22C:16 Practice Problem Set 8

Morning Section: Complete before Tuesday, 4-16-2013
Evening Section: Complete before Monday, 4-15-2013

These practice problems correspond roughly to the material covered in Week 11 (4/8-4/12). The focus of this problem set is on evaluating expressions involving dictionary operations and on writing short functions involving dictionaries.

1. Suppose that \( D \) is the dictionary \{"what":22, "are":11, "you":14, "doing":5, "next":9, "Saturday?":4\}. Write down what the value of \( D \) is after each of the following Python statements. Evaluate each statement starting with the same value of the dictionary \( D \), mentioned above.

(a) \( D["what"] = D["are"] \)
(b) \( D.update({"Sunday":25, "what":5}) \)
(c) \( del D["you"] \)
(d) \( D["which"] = 19 \)
(e) \( D.reset() \)
(f) \( D["you"] = D["doing"] + D["next"] \)
(g) \( D[D["you"]] = "you" \)

2. Suppose that \( D \) is the dictionary \{"what":22, "are":11, "you":14, "doing":5, "next":9, "Saturday?":4\}. What is a possible correct output for the following code fragment?

```python
sum1 = []
sum2 = 0
for x in D.items():
    sum1 = sum1 + [x]
    sum2 = sum2 + D[x[0]]
print sum1, sum2
```

3. Suppose that \( D \) is the dictionary of 5-letter words and their “neighbors” constructed in the playGame program that we wrote in class last week. For words \( u, v \) appearing as keys in \( D \), \((u, v)\) is called an isolated pair if \( u \) has only one neighbor \( v \) and \( v \) has only one neighbor \( u \). Write a function called isolatedPairs that takes this dictionary \( D \) as a parameter and returns the list of all isolated pairs. Thus the function should return a list of elements such that each element is itself a list of size 2. Note that each isolated pair \((u, v)\) will appear in the returned list twice, once as \([u, v]\) and once as \([v, u]\).

4. Suppose that \( D \) is the dictionary of 5-letter words and their “neighbors” constructed in the playGame program that we wrote in class last week. Suppose that we want to write a function called secondNeighbors that takes a word \( w \) that appears as a key in \( D \) and returns the list of all words that can be obtained by replacing either one or two letters in \( w \). Code for this function is provide below with some parts missing. Your task is to complete this function.
def secondNeighbors(w, D):
    neighborsList = D[w]
    secondNeighborsList = neighborsList
    for neighbor in neighborsList:
        # Find the list of neighbors of the word neighbor
        L = _____________________________________________

        # For each word in list L, append it to secondNeighborsList,
        # if it is not already in it. Can be done in 3 lines of code.
        _____________________________________________
        _____________________________________________
        _____________________________________________

    return secondNeighborsList