INFO5002: Intro to Python for Info Sys

Regular Expression

https://docs.python.org/3/howto/regex.html

Northeastern University

Slides created by: Zachary Doucet

A mini language in Python

- Regular expressions (regex) is a language applied to **strings** that allow for the generation of rules.
- Regex makes it easy to verify if user input is of a certain type. e.g. an email address should have an “@” and a “.*{something}*”.

Patterns

- Characters match themselves except for a few key characters which are called metacharacters.
  “test” matches any string that is exactly “test”, “ test” not good.
- . (decimal) matches any character.
- You can define an option (character class) with brackets. Metacharacters don’t mean anything inside brackets.
  “[ab][de]” matches strings “ad”, “ae”, “bd”, “be”
- Can complement the character class with ^ at start.
  “angel[^a]” matches everything that begins with “angel” except for “angela”

- and \

- You can create a range with **-**.

  ”[a-z]”

- \ is an escape character and allows you to match any of the escape characters by prefixing it.

  ”\[” “\\”

Common escapes

- \d: any decimal digit: [0-9]
- \D: any non digit: [^0-9]
- \s: any whitespace character: [\t\n\r\f\v]
- \S: any non-whitespace character: [^ \t\n\r\f\v]
- \w: any alphanumeric character: [a-zA-Z0-9_]
- \W: any non-alphanumeric character: [^a-zA-Z0-9_]

Repeating

- *: zero or more times.
- +: one or more times.
- ?: zero or one times.
- {m,n}: at least m and at most n.

  ”a/{1,3}b” matches “a/b”, “a//b”, “a///b” but not “a////b”
- {m}: exactly m times

In Python

Raw string notation

Pattern object

```
import re

p = re.compile(r"\w*[^z]")
print(p.match("hellotherez"))
```

```
<re.Match object; span=(0, 10), match='hellothere'>
```

End idx (not included)

Matching

- match(): determine if exists a match from start of string.
- search(): find any starting location that matches.
- findall(): find all substrings that match and return as list.
- finditer(): find all substrings that match and return as an iterator.

Match Objects

- match() and search() return either None or a Match object.
- Match objects have following methods:
  - group(): return the matched string
  - start(): return starting pos of match.
  - end(): return ending pos of match (not including).
  - span(): tuple of (start, end)

Metacharacters+

- |: OR: “A|B” => “A”, “B”
- ^: beginning of the line (start of string): “^Hello” => “Hello”
- $: end of the line (end of string): “Bye$” => “Bye”

Let’s practice

- Create a function **validate_email** which takes in an email as a string and uses regular expression to validate if it is a properly formatted email.
- Create a function **validate_phone_number** which takes in a phone number as a string and makes sure that it follows the Canadian convention of (XXX) XXX-XXXX.