Lecture 14 - Regular Expressions (regex)

Regular Expressions

What are **Regular Expressions**?

Regular Expressions

What are **Regular Expressions**?

Regex: a tool for pattern matching in strings

Regular Expressions

What are **Regular Expressions**?

Regex: a tool for pattern matching in strings

```
^(?:(?:(?:0?[13578]|1[02])(\/|-|\.)31)\1|(?:(?:0?[13-9]|
1[0-2])(\/|-|\.)(?:29|30)\2))(?:(?:1[6-9]|[2-9]\d)?\d{2})$
|^(?:0?2(\/|-|\.)29\3(?:(?:(?:1[6-9]|[2-9]\d)?(?:0[48]|
[2468][048]|[13579][26])|(?:(?:16|[2468][048]|[3579][26])
00))))$|^(?:(?:0?[1-9])|(?:1[0-2]))(\/|-|\.)(?:0?[1-9]|1
\d|2[0-8])\4(?:(?:1[6-9]|[2-9]\d)?\d{2})$
```

Simplest Example

Character to character match

`grep` (**g**lobally search a **r**egular **e**xpression and **p**rint)

Simplest Example

```
grep 'Biology' myCV.txt
```

BIOS 101: Biology for non-majors

`BIOS 185: Introduction to biology for majors`

Matches `Biology`, but not `biology` because regex are **case sensitive**

Metacharacters

Metacharacters have special meanings (i.e. don’t match themselves)

`$`

`^`

`.`

`*`

`?`

`{ }`

`[ ]`

`( )`

`|`

`\`

Wildcard

`.`	Any single character (except `\n`)

Wildcard

```
grep '.iology' myCV.txt
```

BIOS 101: Biology for non-majors

BIOS 185: Introduction to biology for majors

Character classes

`[ ]` Designates a **character class**

List multiple options within `[ ]`

Represents a single character

Character classes

```
grep '[Bb]iology' myCV.txt
```

```
BIOS 101: Biology for non-majors
BIOS 185: Introduction to biology for majors
```

Character classes

Metacharacters lose their special meanings inside `[ ]`

```
grep '.iology' myCV.txt
```

```
BIOS 101: Biology for non-majors
BIOS 185: Introduction to biology for majors
```

```
grep '[.]iology' myCV.txt
```

Returns nothing

Character classes

`[0-9]` – Indicates a range of options

`[A-Za-z0-9_]` Concatenate ranges and character options

`[ \t\n]` Represent whitespace

Character classes (Perl-like)

`\d`	`[0-9]`	Single digit
`\w`	`[A-Za-z0-9_]`	Single alphanumeric character or _
`\s`	`[ \t\n]`	Single whitespace character

Negation of character classes

`[^]` `^` Inside of brackets negates a character class

`[^0-9]` `\D` Single non-digit

`[^A-Za-z0-9_]` `\W` Single character, not alphanumeric or _

`[^ \t\n]` `\S` Single non-whitespace character

Quantifiers

`*` zero or more matches

`+` one or more matches

`?` zero or one match (also makes other qualifiers non-greedy)

`{n}` exactly `n` matches of preceding character

`{m,n}` at least `m` and up to `n` matches

Grouping

A quantifier refers to only the preceding single character/class

( )   Groups characters for quantifiers

Escape character

How to literally match a metacharacter?

\   escapes a metacharacter

Example

2139.Rpomonella.hawthorn.Dowagiac.MI.m

2140.Rpomonella.haw.Dowagiac.MI.m

2000.Rpomonella.Haw.Urbana.IL.f

2001.Rpomonella.Hawthorn.Urbana.IL.f

Example

2139.Rpomonella.hawthorn.Dowagiac.MI.m

2140.Rpomonella.haw.Dowagiac.MI.m

2000.Rpomonella.Haw.Urbana.IL.f

2001.Rpomonella.Hawthorn.Urbana.IL.f

```
[0-9]{4}\.Rpomonella\.[Hh]aw(thorn)?\.[A-Z][a-z]+
\.[A-Z]{2}\.[mf]
```