Regular Expressions

What is it?

- describes a pattern in text
- uses:
  - check if a certain (sub)string exists
  - search/replace characters in a string
- CMSC330 goes more in depth
- https://regexr.com/ can be useful

Basic RegEx Syntax

`/abc/`

- In some languages, regular expressions are enclosed, in '`/` ' or like this `r"abc"`
- Not in Java
  - Special characters like "\" must be escaped
  - A guide on this: https://www.baeldung.com/java-regexp-escape-char
- the above matches:
  - `"abc"`, `"abcdef"`, `"defabc"`, `".=abc==.="`
- but doesn't match:
  - `"cba"`, `"fedcba"`, `"aBc"`

Start/End of line

```
/^abc$/
```

- `^` : start of line
- `$` : end of line
- the above ONLY matches `"abc"`
- and doesn't match anything else

- exercise: how can I match "apple" but not "apples"?

Warning! Every character counts

- `/ s/` is NOT the same as `/  s/`
- the first matches ONE space and then an “s”
- the second matches TWO spaces and then an “s”

Character Sets

```
/[bcd]art/
```

- [] : used to define a character set
- the above matches only ONE letter from “b”, “c”, “d”, and then “art”
- so it matches: `"bart"`, `"cart"`, `"dart"`

- exercise: how can I match `"A+"`, `"B+"`, `"A-"`, and `"B-"` with ONE RegEx?

Character Sets (continued, negated)

```
/[^abc]/
```

- ^ : when used initially inside a character set, negates it
- the above matches anything BUT “a”, “b”, or “c”
- so it matches ONLY the “g” in `"agbc"`
- NOTE: if used outside a character set, it means start of line

Character Ranges

- `[A-Za-z]` matches any letter

  matches any character in `"apple"`, `"bAnanA"`, and `"SUPERstiTION"`

- `[0-9]` matches any digit

  matches all in `"123"`, `"092912"`, and `"2402831608"`

- `[A-Z0-9]` matches any UPPERCASE letter or digit

  matches any character in `"A1"`, `"AREA51"`, but nothing in `"area"`

Built-in Character Ranges

- `\b` : word boundary (spaces between words)
- `\B`: non-word boundary (spaces between characters)
- `\d`: any digit (equivalent to [0-9])
- `\D`: any non-digit (equivalent to [^0-9])
- `\s`: any whitespace character (spaces, tabs, newline, etc.)
- `\S`: any non-whitespace character
- `\w`: any word (equivalent to [A-Za-z0-9_])
- `\W`: any non-word

Character Range Examples

- `/^\w\d$/` matches `"A1"`, `"99"`, `"c6"`, `"_8"`
- `/^\s\D$/` matches `"	s"` (tab), `" 0"` (space), `"	 "` (tab and a space, or vice versa)
- `/^\S$/` matches any single character that's *not* whitespace
- `/^\w\W$/` matches `"A+"`, `"B-"`, `"X "` (space), `"_/"`, `"z^"`

Other useful special characters

- `*` : repeats a character zero or more times

  `/a*/` matches `"a"` and `"aa"` and `""` (empty)

- `+` : repeats a character one or more times

  `/b+/` matches `"b"` and `"bbb"` but NOT `""` (empty)

- `.` : any character

  `/.at/` matches `"cat"`, `"bat"`, `"rat"`, etc. but NOT `"at"`

- `{x}` : specified number x of occurrences

  `/c{3}/` matches exactly 3 “c”s, `/c{4,7}/` matches between 4 and 7 “c”s

Groups

- What if I wanted to extract certain substrings from a match?
  - “cs.umd.edu/class/fall2020/cmsc132/” -> fall2020, cmsc132
  - “cs.umd.edu/class/spring2020/cmsc389E/” -> spring2020, cmsc389E
- Need to use **groups** - marked with `/(.*)/`
  - `/cs[.]umd[.]edu\/class\/(fall|spring\d{4})\/(cmsc.*)/` would work CS class links
- Useful when you want to enforce a format, then take chunks of matches
- Escaped groups: same thing, just doesn’t save the substring
  - Starts with `?:` `(?:cmsc.*)`

Look-around Groups

- Positive lookahead `(?=...)`
  - Find expression A where expression B follows: `A(?=B)`
- Negative lookahead `(?!...)`
  - Find expression A where expression B *does not* follow: `A(?!B)`
- Positive lookbehind `(?<=...)`
  - Find expression A where expression B precedes: `(?<=B)A`
- Negative lookbehind `(?<!...)`
  - Find expression A where expression B *does not* precede: `(?<!B)A`

How to use this in Java: Test and Replace

- Use these packages:
  - `import java.util.regex.*`

- Pattern.matches tests a string:
  - `boolean matches = Pattern.matches(".*and.*", “Bread and butter” );`
  - True or false?

- String’s replaceAll: replace all pattern occurrences with ____:
  - `System.out.println("Magn1ficent7".replaceAll("(?<!n)\\d", "8")));`
  - What will that print?

How to use this in Java: Groups

- Can also pre-compile the regular expression into a Pattern object:
  - `Pattern morningPattern = Pattern.compile(".*morning.*");`

- If groups are needed, use .matcher(…)
  - ```
    Matcher matcher = mostAwesomePattern.matcher(contentString);
    matcher.find()
    System.out.println("First group: " + matcher.group(1) );
    ```
  - .find() returns false if nothing is found
  - Group 0 is the whole match!
    ```

End of Presentation

- RegularExpressionsExample code in Eclipse!