character `'*`, there will be a previous valid character to match.

Code Snippets

C++:

```
class Solution {  
public:  
    bool isMatch(string s, string p) {  
  
    }  
};
```

Java:

```
class Solution {  
    public boolean isMatch(String s, String p) {  
  
    }  
}
```

Python3:

```
class Solution:  
    def isMatch(self, s: str, p: str) -> bool:
```