10. Regular Expression Matching

Hard 4552 731 Add to List Share

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

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

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

Note:

- `s` could be empty and contains only lowercase letters `a-z`.
- `p` could be empty and contains only lowercase letters `a-z`, and characters like `.` or `*`.

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: '*' means zero or more of the preceding element, 'a'. Therefore, by
repeating 'a' once, it becomes "aa".
```

```
1  clas
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
   p.ch
20
21
   p.ch
22
23
24
25
26
27
28
29
30 }
```

Your previous

```
Input:
s = "aa"
p = "a*"
Output: true
Explanation: '*' 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 (.)".
```

Example 4:

```
Input:
s = "aab"
p = "c*a*b"
Output: true
Explanation: c can be repeated 0 times, a can be repeated 1 time. Therefore, it
matches "aab".
```

Example 5:

```
Input:
s = "mississippi"
p = "mis*is*p*."
Output: false
```

```
1  clas
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19  p.ch
20
21  p.ch
22
23
24
25
26
27
28
29
30  }
```

Your previous