Section Solution #6

Solution 1: Rationals and Unit Fractions

# Computes a series of decreasing unit fractions that add up
# to the provided rational number. We do so by computing the
# largest unit fraction less than or equal to the supplied
# rational number, which is 1/ceil(den/num). We then
# subtract that unit fraction from the original and repeat the
# same process on the remainder until the remainder is 0.

```python
def unitFractionSum(r):
    """
    Constructs a list of distinct unit fraction
    that add up to the supplied r.
    Examples:
    unitFractionSum(1/3) --> [1/3]
    unitFractionSum(2/3) --> [1/2, 1/6]
    unitFractionSum(21/23) --> [1/2, 1/3, 1/13, 1/359, 1/644046]
    """

    fractions = []
    while r > 0:
        closest = Rational(1, ceil(r.getDenominator()/r.getNumerator()))
        fractions.append(closest)
        r = r - closest
    return fractions
```

Solution 2: Reading data structures from files

```python
class PresidentialWordCloud:
    """
    Defines a class capable of storing information about all presidential
    speeches and the most prominent words in each of them.
    """

    def __init__(self, filename):
        """
        Initializes the PresidentialWordCloud using the information
        stored within the file identified by the supplied name.
        """
        self._speeches = {}
        self._speechTags = {}
        scanner = TokenScanner(). # declare one scanner, configure to skip spaces
        scanner.ignoreWhitespace()
        with open(filename) as infile:
            while True:
                line = infile.readline()
                if line == "": break # "" returned only when EOF encountered
                title = line.strip() # strip away trailing newline
data = infile.readline().strip()
words = []
sizes = {}
```
while True:
    tag = infile.readline().strip()
    if tag == "": break.   # "" marks end of word-color-size list
    scanner.setInput(tag)
    word = scanner.nextToken()
    color = scanner.nextToken() + scanner.nextToken() # "#" + "435812"
    size = int(scanner.nextToken())
    words.append(word)
    if size not in sizes: sizes[size] = []
    sizes[size].append((word, color))

    key = title + ":" + date  # assumes dates formatted YYYY-MM-DD
    self._speeches[key] = words
    self._speechTags[key] = sizes

def getAllWords(self, title, date):
    ""
    Returns the sorted list of all prominent words used
    in the speech identifies by the supplied title and date
    ""
    key = title + ":" + date
    if key not in self._speeches: return []
    return self._speeches[key]

def getAllTags(self, title, date, size):
    ""
    Returns the sorted list of all prominent (word, color) pairs
    that would be drawn in the supplied font size for the speech
    with the supplied title and date
    ""
    key = title + ":" + date
    if key not in self._speeches: return []
    sizes = self._speechTags[key]
    if size not in sizes: return []
    return sizes[size]