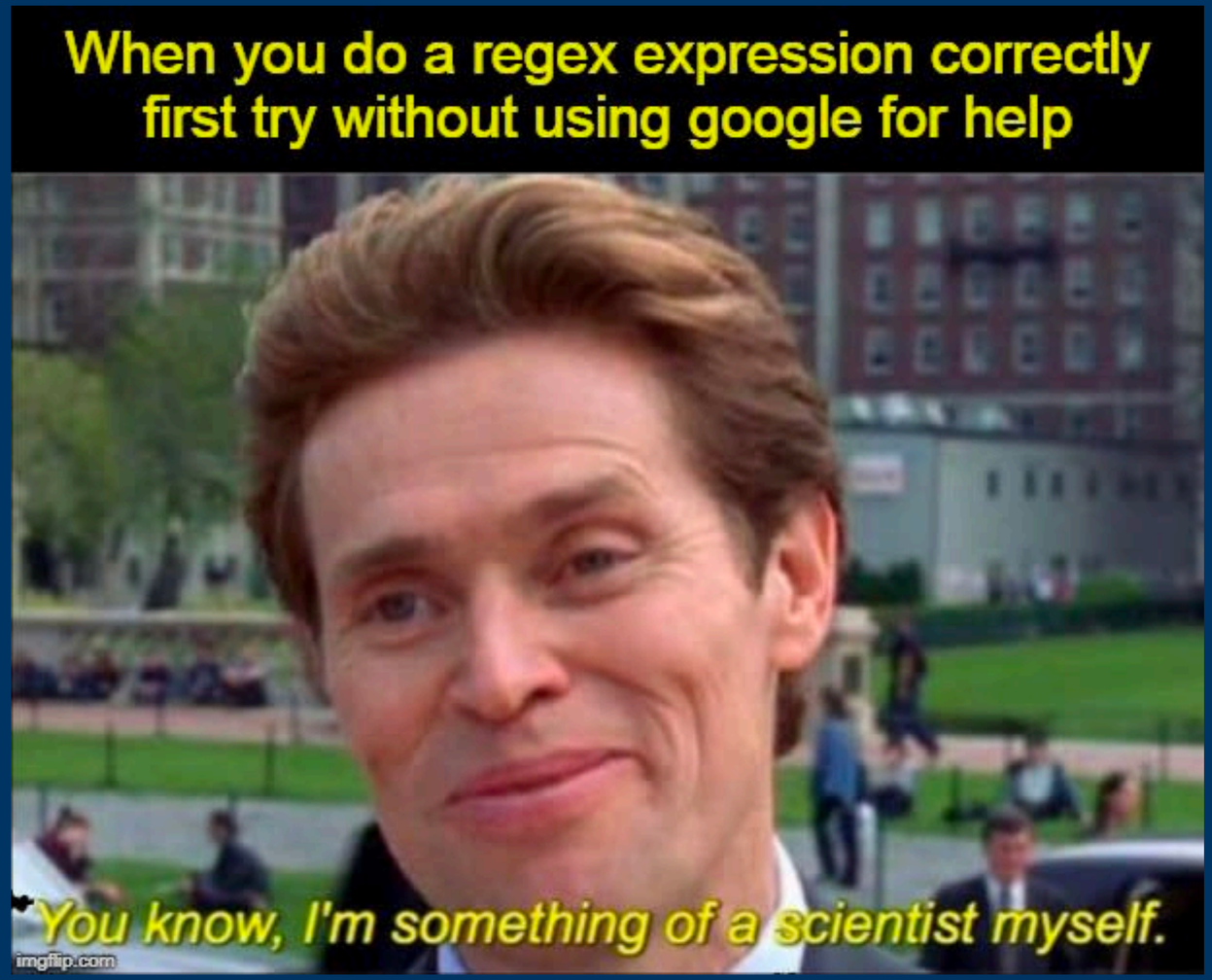
Recitation 8

CS 210

Anamika Lochab



Working with Multi-line Strings

When working with multi-line strings in regex, you typically use the re.MULTILINE flag (or re.M for short).

- This flag changes the behavior of ^ and $ from matching the start and end of the entire string to matching the start and end of each line within the string.
- The re.MULTILINE flag is also helpful when you want to apply regex operations across lines.
- The re.DOTALL flag (or re.S for short) is a different flag that changes the behavior of the dot (‘.’) so that it matches any character, including a newline. By default, the dot does not match newlines.

Greedy vs Non-Greedy Matches

In regex, greedy and non-greedy (or lazy) are terms that describe how a quantifier captures matches.

- **Greedy:** The default behavior of regex quantifiers is greedy, meaning they match as much text as possible. The regex engine will try to match as much of the specified pattern as it can.
  For example, the greedy regex **.*** applied to the string **‘abcdef’** will match the entire string.

- **Non-Greedy (Lazy):** When you make a quantifier non-greedy by placing a ? after it, it matches as little text as necessary, i.e., it stops at the first possible match.
  So, the non-greedy regex **.*?** applied to **‘abcdef’** will match an empty string because * allows for "zero or more" matches, and the ? makes it non-greedy, so it settles for zero characters immediately without consuming any characters.

Substitution with Regular Expressions

The sub() function takes a pattern to match, a replacement string, and the original text. It replaces all occurrences of the pattern with the replacement string.

- re.sub(pattern, repl, string, count=0, flags=0)
- Pattern: The regex pattern to search for.
- Repl: The string to replace matching patterns with.
- string: The string on which to perform the substitutions.
- count: The maximum number of pattern occurrences to replace. `0` means replace all occurrences.
- flags: One or more regex flags to modify the regex's behavior.

Regex Flags

Flags are parameters that you can pass to regex functions like re.search(), re.match(), re.compile() etc., to change how the pattern is interpreted.

- re.IGNORECASE or re.I : This flag allows case-insensitive matching so that characters can match regardless of case.
- re.MULTILINE or re.M : This flag affects the behavior of ^ and $. With this flag, ^ matches at the beginning of the string and at the beginning of each line (immediately following each newline); and $ matches at the end of the string and at the end of each line (immediately preceding each newline).
- re.DOTALL or re.S : This flag makes the . (dot) special character match any character at all, including a newline; without it, the dot matches anything except a newline.
- re.UNICODE or re.U : This flag makes the \w, \W, \b, \B, \s and \S sequences dependent on the Unicode character properties database.
- re.LOCALE or re.L : This flag makes \w, \W, \b, \B, \s and \S dependent on the current locale, which can be set with locale.setlocale(). However, the use of this flag is discouraged as the re.UNICODE flag is the preferred way to handle locale-dependent regex patterns.
- re.ASCII or re.A : This flag makes \w, \W, \b, \B, \d, \D, \s and \S perform ASCII-only matching instead of full Unicode matching. This is only meaningful for Unicode patterns and is ignored for byte patterns.
- re.VERBOSE or re.X : This flag allows you to write regular expressions that are more readable by permitting you to visually separate logical sections of the pattern and add comments.