Regular Expressions

More Tools
Thousands of papers and theses written in LaTeX
Thousands of papers and theses written in LaTeX

Granger’s work on graphs \cite{dd-gr2007,gr2009}, particularly ones obeying Snape’s Inequality \cite{snape87} (but see \cite{quirrell89}), has opened up new lines of research. However, studies at Unseen University \cite{stibbons2002,stibbons2008} highlight several dangers.

\vspace{1cm}
Thousands of papers and theses written in LaTeX

Granger’s work on graphs \cite{dd-gr2007,gr2009}, particularly ones obeying Snape’s Inequality \cite{snape87} (but see \cite{quirrell89}), has opened up new lines of research. However, studies at Unseen University \cite{stibbons2002, stibbons2008} highlight several dangers.

All share a common bibliography
Thousands of papers and theses written in LaTeX

Granger's work on graphs \cite{dd-gr2007,gr2009}, particularly ones obeying Snape's Inequality \cite{snape87} (but see \cite{quirrell89}), has opened up new lines of research. However, studies at Unseen University \cite{stibbons2002, stibbons2008} highlight several dangers.

All share a common bibliography

Want to see how often citations appear together
Thousands of papers and theses written in LaTeX

Granger’s work on graphs \cite{dd-gr2007,gr2009}, particularly ones obeying Snape’s Inequality \cite{s Snape87} (but see \cite{quirrell89}), has opened up new lines of research. However, studies at Unseen University \cite{stibbons2002, stibbons2008} highlight several dangers.

All share a common bibliography

Want to see how often citations appear together

First step: extract citation sets from documents
Granger’s work on graphs \cite{dd-gr2007,gr2009}, particularly ones obeying Snape’s Inequality \cite{snape87} (but see \cite{quirrell89}), has opened up new lines of research. However, studies at Unseen University \cite{stibbons2002, stibbons2008} highlight several dangers.
Granger’s work on graphs \cite{dd-gr2007,gr2009}, particularly ones obeying Snape’s Inequality \cite{snape87} (but see \cite{quirrell89}), has opened up new lines of research. However, studies at Unseen University \cite{stibbons2002, stibbons2008} highlight several dangers.

Multiple labels separated by commas
Citations enclosed in \cite{...}

Granger’s work on graphs \cite{dd-gr2007,gr2009}, particularly ones obeying Snape’s Inequality \cite{snape87} (but see \cite{quirrell89}), has opened up new lines of research. However, studies at Unseen University \cite{stibbons2002, stibbons2008} highlight several dangers.

Multiple labels separated by commas

May be white space
Citations enclosed in \cite{...}

Granger’s work on graphs \cite{dd-gr2007,gr2009}, particularly ones obeying Snape’s Inequality \cite{snape87} (but see \cite{quirrell89}), has opened up new lines of research. However, studies at Unseen University \cite{stibbons2002, stibbons2008} highlight several dangers.

Multiple labels separated by commas

May be white space (including line breaks)
Granger’s work on graphs \cite{dd-gr2007,gr2009}, particularly ones obeying Snape’s Inequality \cite{snape87} (but see \cite{quirrell89}), has opened up new lines of research. However, studies at Unseen University \cite{stibbons2002, stibbons2008} highlight several dangers.
Idea #1: capture everything in cite{...} in a group

print re.search('cite{(.+)}', 'a \cite{X} b').groups()

(\'X\',)
Idea #1: capture everything in cite{...} in a group

```
print re.search('cite{(.+)}', 'a \cite{X} b').groups()
('X',)
```

What about multiple citations?

```
print re.search('cite{(.+)}', 'a \cite{X} b \cite{Y} c').groups()
('X} b \cite{Y',)
```
Idea #1: capture everything in cite{...} in a group

```python
print re.search('cite{(.+)}', 'a \cite{X} b').groups()

(‘X’,)
```

What about multiple citations?

```python
print re.search('cite{(.+)}', 'a \cite{X} b \cite{Y} c').groups()

(‘X} b \cite{Y’,)
```
Idea #1: capture everything in \cite{...} in a group

```
prire.search('cite{(.+)}', 'a \cite{X} b').groups()

('X',)
```

What about multiple citations?

```
prire.search('cite{(.+)}', 'a \cite{X} b \cite{Y} c').gro

('X} b \cite{Y',)
```

Matching is greedy
Idea #2: match everything inside '{\}' except '{'}'
Idea #2: match everything inside '{}' except '}'

Use '^[^}]' to negate the set containing only '}'
Idea #2: match everything inside '{}' except '}'
Use '^[^}]' to negate the set containing only '}'

print re.search('cite{{([^}]*)}}', 'a \cite{X} b').groups()

( 'X', )
Idea #2: match everything inside '{}' except '}'
Use '([^}]\+)' to negate the set containing only '}'

```python
print re.search('cite{([^}]\+)}', 'a \cite{X} b').groups()
(\'X\',)
```
Idea #2: match everything inside '{}' except '}'
Use '[^{}]' to negate the set containing only '}'

```python
print re.search('cite{([^}]*)}', 'a \cite{X} b').groups()
(\'X\',)
```

What about multiple citations?
Idea #2: match everything inside '{}' except '}'
Use '[^}]' to negate the set containing only '}'

print re.search('cite{([^]+)}', 'a \cite{X} b').groups()
('X',)

What about multiple citations?

print re.search('cite{([^]+)}', 'a \cite{X} b \cite{Y} c').groups()
('X',)
Idea #2: match everything inside '{}' except '}'
Use '[^{}]' to negate the set containing only '}'

```python
print re.search('cite{([^}]+)}', 'a \cite{X} b').groups()
('X',)
```

What about multiple citations?

```python
print re.search('cite{([^}]+)}', 'a \cite{X} b \cite{Y} c').groups()
('X',)
```

Need to extract all matches, not just the first
Idea #3: use re.findall instead of re.search
Idea #3: use re.findall instead of re.search

"A programmer is only as good as her knowledge of her language's libraries."
Idea #3: use re.findall instead of re.search

"A programmer is only as good as her knowledge of her language's libraries."

print re.findall('cite{([^}]*)}', 'a cite{X} b cite{Y} c')

['X', 'Y']
Idea #3: use re.findall instead of re.search

"A programmer is only as good as her knowledge of her language's libraries."

```python
print re.findall('cite{([\^]\+)}', 'a cite{X} b cite{Y} c')
['X', 'Y']
```
Idea #3: use `re.findall` instead of `re.search`

"A programmer is only as good as her knowledge of her language's libraries."

```python
print re.findall('cite{([^}]*)}', 'a \cite{X} b \cite{Y} c')
['X', 'Y']
```

What about spaces?
Idea #3: use re.findall instead of re.search

"A programmer is only as good as her knowledge of her language's libraries."

print re.findall('cite{\([^}\]+)}', 'a \cite{X} b \cite{Y} c')

['X', 'Y']

What about spaces?

print re.search('cite{\([^]\]+)}', 'a \cite{ X} b \cite{Y } c')

[' X', ' Y']
Could tidy this up after matching using string.strip()
Could tidy this up after matching using `string.strip()`

Let's modify the pattern instead
Could tidy this up after matching using string.strip()
Let's modify the pattern instead

```python
print re.findall('cite{.*([^}]*)+}.*', 'a cite{X} b cite{Y}

['X', 'Y']
```
Could tidy this up after matching using `string.strip()`
Let's modify the pattern instead

```python
print re.findall('cite{[^}]+}', 'a cite{X} b cite{Y} c')
```

```
['X', 'Y ']
```
Could tidy this up after matching using string.strip()
Let's modify the pattern instead

```python
print re.findall('cite{\s*([^}\]+)\s*}', 'a cite{ X} b cite{ Y} c')
['X', 'Y ']
```

Still capturing the space after 'Y'
Could tidy this up after matching using string.strip()
Let's modify the pattern instead

```python
print re.findall('cite{\s*([^}]+)\s*}', 'a \cite{ X} b \cite{Y} c')
```
```
['X', 'Y ']
```

Still capturing the space after 'Y'

Match the word-to-nonword transition as well
Could tidy this up after matching using string.strip()

Let's modify the pattern instead

```
print re.findall('cite{\s*([^\}]+)\s*}', 'a cite{ X} b cite{ Y} c')
['X', 'Y ']
```

Still capturing the space after 'Y'

Match the word-to-nonword transition as well

```
print re.findall('cite{\s*[\^\}]+\s*}', 'a cite{ X} b cite{ Y} c')
['X', 'Y']
```
Could tidy this up after matching using `string.strip()`

Let's modify the pattern instead

```
print re.findall('cite{\s*\b([^}]\+)+\b\s*}', 'a \cite{ X} b \cite{ Y} c')
```

```
[ 'X', 'Y ']
```

Still capturing the space after 'Y'

Match the word-to-nonword transition as well

```
print re.findall('cite{\s*\b([^}]\+)+\b\s*}', 'a \cite{ X} b \cite{ Y} c')
```

```
[ 'X', 'Y ']
```
What about multiple labels in a single citation?
What about multiple labels in a single citation?

print re.findall('cite{\s*\b([^}]\b\s*}', 'cite{X,Y} ')
['X, Y']

print re.findall('cite{\s*\b([^}]\b\s*}', 'cite{X, Y, Z} ')
['X, Y, Z']
What about multiple labels in a single citation?

```
print re.findall('cite{\s*\b[^}\]+\b\s*}', '\cite{X,Y} ')
['X, Y']
print re.findall('cite{\s*\b[^}\]+\b\s*}', '\cite{X, Y, Z} ')
['X, Y, Z']
```

Actually can be done, but it's very complex
What about multiple labels in a single citation?

```
print re.findall('cite{\s*\b([^}\]+)\b\s*}', '\cite{X,Y} ')
['X, Y']
```

```
print re.findall('cite{\s*\b([^}\]+)\b\s*}', '\cite{X, Y, Z} ')
['X, Y, Z']
```

Actually can be done, but it's very complex

Use `re.split()` to break matches on `\s*,\s*`
# Start with a working skeleton.

def get_citations(text):
    '''Return the set of all citation tags found in a block of text.'''
    return set()

if __name__ == '__main__':
    test = '''
Granger's work on graphs \cite{dd-gr2007,gr2009}, particularly ones obeying Snape's Inequality \cite{snape87} (but see \cite{quirrell89}), has opened up new lines of research. However, studies at Unseen University \cite{stibbons2002, stibbons2008} highlight several dangers.'''

    print get_citations(test)

    set([])
import re

CITE = 'cite{\s*\b[^}\]+\b\s*}'
SPLIT = '\s*,\s*'

def get_citations(text):
    '''Return the set of all citation tags found in a block of text.'''

    result = set()
    match = re.findall(CITE, text)
    if match:
        for citation in match:
            cites = re.split(SPLIT, citation)
            for c in cites:
                result.add(c)

    return result
import re

CITE = re.compile('cite{\s*\b[^}\]+\b\s*}')
SPLIT = re.compile('\s*,\s*')

def get_citations(text):
    '''Return the set of all citation tags found in a block of text.'''

    result = set()
    match = CITE.findall(text)
    if match:
        for citations in match:
            label_list = SPLIT.split(citations)
            for label in label_list:
                result.add(label)

    return result
import re

CITE = re.compile('cite\{\s*\b([^}\]+)\b\s*\}')
SPLIT = re.compile('\\s*,\\s*')

def get_citations(text):
    '''Return the set of all citation tags found in a block of text.'''

    result = set()
    match = CITE.findall(text)
    if match:
        for citations in match:
            label_list = SPLIT.split(citations)
            for label in label_list:
                result.add(label)

    return result
# Now test it all out.

```python
if __name__ == '__main__':
    test = '''
Granger's work on graphs \cite{dd-gr2007,gr2009},
particularly ones obeying Snape's Inequality
\cite{ snape87 } (but see \cite{quirrell89}),
has opened up new lines of research. However,
studies at Unseen University \cite{stibbons2002,
stibbons2008} highlight several dangers.''

    print get_citations(test)

     'snape87', 'quirrell89'])
```
created by

Greg Wilson

June 2010