**Homework #3: algorithm design and psuedo code**

**The main goal** of this homework is to apply the problem solving process to address two practical issues in the real world.  You are required to describe your algorithms in pseudo code.  You are not required  to implement your algorithms.

**Problem description:**

**Problem 1.**  Given an article such as [this one at nytimes.com](https://www.nytimes.com/2014/11/11/books/liu-cixins-the-three-body-problem-is-published-in-us.html), design an algorithm to find the top 150 most frequently co-occurring word-pairs in this article. Two words are said to co-occur if they appear in the same sentence. For example, the last sentence in this article “It’s really a milestone in Chinese science fiction.” contain the following word pairs:

('It's', 'really')

('It's', 'a')

('It's', 'milestone')

('It's', 'in')

('It's', 'Chinese')

('It's', 'science')

('It's', 'fiction')

('really', 'a')

('really', 'milestone')

('really', 'in')

('really', 'Chinese')

('really', 'science')

('really', 'fiction')

('a', 'milestone')

('a', 'in')

('a', 'Chinese')

('a', 'science')

('a', 'fiction')

('milestone', 'in')

('milestone', 'Chinese')

('milestone', 'science')

('milestone', 'fiction')

('in', 'Chinese')

('in', 'science')

('in', 'fiction')

('Chinese', 'science')

('Chinese', 'fiction')

('science', 'fiction')

* you can assume you have access to a subroutine, sentenceSplitter(article), that can accurately segment an article into separate sentences and return these sentences in an array-like structure.
* You can also assume that you have access to another routine tokenizer( sentence), that can accurately identify the individual words contained in the input sentence and return these words in another array-like data structure.

 Please describe your algorithm unambiguously using pseudo code with necessary comments in English.

**Problem 2.**  Given

* P={p\_1, p\_2, ..., p\_m}, the set of products available at a local grocery store;
* D={T\_1, T\_2, ..., T\_N}, the set of transactions that were completed at this store in the past year, where each T\_i (i in [1,N]) is a subset of P;
* min\_freq, a user-specified parameter (e.g., 5%).

A product is said to be frequent if it occurs in at least min\_freq of the transactions in D. By the same token, a pair of products is said to be frequent if it occurs in at least min\_freq of the transactions in D. Also, it's straightforward to prove the following claim: ***if a product is not frequent, none of the pairs that involve this product will be frequent***. Use this claim to design an algorithm to find all the frequent products and frequent product pairs in D. Represent your algorithm in pseudo code.

**Submission instruction:** Submit your algorithms in one plain text file, MS Word document or a PDF file. A PDF file is highly preferred.