COMS 3101 - Fall 2013
Perl Homework 1

- Due by start of class (Monday 4pm).
- See submission instructions.

1. Open a blank file and call it "hw1.pl". You will use this script throughout the assignment. Each problem solution consists of one or several Perl statements. You should add these statements to your script. Don't forget to include the interpreter directive.

2. Build a Hash:

a) Given a hard-coded array, build a hash with each element of the array as the key and the number of characters in the array element as the value.
For example, given @arr = ("Boston", "Texas") build %hash such that $hash{"Boston"} = 6 and $hash{"Texas"} = 5.
This should work for an arbitrary sized array.

b) Print each element of the array followed by the number of characters. Each pair should be displayed on a different line.

3. Arrays and Loops:

a) A polynomial is an expression of the form
a + bx + cx^2 + dx^3 ... (^ indicates exponentiation).
Assume a polynomial is represented by a hard-coded array named @coeffs which contains coefficients of increasing degree terms.

Example:
(12,5,2) ==> 12 + 5x + 2x^2
(3,0,4) ==> 3 + 4x^2
(0,0,3,6) ==> 3x^2 + 6x^3

Given @coeffs and $x (also hard-coded), evaluate the polynomial, and assign the result to the variable $res.

Note: in the second example above, for $x=4, $res gets the value 67.
Note: You will need a loop on @coeff

b) Print the result.

4. Hashes, Conditionals, Subroutines:

a) Implement a subroutine Word_Lookup which looks up a hash irrespective of the cAsE of letters in the key.

The hash %hash is defined globally. All keys of %hash are either all UPPERCASE or all lowercase. For any given key, only one of either UPPERCASE or lowercase version will be present as a key in the %hash. i.e. it may contain one of either "JEN" or "jen" as key, but not both. (Note that, both are valid distinct keys as far a Perl is concerned, but assume our hash will have only either one.

$key is passed as an argument to Word_Lookup.
Irrespective of the cAsE of any letter of $key, Word_Lookup should return the value in the %hash. It should return the string "NIL" if the $key is not present in either UPPER or lower case.

For example, here is a hard-coded hash that you can use to verify your code:

```
%hash = ( "boston" => 4,
          "DALLAS" => 5,
          "dayton => 3.4,
          "FRESNO" => 6,
          "newark" => 3 );
```

b) Include some test-code to verify that your subroutine works. For example, make some calls to the Word_Lookup and print the returned values.

5. Input/Output, hashes, Sort:

a) The file article.txt contains a recent article.

Create a file wordcounts.txt containing the list of unique words and corresponding frequencies in the decreasing order of frequencies.
Also assign the number of unique words to the variable $count.

Assume a space character splits a line to words.
For each line read from the article.txt, you can simply split the line using space as the delimiter to get the list of words. In other words, do not worry about punctuations etc.

For example, "He said, it's OK" would be 4 words
("He", "said,", "it's", "OK)

Each line in wordcounts.txt should contain the word and the frequency separated by space, for example:

the 55
and 40
she 10

b) Print the number of unique words you found, the word with the highest frequency (top of the list) and its corresponding frequency.