Um conjunto de comentários sobre os produtos da sua empresa foi disponibilizado (textos em inglês). Todos eles são comentários positivos sobres os produtos. O seu gerente solicitou que você prepare uma solução que possa atender os seguinte requisitos:

I. Armazenar os textos e extrair 15 palavras chaves que são mais citadas pelos clientes;
II. Sua companhia possui 4 milhões de clientes e há uma interação com estes clientes pelo menos uma vez por mês;
III. Para cada mês deve haver uma atualização nas palavras ;
IV. Em média um cliente interage com a companhia 1 vez por mês; Cada interação gera um arquivo texto a partir do sistema fonte de informação que armazena este dado;
V. Algumas palavras deve ser descartadas: 'by', 'above', 'all', 'none', 'nobody'.....

Sua missão é fazer uma prova de conceito para o seu gerente. Ele ouviu falar de processamento paralelo e leu um artigo na '*Computer magazine*' falando sobre o movimento NoSQL: quer entender no cenário da empresa como isto poderia funcionar. Ele já solicitou um extração de 1000 arquivos sobre os comentário dos clientes. Em uma reunião foi definido que você deve entregar um exemplo funcionando amanhã

Você deve então:

a- Usar uma implementação simples e rápida;
b- Permitir que isto seja executada em 50 máquinas na empresa;
c- No final gerar um arquivo com a contagem das palavras;

Vamos começar com a implementação:

1. Faça a instalação do python 2.7;
2. Crie o diretório exerc\ ;
3. Crie o diretório exerc\textos;
4. Copie o arquivo zipado(2.1 - textos.zip) para o diretório criado no passo anterior;
5. Copie o arquivo python do mincemeat para o diretório exerc\
   https://github.com/michaelfairley/mincemeatpy
6. Crie um arquivo texto e digite a primeira parte do código – Importação dos arquivos:

```
import mincemeat
import glob
import csv

text_files = glob.glob('Exerc\\textos\\*')


def file_contents(file_name):
    f = open(file_name)
    try:
        return f.read()
    finally:
        f.close()

source = dict((file_name, file_contents(file_name))for file_name in text_files)
```

   Estas parte faz a importação dos dados e gera um objeto '*source*' que é do tipo dicionário: nome do arquivo e conteúdo do arquivo;

7. Faça a implementação do método *map*:

```
def mapfn(k, v):
        print 'map ' + k
        from stopwords import allStopWords
        for line in v.splitlines():
                for word in line.split():
                        if (word not in allStopWords ):
                                yield word, 1
```

Neste caso esta sendo utilizada um implementação rápida e simples;

8. Faça a implementação do método *reduce:*

```
def reducefn(k, v):
    print 'reduce ' + k
    return sum(v)
```

9. Utize agora o *mincemeat* ele vai simular o papel da DFS e do '*name mode*':

```
s = mincemeat.Server()

# A fonte de dados pode ser qualquer objeto do tipo dicionário
s.datasource = source
s.mapfn = mapfn
s.reducefn = reducefn

results = s.run_server(password="changeme")

w = csv.writer(open("Exerc\\RESULT.csv", "w"))
for k, v in results.items():
    w.writerow([k, v])
```

10. Salve o nome do arquivo como exerc21.py;
11. Vamos fazer a execução paralela deste código:
    a- server: python exerc21.py
    b- Crie dois ou três clientes com o comando:
        python mincemeat.py -p changeme localhost
    Para executar remoto,faça a instalação do python e mincemeat conforme passos 1 e 4, respectivamente;
12. Veja o arquivo gerado