Problem: search for strings in other strings

“WTF is a regexp?! It’s short for regular expression.”

Problem: search for strings in other strings

```
>>> my_str = "WTF is a regexp?! It's short for regular expression."
>>>
>>> my_str.find("short")
23

>>> my_str.find("re")
9
```

How do we find **ALL** instances of “`re`” in `my_str`?

Problem: search for strings in other strings

```
def find_all(a_str, sub):
    start = 0
    while True:
        start = a_str.find(sub, start)
        if start == -1: return
        yield start
        start += len(sub) # use start += 1 to find overlapping matches
```

```
>>> list(find_all(my_str, "re"))
[9, 33, 44]
```

How do we find ALL instances of “re” in my_str?

Problem: search for strings in other strings

- Regular expressions are a powerful string manipulation tool
- All modern languages have similar library packages for regular expressions
- Use regular expressions to:
  - Search a string (`finditer`, `search` and `match`)
  - Replace parts of a string (`sub`)
  - Break strings into smaller pieces (`split`)

Problem: search for strings in other strings

```
>>> import re
>>> re.search("re", my_str)
<_sre.SRE_Match object; span=(9, 11), match='re'>

>>> if re.search("re", my_str):
...     print("I found one!")
...
I found one!
```

How do we find ALL instances of “`re`” in `my_str`?

Problem: search for strings in other strings

```
>>> import re
>>> re.search("re", my_str)
<_sre.SRE_Match object; span=(9, 11), match='re'>


>>> [m.start() for m in re.finditer('re', my_str)]
[9, 33, 44]
```

How do we find ALL instances of “`re`” in `my_str`?

Problem: search for strings in other strings

```
>>> if re.search(“re", my_str) or re.search(“re", my_str):
...     print("I found one!")
...
I found one!
```

How do we test whether **`my_str`** contains either **“re”** or **“sh”?**

Problem: search for strings in other strings

```
>>> if re.search(“re|sh", my_str):
...     print("I found one!")
...
I found one!
```

How do we test whether **`my_str`** contains either **“re”** or **“sh”?**

Problem: search for strings in other strings

```
>>> [m.start() for m in re.finditer('re', my_str)]
[9, 33, 44]
>>>
>>> [m.start() for m in re.finditer('re|sh', my_str)]
[9, 23, 33, 44]
```

How do we test whether **`my_str`** contains either **`“re”`** or **`“sh”?`**

Python’s Regular Expression Syntax

- Most characters match themselves
  The regular expression “test” matches the string ‘test’, and only that string
- [x] matches any *one* of a list of characters
  “[abc]” matches ‘a’,‘b’,or ‘c’
- [^x] matches any *one* character that is not included in *x*
  “[^abc]” matches any single character *except* ‘a’,’b’,or ‘c’

Python's Regular Expression Syntax

- "." matches any single character
- Parentheses can be used for grouping
  "(abc)+" matches `'abc', 'abcabc', 'abcabcabc',` etc.
- *x*|*y* matches *x* or *y*
  "this|that" matches `'this' and 'that',` but not `'thisthat'.`

Python's Regular Expression Syntax

- *x** matches zero or more *x*'s
  "a*" matches `''`, `'a'`, `'aa'`, etc.
- *x*+ matches one or more *x*'s
  "a+" matches `'a'`, `'aa'`, `'aaa'`, etc.
- *x*? matches zero or one *x*'s
  "a?" matches `''` or `'a'`
- *x{m, n}* matches *i x*'s, where $m \leq i \leq n$
  "a{2,3}" matches `'aa'` or `'aaa'`

Example: email addresses

How can we easily tell these two apart?

coletrap@uw.edu **spam@go.away**

Here's a pattern to match simple email addresses:

\w+@(\w+\.)+(com|org|net|edu)

```
>>> pat1 = "\w+@(\w+\.)+(com|org|net|edu)"
>>> r1 = re.match(pat,"coletrap@uw.edu")
>>> r1.group()
'coletrap@uw.edu'
```

Extracting bits of the match

We can put parentheses around groups we want to be able to reference

```
>>> pat2 = "(\w+)@((\w+\.)+(com|org|net|edu))"
>>> r2 = re.match(pat2,”coletrap@uw.edu”)
>>> r2.group(1)
'coletrap'
>>> r2.group(2)
‘uw.edu'
>>> r2.groups()
r2.groups()
('coletrap', ‘uw.edu', 'uw.', ‘edu’)
```

Regular Expression Syntax

- “\d” matches any digit; “\D” any non-digit
- “\s” matches any whitespace character; “\S” any non-whitespace character
- “\w” matches any alphanumeric character; “\W” any non-alphanumeric character
- “^” matches the beginning of the string; “$” the end of the string
- “\b” matches a word boundary; “\B” matches a character that is not a word boundary

Sample problem 1

Write a regular expression that matches only DNA sequences (case insensitively).

Regular Expression Quick Guide

^ Matches the beginning of a line
$ Matches the end of the line
. Matches any character
\s Matches whitespace
\S Matches any non-whitespace character
* Repeats a character zero or more times
*? Repeats a character zero or more times (non-greedy)
+ Repeats a chracter one or more times
+? Repeats a character one or more times (non-greedy)
[aeiou] Matches a single character in the listed set
[^XYZ] Matches a single character not in the listed set
[a-z0-9] The set of characters can include a range
(Indicates where string extraction is to start
) Indicates where string extraction is to end

```
>>> if re.match("[ACTGactg]+", "ACGactg"):
...     print("It's DNA!")
... else:
...     print(("It's not DNA"))
...
It's DNA!
```

Sample problem 2

Write a function that uses regular expressions to tell the difference between RNA and DNA

Regular Expression Quick Guide

^ Matches the beginning of a line
$ Matches the end of the line
. Matches any character
\s Matches whitespace
\S Matches any non-whitespace character
* Repeats a character zero or more times
*? Repeats a character zero or more times (non-greedy)
+ Repeats a chracter one or more times
+? Repeats a character one or more times (non-greedy)
[aeiou] Matches a single character in the listed set
[^XYZ] Matches a single character not in the listed set
[a-z0-9] The set of characters can include a range
(Indicates where string extraction is to start
) Indicates where string extraction is to end

```
def which_nuc_acid(sequence):
    if re.match("[actg]+", sequence, re.IGNORECASE):
        print("It's DNA!")
    elif re.match("[acug]+", sequence, re.IGNORECASE):
        print("It's RNA")
    else:
        print("It's neither!")
```

```
>>> which_nuc_acid("uuucgagcuu")
It's RNA
>>>
>>> which_nuc_acid("gattaca")
It's DNA!
>>>
>>> which_nuc_acid("Jurassic Park")
It's neither!
```