COMP30018/COMP90049
Knowledge Technologies (Semester 2, 2015)
Workshop solutions: Week 2

1. Observe from the lecture slides that we have definitions (of sorts) for *data* and *information*, but no explicit definition of *knowledge* is given. Reread the lecture slides and revise the definitions of the two terms above, and choose salient points of the lecture to define *knowledge* (perhaps in terms of the other two concepts).

   - Data: measurements (bit patterns for computers)
   - Information: processed data; patterns that are satisfied for given data
   - Knowledge: information interpretted with respect to a user's context to extend human understanding in a given area (where we have data)

2. Revise the definition of "knowledge tasks", with respect to "concrete tasks". Consider the following, and decide into which category you believe they fall, referring to the definition you have decided upon above.

   - Concrete tasks: mechanically processing data to an unambiguous solution; limited contribution to human understanding
   - Knowledge tasks: data is unreliable or the outcome is ill-defined (usually both); computers mediate between the user and the data, where context (for the user) is critical; enhance human understanding

   (a) Multiplying two floating-point numbers in base 16
      - Concrete: well-defined solution

   (b) Playing a competitive game of naughts-and-crosses
      - Concrete: solution space completely defined

   (c) Playing a competitive game of go
      - Knowledge: goal is clear, but mechanism for arriving at the outcome is not; heuristics and best-guess strategies (note that go is becoming "solved" with advances in processing power/better definition of positional strategy)

   (d) Playing a competitive game of tennis
      - Not exactly clear what this means in a computational context: let's say "coming up with winning strategies for a tennis-playing robot"
      - Knowledge: goal is clear, but mechanism for arriving at the outcome is not; heuristics and best-guess strategies

   (e) Calculating the trajectory of a thrown book
      - Concrete: well-defined solution

   (f) Selecting appropriate counter-measures after someone has thrown a book at you
      - Concrete: generally well-defined "solution", although many different measures are possible, and choosing the best one may depend on user's context (so possibly could be construed as a knowledge task)

   (g) Selecting a book that a given person will enjoy reading
      - Knowledge: context of user is critical; no single "correct" solution

   (h) Translating a program written in C into Java
      - Concrete: generally well-defined solution; depends somewhat on how closely you want to mimic the C behaviour; occasionally many solutions

   (i) Translating a document written in Japanese in English
      - Knowledge: context of user (both original writer and output consumer) is critical

3. What are “structured data” and “unstructured data”? Give an example of each, and indicate how you would handle each in a computational setting.
   What about “semi-structured data”? Are any of the examples you gave above actually instances of semi-structured data? In what ways is it easier or more challenging to handle semi-structured data?

   - Structured data: conforms to a schema, e.g. database
   - Unstructured data: data without regular decomposable structure, e.g. plain text
   - Semi-structured data: data which corresponds in part to a schema, but irregular or incomplete or rapidly changing; some important information is unavailable even with the schema
   - In practice, all data is semi-structured: without a schema, a bit pattern is uninterpretable; without context, a measurement is just a number, etc. Even “structured” data typically contains inaccessible information: consider a database with a `name` field. Name of what? A person (given name? surname? both?)? An organisation? A city? A street? Need context to disambiguate.

4. What is “knowledge discovery”? Describe a process through which we might be able to answer the question “Where shall we go for dinner tonight?” using Google (`http://www.google.com`) as a resource. Identify the various instances of “data”, “information”, and “knowledge” encountered in your system. Why are the following poor answers (and why might they be proposed by your system)?

   - Conceptually, we want to construct a query out of good keywords, parse the results returned by Google, find restaurants which meet our criteria (somehow), and choose a single restaurant which best meets our criteria out of the (many) candidates.

   (a) “Jeremy Nicholson’s house”
      - No way!

   (b) “Djon’s Steak and Lobster House”
      - Melbourne, FL, USA

   (c) “maze”
      - Closed August 2011

5. Revise the following regular expression operators:

```
( ) [ ] { } . * + ? ^ $ | \
```

   For each of the following, give a couple of examples of strings which the regular expression would match. Describe (colloquially, in a manner that a non-technical person would understand) the set of string that the pattern is designed to match.

   (a) `/[a-zA-Z]+/`
      - One or more letters of the (English) Latin alphabet, e.g. `hello`, `world`, `123username` (note that the `username` part matches even if the `123` does not)

   (b) `/^[A-Za-z][a-z]*$/`
      - A string comprised of one or more letters of the Latin alphabet, where the first is optionally uppercase (a “word”), e.g. `Alphabet`

   (c) `/p[aeiou]{,2}t/`
      - Up to two (English) vowels between the letters `p` and `t`, e.g. `pout`, `carpet`, `pt`

   (d) `/\s(\w+)\s\1/`
      - A whitespace character followed by one or more alphanumeric (“word”) characters (memoised as the first group), followed by another (possibly different) whitespace character, followed by a copy of the first “word”, e.g. `␣test\ttesting`

(e) `/^(\+[0-9][0-9]|0)[0-9] ([0-9][ -]?){8}$/`

- An Australian (8 digit) telephone number with optional separators and either region or country code, e.g. `+613␣8344␣1234`

6. Write a regular expression to solve the following problems:

(a) Match a price

- Depends on how stringently we want to handle cases like $001.230 — one possible solution: `/\$(0|[1-9][0-9]*)(\.\d{1,2})?/`

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

- Again, various possibilities (consider 0.00e-003): `/\d(\.\d+)?[eE](\+|-)?\d+/`

(c) Remove HTML comments

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

(d) Validate an email address

- 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})$/`