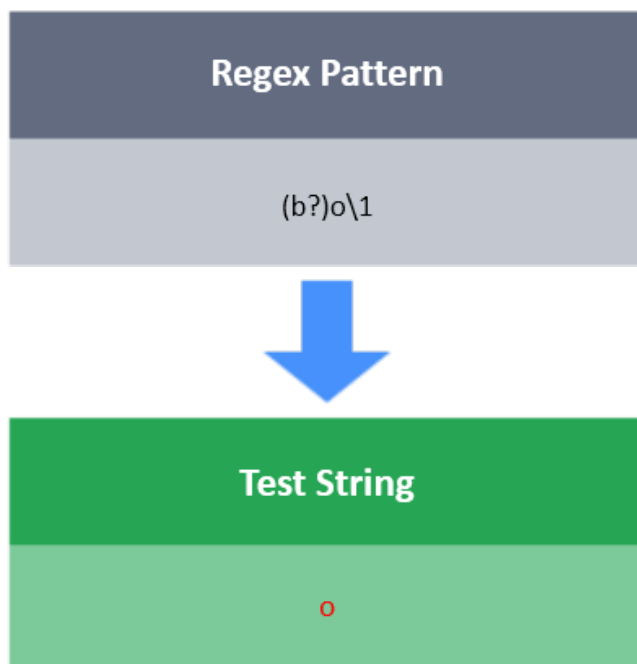# Backreferences To Failed Groups

Backreference to a capturing group that match nothing is different from backreference to a capturing group that did not participate in the match at all.
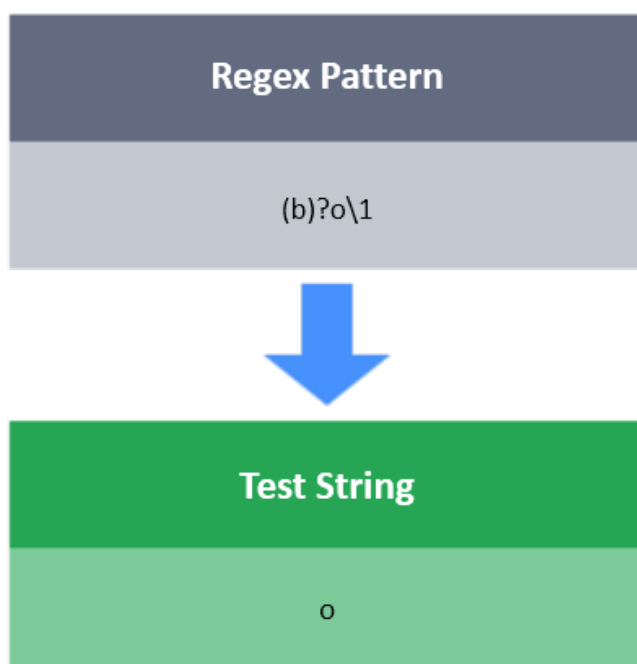
**Capturing group that match nothing**



*In the above image, Regex Pattern is matched with the Test String.*

Here, b? is optional and matches nothing.
Thus, (b?) is successfully matched and capture nothing.
o is matched with **o** and \1 successfully matches the nothing captured by the group.

**Capturing group that didn't participate in the match at all**



*In the above image, Regex Pattern is not matching the Test String.*

In most regex flavors (excluding JavaScript), (b)?o\1 fails to match **o**.

Here, `(b)` fails to match at all. Since, the whole group is optional the regex engine does proceed to match **o**.

The regex engine now arrives at `\1` which references a group that did not participate in the match attempt at all.

Thus, the backreference fails to match at all.

---

## Task

You have a test string $S$.
Your task is to write a regex which will match $S$, with following condition(s):

- $S$ consists of 8 digits.

- $S$ may have "—" separator such that string $S$ gets divided in $4$ parts, with each part having exactly two digits. (Eg. 12-34-56-78)

### Valid $S$

```
12345678
12-34-56-87
```

### Invalid $S$

```
1-234-56-78
12-45-7810
```

## Note

This is a regex only challenge. You are not required to write any code.
You only have to fill the regex pattern in the blank (_____ ).