```
1
2  # Copyright 2010 Google Inc.
3  # Licensed under the Apache License, Version 2.0
4  # http://www.apache.org/licenses/LICENSE-2.0
5
6  """Words exercise
7
8  The main() below is already defined and complete. It calls print_words()
9  and print_top() functions which you write.
10
11 1. For the --count flag, implement a print_words(filename) function that counts
12 how often each word appears in the text and prints:
13 word1 count1
14 word2 count2
15 ...
16
17 Print the above list in order sorted by word (python will sort punctuation to
18 come before letters -- that's fine). Store all the words as lowercase,
19 so 'The' and 'the' count as the same word.
20
21 2. For the --topcount flag, implement a print_top(filename) which is similar
22 to print_words() but which prints just the top 20 most common words sorted
23 so the most common word is first, then the next most common, and so on.
24
25 Use str.split() (no arguments) to split on all whitespace.
26
27 Workflow: don't build the whole program at once. Get it to an intermediate
28 milestone and print your data structure and sys.exit(0).
29 When that's working, try for the next milestone.
30
31 Optional: define a helper function to avoid code duplication inside
32 print_words() and print_top().
33
34 """
35
36 import sys
37
38 # +++your code here+++
39 # Define print_words(filename) and print_top(filename) functions.
40 # You could write a helper utility function that reads a file
41 # and builds and returns a word/count dict for it.
42 # Then print_words() and print_top() can just call the utility function.
43
44
45 def utility(filename):
46     f = open(filename, 'r')
47     text = f.read()
48     text_list = text.split()
49
50     words = {}
51     for w in text_list:
52         w = w.lower()
53         if w in words:
54             words[w] = words[w] + 1
55         else:
56             words[w] = 1
57
58     return words
59
60 def get_count(words_tuple):
61     return words_tuple[1]
62
```

```
63 def print_words(filename):
64     word_count = utility(filename)
65     words = sorted(word_count)
66 
67     for word in words:
68         print(f'{word}       \t{word_count[word]}')
69 
70 
71 def print_top(filename):
72     word_count = utility(filename)
73 
74     items = sorted(word_count.items(), key=get_count, reverse=True)
75 
76     for item in items[:20]:
77       print(f'{item[0]} -> {item[1]}')
78 
79 ###
80 
81 # This basic command line argument parsing code is provided and
82 # calls the print_words() and print_top() functions which you must define.
83 
84 
85 def main():
86     if len(sys.argv) != 3:
87         print('usage: python words.py {--count | --topcount} file')
88         sys.exit(1)
89 
90     option = sys.argv[1]
91     filename = sys.argv[2]
92     if option == '--count':
93         print_words(filename)
94     elif option == '--topcount':
95         print_top(filename)
96     else:
97         print('unknown option: ' + option)
98         sys.exit(1)
99 
100 
101 if __name__ == '__main__':
102     main()
103 
```