

Homework 0: Python (nothing to turn in)

CSCI 562, Fall 2011

August 25, 2011

Instructions

1. This assignment is for you to get familiarized with Python, and to help you prepare for Quiz 0 (to be held at the beginning of class on Tuesday, August 30).
2. This assignment is optional, and you don't need to submit anything for it.
3. Solutions will be posted on the course website before the quiz.

Problem 1: Word Frequency

Write a program that takes some text as input, and, for each unique word (NLP people say, each word *type*), outputs a line containing the number of occurrences of the word, followed by the word itself. The lines should be sorted by decreasing frequency. For example:

Sample Input

```
Beautiful is better than ugly.  
Explicit is better than implicit.  
Simple is better than complex.  
Complex is better than complicated.  
Flat is better than nested.  
Sparse is better than dense.
```

Correct Output

```
6 than  
6 is  
6 better  
1 ugly.  
1 nested.  
1 implicit.  
1 dense.  
1 complicated.  
1 complex.  
1 Sparse  
1 Simple  
1 Flat  
1 Explicit  
1 Complex  
1 Beautiful
```

Problem 2: Permutations

Write a program which takes in a pair of positive integers on the command-line, n and m ($0 < m \leq n$), and outputs all the possible permutations of m elements from the set $\{1, 2, \dots, n\}$. The output should have one permutation per line and should be sorted. For example, for the input `3 2`, the correct output is:

```
1 2
1 3
2 1
2 3
3 1
3 2
```

The sorting should be lexicographic, so that `1 2` precedes `2 1`, as above; numbers should be compared as numbers, so that `9` precedes `10`.

Note To get n and m from the command line, use:

```
import sys
n, m = int(sys.argv[1]), int(sys.argv[2])
```

Problem 3: Merge Sort

Implement merge sort in Python. Your program should read in some lines of text and output the lines in sorted order. Please don't use the built-in `sort` method or `sorted` function.

Note Merge sort is a divide-and-conquer algorithm. The base case is a list of length 1, which is trivially sorted. For a list of length greater than 1, split the list (roughly) in half, recursively merge-sort the two halves, and then merge the two sorted halves into a single sorted list.