Map Reduce Advanced - Count number of friends

Mappers and Reducers

Here's a quick but comprehensive introduction to the idea of splitting tasks into a MapReduce model. The four important functions involved are:

```
Map (the mapper function)
EmitIntermediate(the intermediate key,value pairs emitted by the mapper functions)
Reduce (the reducer function)
Emit (the final output, after summarization from the Reduce functions)
```

We provide you with a single system, single thread version of a basic MapReduce implementation.

Task

Joins are

The input is a number of lines with pairs of name of friends, in the form:

```
[Friend1] [Friend2]
```

The required output is to print the number of friends of each person, in the format shown. The code for the MapReduce class, parts related to IO etc. has already been provided. However, the mapper and reducer functions are incomplete. Your task is to fill up the mapper and reducer functions appropriately, such that the program works, and outputs the list of number of friends of each person , in lexicographical order.

Also, this program outputs certain information to the error stream. This information has been logged to help beginners gain a better understanding of the the intermediate steps in a map-reduce process.

Languages Supported

Currently, we provide the base code in Python.

Input Format

A list of single space separated pairs of friend names. We have already written the input handling code to read in this data.

Output Format

Again, the output handling part has already been provided in the template code. The Key contains [Person name] and the value contains the number of friends, sorted in lexicographical order. The entities in this list, will naturally be confined to only those people provided in the input data.

Sample Input

```
Joe Sue
Sue Phi
Phi Joe
Phi Alice
```

Sample Output

```
{"key":"Alice","value":"1"}
{"key":"Joe","value":"2"}
{"key":"Phi","value":"3"}
{"key":"Sue","value":"2"}
```

Explanation

We have computed the number of friends for each person via the Mapper and Reducer functions.