1. Compute the edit distance between “Paris” and “alice”

```
EDITDISTANCE(s1, s2)
 1  int m[i, j] = 0
 2  for i ← 1 to |s1|
 3  do m[i, 0] = i
 4  for j ← 1 to |s2|
 5  do m[0, j] = j
 6  for i ← 1 to |s1|
 7  do for j ← 1 to |s2|
 8     do m[i, j] = min{m[i − 1, j − 1] + if (s1[i] = s2[j]) then 0 else 1fi,
 9                      m[i − 1, j] + 1,
10                      m[i, j − 1] + 1}
11  return m[|s1|, |s2|]
```

SOLUTION.

		a	l	i	c	e
	0	1 1	2 2	3 3	4 4	5 5
p	1 / 1	1 2 / 2 1	2 3 / 2 2	3 4 / 3 3	4 5 / 4 4	5 6 / 5 5
a	2 / 2	1 2 / 3 1	2 3 / 2 2	3 4 / 3 3	4 5 / 4 4	5 6 / 5 5
r	3 / 3	3 2 / 4 2	2 3 / 3 2	3 4 / 3 3	4 5 / 4 4	5 6 / 5 5
i	4 / 4	4 3 / 5 3	3 3 / 4 3	2 4 / 4 2	4 5 / 3 3	5 6 / 4 4
s	5 / 5	5 4 / 6 4	4 4 / 5 4	4 3 / 5 3	3 4 / 4 3	4 5 / 4 4

2. Write down the entries in permutation index dictionary that are generated by the term “mama”

SOLUTION.

mama$,ama$m,mama,amam,$mama.

3. Apply map reduce to problem of counting how often each term occus in a set of files. Speify map and redeuce operations for this task. Write down an example along the lines

4.

If we need $n \log_2 n$ comparisons (where n is the number of termID-docID pairs) and 2 disk seeks for each comparison, how much time would index construction for Reuters-RCV1 take if we used disk instead of memory for storage and an unoptimized sorting algorithm (i.e., not an external sorting algorithm)? Use the system parameters in Table 4.1.

SOLUTION.

4.1 An unoptimized sorting algorithm would be to have all postings stored on disk and transfer them back and forth from disk to memory to make comparisons.

A trivial Index construction task would consist of these 2 steps:

1. Parsing the documents and creating the postings(n) 2. Sorting the postings

Step 1 would take O(n) time

Step 2 takes $2*(n*\log_2 n)$*disk seek time. For RCV-1 corpus, n=100 million=10^8

In this case, Step 2 dominates the total time by a large factor.

Total time=$2*(10^8*\log_2 10^8)*5*10^{-5} s$ =

5. Can the tf-idf weight of a term in a document exceed 1?

$$\text{idf}_t = \log \frac{N}{\text{df}_t}.$$

Yes

6.

One measure of the similarity of two vectors is the *Euclidean distance* (or L_2 distance) between them:

$$|\vec{x} - \vec{y}| = \sqrt{\sum_{i=1}^{M}(x_i - y_i)^2}$$

Given a query q and documents $d_1, d_2, \ldots$, we may rank the documents d_i in order of increasing Euclidean distance from q. Show that if q and the d_i are all normalized to unit vectors, then the rank ordering produced by Euclidean distance is identical to that produced by cosine similarities.

	query					document			
word	tf	wf	df	idf	q_i = wf-idf	tf	wf	d_i = normalized wf	$q_i \cdot d_i$
digital			10,000						
video			100,000						
cameras			50,000						

► **Table 4** Cosine computation for Exercise 6.19.

SOLUTION.
$\sum(q_i - w_i)^2 = \sum q_i^2 - 2\sum q_i w_i + \sum w_i^2 = 2(1 - \sum q_i w_i)$
Thus: $\sum(q_i - v_i)^2 < \sum(q_i - w_i)^2 \Leftrightarrow 2(1 - \sum q_i v_i) < 2(1 - \sum q_i w_i) \Leftrightarrow \sum q_i v_i > \sum q_i w_i$

Exercise 0.91