REGEX CHEAT SHEET

regexguide.com/patterns/credit-card-amex.html

American Express

Amex card — starts with 34 or 37, 15 digits total.

```
^3[47]\d{13}$
```

Token-by-token breakdown

Token	Meaning
^	start of string (or line in multiline mode)
3	literal text “3”
[47]	any of: “4”, “7”
\d{13}	exactly 13 times: digit (0–9)
$	end of string (or line in multiline mode)

Quick usage in 7 languages

JavaScript

```
const re = /^3[47]\d{13}$/;
re.test(value);
```

Python

```
import re
re.match(r"^3[47]\d{13}$", value)
```

Java

```
Pattern.compile("^3[47]\\d{13}$").matcher(value).matches();
```

C# / .NET

```
Regex.IsMatch(value, @"^3[47]\d{13}$");
```

Go

```
regexp.MustCompile(`^3[47]\d{13}$`).MatchString(value)
```

Ruby

```
/^3[47]\d{13}$/.match?(value)
```

PHP

```
preg_match('~^3[47]\d{13}$~', $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 american express against a source of truth (database, API, or checksum) where it matters.