School of Computing and Information Systems
The University of Melbourne
COMP90049
Knowledge Technologies (Semester 1, 2018)
Workshop exercises: Week 3

1. Following on from last week, write a **regular expression** which will:

   - Of course, there are numerous ways of writing these. They can also be made far more complicated by dealing with stranger and stranger edge-cases:

   (a) Match a string according to whether it contains a price (like $20 or $0.99, but not 11.30 or 0$n1a).

   - `/\b\$(0|[1-9][0-9]*)(\.\d{1,2})?\b/`

   (b) Match a number in scientific E notation (e.g. `2.00600e+003`)

   - `/^\d(\.\d+)?[eE](\+|-)?\d+$/`

   (c) Remove all HTML comments from an HTML document (defined as a string)

   - The HTML standard is a bit of a moving target. From `https://blog.ostermiller.org/find-comments-html`:
     `s/\<![ \r\n\t]*(--([^\-]|[\r\n]|-[^\-])*--[ \r\n\t]*)\>//g`

   (d) Validate an email address (i.e. the string will match if it is an email address, and will mismatch otherwise)

   - Note that an email address can be a tricky thing to define. See `http://www.ex-parrot.com/~pdw/Mail-RFC822-Address.html` for a (long!) Perl regular expression that validates acccording to the RFC 822 grammar (RFC 5322 is too hard for regexes). See a relevant discussion at `http://stackoverflow.com/questions/201323/using-a-regular-expression-to-validate-an-email-address`. A (flawed) example solution from that thread:
     `/^(\w|-)+(\.(\w|-)+)*@(\w|-)+(\.(\w|-)+)*(\.[a-z]{2,4})$/`

&

Suppose that we have observed the token `lended`, and we have a dictionary as follows:

```
addendum
blenders
commodity
deaden
end
leader
leant
lent
lemonade
pleading
```

2. Which, if any, of the above dictionary entries be returned using a Neighbourhood Search with a neighbourhood of 1? 2? 3?

   - There aren't any items in the dictionary requiring only a single change from `lended`.
   - With a neighbourhood size of 2, there is a dictionary entry:
     - `leader`, by Replacing the `n` with `a`, and the second `d` with `r`
   - Along with the above, the following are also within a neighbourhood of 3:

- `blenders`, by Inserting the `b`, Replacing the second `d` with `r`, and Inserting the `s`
- `deaden` (three Replaces)
- `end` (three Deletions)
- `lent` (one Replace and two Deletions)

3. With respect to the input string `lended` and the dictionary entry `deaden`, calculate the following:

   (a) the Global Edit Distance, using the parameter $[m, i, d, r] = [+1, -1, -1, -1]$

(a)	ε		`l`		`e`		`n`		`d`		`e`		`d`
ε	0	←	-1	←	-2	←	-3	←	-4	←	-5	←	-6
	↑	↖		↖		↖		↖				↖	
`d`	-1		-1	←	-2	←	-3		-2	←	-3	←	-4
	↑	↖	↑	↖						↖			
`e`	-2		-2		0	←	-1	←	-2		-1	←	-2
	↑	↖	↑		↑	↖		↖			↑	↖	
`a`	-3		-3		-1		-1	←	-2		-2		-2
	↑	↖	↑		↑	↖	↑	↖				↖	
`d`	-4		-4		-2		-2		0	←	-1		-1
	↑	↖	↑	↖	↑	↖	↑		↑	↖			
`e`	-5		-5		-3		-3		-1		1	←	0
	↑	↖	↑		↑	↖			↑		↑	↖	
`n`	-6		-6		-4		-2		-2		0		0

   - From the first table overleaf, we can observe that the Global Edit Distance is 0, corresponding to the following sequence of operations: Replace, Match, Replace, Match, Match, Replace, which I will abbreviate as `rmrmmr`. (You can follow along with the highlighted back-pointers.)

4. Find the best approximate match (or matches, if there are ties) in the dictionary for the string `lended`, based on the following methods; consider different parameters where necessary:

   (a) the Global Edit Distance

   - Using the above scoring parameter, the closest matches are `blenders` (+2) and `leader` (+2)
   - You might like to try some other parameter setting(s), to see if they give different results.

   (b) (continued next week)