MA410 Prolog Practical 7/8 - Machine Translation & NLP

1 Basic Translation

(a) Download `lab7p1.pl`, `lab7fns.pl`, `eng.pl`, `fra.pl`, `gae.pl` and `pol.pl`.

(b) Load `lab7p1.pl` into Prolog.

(c) Try out the following:

```
?- listing(se).
?- listing(eng:se).
?- eng:se(GS, S, []).
?- eng:se(GS, [the, dog, eats, the, food], []).
?- eng:se(sentence(nphrase(det(the), noun(dog)),
                   vphrase(verb(eats))), S, []).
```

(d) Try with some of the other languages.

(e) Download & edit `lab7fns.pl`. Define the predicate:

```
trans(LANG_IN, LANG_OUT, ID, SNT, TRANS) :-
    FROM=..[ID,GS,SNT,[]], TO=..[ID,GS,TRANS,[]],
    call(LANG_IN:FROM), call(LANG_OUT:TO).
```

(f) Try out some translations, e.g.

```
?- translate(eng,fra,se,[the,dog,eats,the,food],T).
?- translate(pol,gae,se,[pies,je,jedzenie],T).
```

(g) Include `sent2lst.pl` from lab 6 in `lab7fns.pl`.

Note: make sure to test using examples after each of the steps below.

(h) Create a predicate `list_to_sentence(L)` that takes in a list `L` and writes the elements of the list to the screen separated by spaces. *(Hint: Refer to predicate `write_list` in `ltos.pl` from lab 2 as a guide.)*

(i) Using `read_in` & `list_to_sentence`, create predicate `rtrans(LANG_IN, LANG_OUT)`

that asks for a sentence in language `LANG_IN` and writes the translation in `LANG_OUT` to the screen.

(j) Create a similar predicate `rtrans2` that writes to the screen (with blanks filled in):

```
The translation from .. to .. of ... is ...
```

(k) Create code to allow the user enter into prolog:

```
?- translate from LANG_IN to LANG_OUT.
```

(Recall use of the `op` command)

(l) Alter the modules with some additional words. Use the internet to translate words if need be.

Task: Create your own language module (that uses romanised script). If you're not familiar with some other language, then make up your own.

2 Translating Questions

(a) Draw out a basic parse tree of the structure:

- question = interrogative + verb + noun phrase
- noun phrase = object or (object + prepn + *proper* noun)
- object = *uncountable/plural* noun or (det'r + *countable* noun)

(b) Restart Prolog and download `l7p2eng.pl`, `l7p2esp.pl` & `lab7p2.pl`.

(c) Suppose verb and noun phrase are classified by number(*singular* or *plural*) and that nouns are classified by type(*countable*, *uncountable* or *proper*).

Edit `l7p2eng.pl` and add the following (noting two-letter abbreviations, e.g. `ne`=noun phrase etc.):

i. Definitions for `ne` & `ot` using structure above.
ii. Interrogative `what`.
iii. Verbs 'to teach', 'to be' in singular and plural.
iv. determiners `a`, `some` & `the`.
v. Proper nouns `alberto`, `jose` and `bella`.
vi. Countable nouns `dog`, `brother` and `sister` in `singular` and `plural`.

(d) Alter `l7p2esp.pl` so it contains the same question structure & breakdown as `l7p2eng.pl`.

(e) Given the following facts for spanish:

- words "spanish" and "english" are masculine.
- mathematics is feminine singular.
- 'the' is translated as:

	masculine	feminine
singular	el	la
plural	los	las

i. Add classification gender (*male* or *female*) for determiners and nouns.
ii. Add determiner "`the`".

(f) Add in the same names of people as in `l7q2eng.pl`.

(g) Load `lab7p2.pl` into prolog. Try out the predicate `to_fn(X, L)`, e.g. `?- to_fn(X, [a,b,c]).`

(h) Use `to_fn`, `read_in` with family tree KB from Lab 2 to create predicates `esp_sisters`/`esp_brothers`, which allow a user to enter in spanish equivalents of "who are the sisters/brothers of Y".

(i) Try out the examples:

- `quien son las hermanas de bella.`
- `quien son los hermanos de carl.`