Year + quarter

Quarter format like Q1 2024 or 2024-Q1.

```
^(Q[1-4][\s\-/]?\d{4}|\d{4}[\s\-/]?Q[1-4])$
```

Token-by-token breakdown

Token	Meaning
`^`	start of string (or line in multiline mode)
`(`	start of a capturing group
`Q`	literal text “Q”
`[1-4]`	any of: 1–4
`[\s\-/]?`	optional (zero or one): any of: whitespace, “-”, “/”
`\d{4}`	exactly 4 times: digit (0–9)
`|`	OR — try the alternative
`\d{4}`	exactly 4 times: digit (0–9)
`[\s\-/]?`	optional (zero or one): any of: whitespace, “-”, “/”
`Q`	literal text “Q”
`[1-4]`	any of: 1–4
`)`	end of group
`$`	end of string (or line in multiline mode)

Quick usage in 7 languages

JavaScript

```
const re = /^(Q[1-4][\s\-\/]?\d{4}|\d{4}[\s\-\/]?Q[1-4])$/;
re.test(value);
```

Python

```
import re
re.match(r"^(Q[1-4][\s\-/]?\d{4}|\d{4}[\s\-/]?Q[1-4])$", value)
```

Java

```
Pattern.compile("^(Q[1-4][\\s\\-/]?\\d{4}|\\d{4}[\\s\\-/]?Q[1-4])$").matcher(value).matches();
```

C# / .NET

```
Regex.IsMatch(value, @"^(Q[1-4][\s\-/]?\d{4}|\d{4}[\s\-/]?Q[1-4])$");
```

Go

```
regexp.MustCompile(`^(Q[1-4][\s\-/]?\d{4}|\d{4}[\s\-/]?Q[1-4])$`).MatchString(value)
```

Ruby

```
/^(Q[1-4][\s\-\/]?\d{4}|\d{4}[\s\-\/]?Q[1-4])$/.match?(value)
```

PHP

```
preg_match('~^(Q[1-4][\s\-/]?\d{4}|\d{4}[\s\-/]?Q[1-4])$~', $value);
```

Common pitfalls

- **Anchored to the whole string.** This pattern uses ^ and $, so it requires the entire input to match. To find it inside a longer text, drop the anchors and use the global (g) flag.
- **Escape it correctly per language.** In Java and JavaScript strings each backslash must be doubled (\\d); in Python, Go, and C# use raw/verbatim strings so the backslashes survive.

• **Validate beyond format.** Matching the format doesn't guarantee the value is real. Confirm the year + quarter against a source of truth (database, API, or checksum) where it matters.