Json Parser

```
JSON = grammar("""
object => { } | { members }
members => pair , members | pair
pair => string : value
array => [[] []] | [[] elements []]
elements => value , elements | value
value => string | number | object | array | true | false | null
string => "[^"]*"
number => int frac exp | int frac | int exp | int
int => -?[1-9][0-9]*
frac => [.][0-9]+
exp => [eE][-+]?[0-9]+
""", whitespace='\s*')
```

Inverse function

```
def slow_inverse(f, delta=1/1024.):
    """Given a function y = f(x) that is a monotonically increasing
    non-negatve numbers, return the function x = f_1(y) that is an
    inverse, picking the closest value to the inverse, within delta
    def f_1(y):
        x = 0
        while f(x) < y:
            x += delta
        # Now x is too big, x-delta is too small; pick the closest
        return x if (f(x)-y < y-f(x-delta)) else x-delta
    return f_1

def inverse(f, delta = 1/1024.):
    """Given a function y = f(x) that is a monotonically increasing
    non-negatve numbers, return the function x = f_1(y) that is an
    inverse, picking the closest value to the inverse, within delta
    def f_1(y):
        lo, hi = find_bounds(f,y)
        return binary_search(f,y,lo,hi,delta)
    return f_1
```

Find Html Tags

```
8
9  import re
0
1  def findtags(text):
2      parms = '(\w+\s*=s*"[^"]*"\s*)*'
3      tags = '(<\s*\w+\s*' + parms + '\s*/?>)'
4      return re.findall(tags,text)
5
```