BIS 420 PROGRAMMING FOR DATA SCIENCE
PRAJAKTA POHARE
CHAPTER 12 EXERCISE 12.2
ILLINOIS STATE UNIVERSITY

In this example, ties are broken by comparing words, so words with the same length appear in reverse alphabetical order. For other applications you might want to break ties at random. Modify this example so that words with the same length appear in random order. Hint: see the random function in the random module. Solution: http: // thinkpython. com/ code/unstable_ sort. py .

```python
from __future__ import print_function, division

import random

def sort_by_length(words):
    t = []
    for word in words:
        t.append((len(word), word))

    t.sort(reverse=True)

    res = []
    for length, word in t:
        res.append(word)
    return res

def sort_by_length_random(words):
    t = []
    for word in words:
        t.append((len(word), random.random(), word))

    t.sort(reverse=True)

    res = []
    for length, _, word in t:
        res.append(word)
    return res

if __name__ == '__main__':
    words = ['banana', 'pie', 'Washington', 'book', 'apple', 'pear', 'a', 'zoo', 'car']
```

```
t = sort_by_length_random(words)
for x in t:
    print(x)
```

```python
1   from __future__ import print_function, division
2
3   import random
4
5
6   def sort_by_length(words):
7       t = []
8       for word in words:
9           t.append((len(word), word))
10
11      t.sort(reverse=True)
12
13      res = []
14      for length, word in t:
15          res.append(word)
16      return res
17
18
19  def sort_by_length_random(words):
20      t = []
21      for word in words:
22          t.append((len(word), random.random(), word))
23
24      t.sort(reverse=True)
25
26      res = []
27      for length, _, word in t:
28          res.append(word)
29      return res
30
31
32  if __name__ == '__main__':
33      words = ['banana', 'pie', 'Washington', 'book', 'apple', 'pear', 'a', 'zoo', 'car']
34
35      t = sort_by_length_random(words)
36      for x in t:
37          print(x)
```