RegularExpressions

January 18, 2026

1 Regular Expressions

1.1 External Resources

- C++ Reference for Regular Expressions: https://en.cppreference.com/w/cpp/regex.html
- YouTube Video - Regular Expressions - https://youtu.be/xxOq9YE466w
- YouTube Podcast - Regular Expressions - https://youtu.be/xDOTiIwtHXI
- NotebookLM learning materials - https://notebooklm.google.com/notebook/bfb466f1-6060-4b1a-8406-b50508f3c035

1.2 Overview

- A regular expression (regex or regexp) is a sequence of characters that forms a search pattern. It can be used for string matching and manipulation.
- Regular expressions are commonly used in programming languages for tasks such as input validation, searching, and replacing text.

1.3 Basic Syntax

- `.` : Matches any single character except newline.
- `^` : Matches the start of a string.
- `$` : Matches the end of a string.
- `*` : Matches 0 or more repetitions of the preceding element.
- `+` : Matches 1 or more repetitions of the preceding element.
- `?` : Matches 0 or 1 repetition of the preceding element.
- `[]` : Matches any one of the characters inside the brackets.
- `|` : Acts as a logical OR between expressions.
- `()` : Groups expressions and captures the matched text.
- `{n}` : Matches exactly n repetitions of the preceding element.
- `{n,}` : Matches n or more repetitions of the preceding element.
- `{n,m}` : Matches between n and m repetitions of the preceding element.

1.4 Character Classes

- `\d` : Matches any digit (equivalent to `[0-9]`).
- `\D` : Matches any non-digit (equivalent to `[^0-9]`).
- `\w` : Matches any word character (alphanumeric plus underscore, equivalent to `[a-zA-Z0-9_]`).

- `\W` : Matches any non-word character (equivalent to `[^a-zA-Z0-9_]`).
- `\s` : Matches any whitespace character (spaces, tabs, line breaks).
- `\S` : Matches any non-whitespace character.

1.5 Examples

- To match an email address (e.g., example@example.com):

  `[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}`

- To match a phone number (e.g., 123-456-7890): "'regex 3-3-4

[ ]:
```
#include <iostream>
#include <regex>

using namespace std;
```

[ ]:
```
string email = "this is an email from example@example.com";
```

[12]:
```
regex email_pattern(R"([a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,})");
```

[13]:
```
smatch match;
regex_match(email, match, email_pattern)
```

[13]: false

[10]:
```
cout << match.str(0) << endl;
```

```
example@example.com
```

[ ]:
```

```