MA410 Artificial Intelligence Practical TRIAL Test

Total marks given per question: 1. *[14 marks]* 2. *[43 marks]* 3. *[43 marks]*

Marks for individual parts are given in square brackets next to question.

All previous code may be copied and re-used.
Note that marks may be lost for inclusion of irrelevant code.
Throughtout `<y>` will indicate your initials.

Instructions

Attempt *all* three questions. Time allowed: *1* hr. *50* mins.

References allowed:

- All relevant written materials and prolog resources.
- Previous code from labs saved on computer.
- Course website and prolog resources given there.

References **NOT** allowed:

- E-mail, messenger services, forums or any online communication.
- Other on-line prolog resources apart from those given on course website.

Use of prohibited references will result in *disqualification* from test.

Steps:

1. Create directories `trial_<y>` and subdirectories `tq1`, `tq2` and `tq3`.
2. Go to the course webpage: `http://www.maths.nuigalway.ie/~barry`
3. Under the heading TRIAL LAB TEST (13th January 2011), enter in full name and click 'Go'.
4. Files for each question will then be shown there.
5. For each file right-click, select "save target as" and save in the appropriate directory.
6. When prompted for username and password, enter in your firstname in all lowercase as username and password as '`n`' + (last 4 digits of your id) + '`lp`' .

   E.g Joe Soap with student id 09123456 would enter username `joe` and password `n3456lp`.

Recording of Answers:

Answers should be recorded in `.pl` files, a `.inp` file and `.out` file as downloaded from website. Only `.pl` files containing your initials should be altered.

- `.pl`: contains prolog database and rules
- `.inp`: input for prolog interpreter (i.e. the commands you enter at prompt)
- `.out`: output received for commands entered.

Written answers, ideas and approaches should be recorded in the booklet provided.

Question 1: Short Questions [14 marks]

Write answers to Question 1 on this sheet and staple to answer book.
Each part within this question carries equal marks.
You can use the file `tq1_<y>.pl` for testing your code if necessary.

(i) Write down a MGU of `p(X,H,[a,b,c])` and `p([H|T], a, T)`.

(ii) Translate the following facts into prolog code using predicate `under`:
"The floor is under the table" and *"The table is under the computer"*.

(iii) Translate the rule

"if X is under Y and Y is under Z then X is under Z"

into prolog using the predicate `under`.

(iv) What problem arises with the above code?

(v) Write down adjusted version of code so that the rule would work properly.

(vi) Write down the output of the query: `?- under(X,Y).`

(vii) Using `under`, write down a corresponding predicate `over`.

Question 2: Family Tree [43 marks]

I. (a) [*3 marks*] Using definitions in `tq2fam.pl`, add the following predicates to `tq2_<y>.pl`:
i. `grandchild` ii. `ancestor` iii. `descendant`.

(b) [*5 marks*] Use `paternal_sibling` & `maternal_sibling` to create predicates:

i. `sibling(X,Y)` : `X` is both a paternal and maternal sibling of `Y`.
ii. `half_sibling(X,Y)` : `X` is either a paternal or maternal sibling of `Y` but not both.
iii. `brother(X,Y)` : `X` is a brother of `Y` with same two parents.
iv. `sister(X,Y)` : `X` is a sister of `Y` with same two parents.

(c) [*3 marks*] A set of double first cousins are cousins on both the father and mother's side (e.g. if siblings from one family married two siblings from a different family).

Using the already-defined predicates:

- `double_first_cousin`, `pat_first_cousin`, and `mat_first_cousin`

create the predicate `first_cousin(X,Y)` ensuring to avoid repeat pairs `(X,Y)`.

(d) [*6 marks*] nth cousins are children of (n-1)st cousins.

Define `cousin(1,X,Y)` to be `first_cousin(X,Y)`.

Create a predicate `cousin(N,X,Y)` where `X` is the `N`th cousin of `Y`.

(e) [*5 marks*] Enter the commands in `tq2_<y>.inp` and results in `tq2_<y>.out` for:

i. The male grandchildren of `alice`.
ii. The sets of half-siblings.
iii. The sets of double first cousins.
iv. The first cousins of the parents of `dorothy`.
v. The second cousins of `derek`.

II. (a) Alter code in `tq2_<y>.pl` to allow user to enter in the following:

i. [*3 marks*] `List out the sisters of Y.`: where List is a prolog list of `Y`'s sisters.
ii. [*5 marks*] `who are the sisters of Y.` : where the computer outputs:
- "`Y has no sisters`", if `Y` doesn't have a sister ; or else
- the sisters of `Y` in the format "`The sisters of Y are ....`"

iii. [*3 marks*] `how many sisters has Y.` : outputs the number of sisters that `Y` has.

(b) [*6 marks*] Create a generalised version of parts (a) i, ii so that sisters can be replaced by `X` where `X` can be `first_cousins` or `children`.

(c) [*4 marks*] Enter the results in `tq2_<y>.out` for the following:

i. `List out the sisters of dana.`
ii. `how many sisters has daphne.`
iii. `who are the first_cousins of diarmuid.`
iv. `who are the children of aileen.`

Question 3: Definite Clause Grammars & NLP [43 marks]

I. (a) [*4 marks*] Draw a simple parse tree diagram of the question structure breakdown:

- question = interrogative + verb phrase
- verb phrase = verb + object
- object = *uncountable/plural* noun or (determiner + *countable* noun)

(b) [*5 marks*] Given that verb/noun are classified by number(*singular* or *plural*) and noun also by type(*countable* or *uncountable*), add the following to `tq3eng_<y>.pl`:

i. missing definitions for `obj`. ii. interrogative `where`.
iii. verbs 'to teach', 'to be' for all cases[1]. iv. determiners `all` & `the`.
v. countable noun `student` in singular and plural.

(c) [*2 marks*] Record output & command for structure of question *"where is the student"*.

(d) [*3 marks*] Edit `tq3v_<y>.pl`, and using command `read_in` (in `tq3fns.pl`), create:

`is_valid_question`	outputs `true` if question is valid according to our database and `false` otherwise.

(e) [*2 marks*] Validate: i. `who studies the latin.` ii. `where are all students.`
Explain briefly steps taken and output found.

II. (a) [*2 marks*] Alter Italian module `tq3ita_<y>.pl` to contain same structure as `tq3eng_<y>.pl`.

(b) [*7 marks*] Given that, in Italian,

- "latin" is masculine, and both "mathematics" & "history" are feminine/singular.
- female version of "student" is "studentessa" and plural is "studentesse".
- "the" is translated as:

	masc	*mascsz*	*fem*
singular	il	lo	la
plural	i	gli	le

where *masczs* means a masculine word beginning with 's'+consonant or 'z'.

Add determiner "`the`" and add gender (*masc*, *mascsz* or *fem*) for determiners & nouns.

(c) [*2 marks*] Use `trans(LANG_IN, LANG_OUT, TRANS)` (as in `tq3fns.pl`) to translate
i. `who teaches latin.` ii. `che insegna lo studente.`

(d) [*3 marks*] Explain briefly how predicate `trans` works using one of the above examples.

III. Restart Prolog and edit `tq3st_<y>.pl`.

(a) [*2 marks*] Explain briefly how the predicate `to_fn(X, L)` (as in `tq3fns.pl`) works.

(b) [*9 marks*] Using predicate `to_fn`, create a predicate `ita_ques` (with explanation) that

- prompts the user so that they can enter in Italian equivalents of "who teaches X" or "who studies Y" for any subjects X and Y given in `tq3st_<y>.pl`.
- returns `nessuno` if none can be found and otherwise outputs the names.

(c) [*2 marks*] Running the query `?- ita_ques`
input the following and record the output received:
i. `che studia storia.` ii. `che insegna matematica.`

[1] assume verbs to be in third person, e.g. it teaches, they teach