```
    from __future__ import unicode_literals
    from six import text_type

    def hey(s):
5       s = text_type(s).strip()
        criteria = [
            lambda s: 'Fine. Be that way!'  if not s else None,
            lambda s: 'Woah, chill out!'    if s.isupper() else None,
            lambda s: 'Sure.'               if s.endswith('?') else None,
10          lambda s: 'Whatever.'
        ]

        for c in criteria:
            rv = c(s)
15          if rv:
                return rv
```

```
    gifts = [
        'a Partridge in a Pear Tree',
        'two Turtle Doves',
        'three French Hens',
 5      'four Calling Birds',
        'five Gold Rings',
        'six Geese-a-Laying',
        'seven Swans-a-Swimming',
        'eight Maids-a-Milking',
10      'nine Ladies Dancing',
        'ten Lords-a-Leaping',
        'eleven Pipers Piping',
        'twelve Drummers Drumming',
    ]
15
    days = [
        'first',
        'second',
        'third',
20      'fourth',
        'fifth',
        'sixth',
        'seventh',
        'eighth',
25      'ninth',
        'tenth',
        'eleventh',
        'twelfth',
    ]
30

    def verse(day):
        pre = 'On the {} day of Christmas '.format(days[day-1]) +\
            'my true love gave to me,'
35
        # comma-separated list of gifts from today's down to doves, inclusive
        todays_gifts = ', '.join(gifts[day-1:0:-1])

        # add conjunction if there are many gifts
40      if day > 1:
            todays_gifts += ', and '
        # a partridge in a pear tree!
        todays_gifts += gifts[0]

45      post = '.\n'
        return pre + todays_gifts + post

    def verses(start, end):
        return '\n'.join([verse(n) for n in range(start, end + 1)]) + '\n'
50
    def sing():
        return verses(1, 12)
```

```python
     import string, re

     def word_count(arg):
         arg = arg.lower()
         # generate pattern that matches any punctuation or whitespace character
         pattern = '[' + string.punctuation + '\s]'
         # split into non-empty words as separated by pattern
         words = [x for x in re.split(pattern, arg) if x]

         rv = {}
         for s in words:
             # initialize with 1 if s is not in rv; otherwise, increment rv[s]
             rv[s] = rv.get(s, 0) + 1
         return rv
```