ECM2418 Computer Languages and Representations
Continuous Assessment 2: Logic Programming

Dr David Wakeling

Handed out	Handed in
Thursday 12th November (T1:08)	Thursday 10th December 2020 (T1:12)

This Continuous Assessment is worth 15% of the module mark.

The solutions that you produce should all run on

```
http://www.tutorialspoint.com/execute_prolog_online.php
```

All students are reminded of the University regulations on academic honesty and plagiarism.

Question 1

On October 22nd 2020, the Exeter *Express and Echo* newspaper published its weekly *Niner* puzzle. The puzzle was as follows.

> Each number from 1 to 9 represents a different letter. Solve the clues and insert the letters in the appropriate squares to discover a word which uses all nine letters.
>
> - `452589` — gives an animal;
> - `658785` — gives a vegetable;
> - `7378719` — gives a mineral.

1	2	3	4	5	6	7	8	9

Question 1.1

Show a Prolog predicate that says an aardvark is an animal, as is an antelope, a coyote, a dingo, a donkey, an elephant, a horse, a jaguar and a kangaroo. So that, for example,

```
animal( kangaroo ).
true
```

```
animal( aardvark ).
animal( antelope ).
animal( coyote ).
animal( dingo ).
animal( donkey ).
animal( elephant ).
animal( horse ).
animal( jaguar ).
animal( kangaroo ).
```

Show another Prolog predicate that says an artichoke is a vegetable, as is a cabbage, a carrot, celery, a leek, lettuce, a marrow, an onion and a potato. So, that for example,

```
vegetable( onion ).
true
```

```
vegetable( artichoke ).
vegetable( cabbage ).
vegetable( carrot ).
vegetable( celery ).
vegetable( leek ).
vegetable( lettuce ).
vegetable( marrow ).
vegetable( onion ).
vegetable( potato ).
```

Show another Prolog predicate that says anatase is a mineral, as is basalt, cobolt, copper, galena, nickel, sodium, silver and zircon. So that, for example,

```
mineral( sodium ).
true
```

```
mineral( anatase ).
mineral( basalt ).
mineral( cobalt ).
mineral( copper ).
mineral( galena ).
mineral( nickel ).
mineral( sodium ).
mineral( silver ).
mineral( zircon ).
```

(5 marks)

Question 1.2

Show a Prolog predicate that says how animals, vegetables and minerals are spelled in English, as a list of atoms. So that, for example,

```
spell( cobalt, X ).
X = [ c, o, b, a, l, t ]
```

```
spell( A, R )
  :- atom_chars( A, R ).
```

(5 marks)

Question 1.3

Show a Prolog predicate "main" that finds a solution to the niner, so that

```
main.
syncopate
```

```
solution( X1, X2, X3, X4, X5, X6, X7, X8, X9 )
  :- animal( A ),
     spell( A, [ X4, X5, X2, X5, X8, X9 ] ),
     vegetable( V ),
     spell( V, [ X6, X5, X8, X7, X8, X5 ] ),
     mineral( M ),
     spell( M, [ X7, X3, X7, X8, X7, X1, X9 ] ).
```

```
main
  :- solution( X1, X2, X3, X4, X5, X6, X7, X8, X9 ),
     write( X1 ),
     write( X2 ),
     write( X3 ),
     write( X4 ),
     write( X5 ),
     write( X6 ),
     write( X7 ),
     write( X8 ),
     write( X9 ).
```

(10 marks)

Question 2

On September 20th 2020, *The Sunday Times* published its weekly *Teaser*. Teaser 3026, written by Graham Smithers, was as follows.

> A four-digit number with different positive digits and with the numbers represented by its last two digits a multiple of the number represented by its first two digits, is called a PAR.
>
> A pair of PARs is a PARTY if no digit is repeated and each PAR is a multiple of the missing positive digit.
>
> I wrote down a PAR and challenged Sam to use it to make a PARTY. He was successful.
>
> I then challenged Beth to use my PAR and the digits in Sam's PAR to make a different PARTY. She too was successful.
>
> What was my PAR?

An example PAR is `7826`, and an example PARTY is `( 9632, 5418 )`.

Note that in this Teaser, zero is not considered to be a positive digit.

Question 2.1

Show a Prolog predicate "`par`" that given a number (a Prolog integer), returns true if the number is a PAR, so that

```
par( 7826 ).
true


%% Note PAR predicate wrong in CA!
%% Accept both DW and ST below.
par( N )
  :- digits( N, [X1,X2,X3,X4] ),
     X1 > 0, X2 > 0, X3 > 0, X4 > 0,
     nub( [X1,X2,X3,X4], [X1,X2,X3,X4] ),
     % 0 is ( 10 * X1 + X2 ) mod ( 10 * X3 + X4 ). % DW
     0 is ( 10 * X3 + X4 ) mod ( 10 * X1 + X2 ).   % ST

digits( N, [X1,X2,X3,X4] )
  :- divMod( N,  1000, X1, N1 ),
     divMod( N1,  100, X2, N2 ),
     divMod( N2,   10, X3, X4 ).

divMod( A, B, D, M )
  :- D is A div B,
     M is A mod B.

nub( [], [] ).
nub( [X|XS], [X|RS] )
  :- \+ member( X, XS ),
     nub( XS, RS ).
nub( [X|XS], RS )
  :- member( X, XS ),
     nub( XS, RS ).
```

(20 marks)

Question 2.2

Show a Prolog predicate "`pars`" that may be used to find all PARs, so that

```
pars( PARS ).
PARS = [ 2613, 2814, 3216, ... 9814 ]
```

Note that the PARs may appear in any order that you like.

```
pars( PARS )
```

```
  :- loop( 1111, 9999, PARS ).

loop( I, LIMIT, [] )
  :- I > LIMIT.
loop( I, LIMIT, [I|RS] )
  :- par( I ),
     I1 is I + 1,
     loop( I1, LIMIT, RS ).
loop( I, LIMIT, RS )
  :- I1 is I + 1,
     loop( I1, LIMIT, RS ).
```

(10 marks)

Question 2.3

Show a Prolog predicate "party" that given two numbers (two Prolog integers), returns true if those numbers are a PARTY, so that

```
party( 9632, 5418 ).
true


party( M, N )
  :- par( M ),
     par( N ),
     digits( M, [X1,X2,X3,X4] ),
     digits( N, [Y1,Y2,Y3,Y4] ),
     intersection( [X1,X2,X3,X4], [Y1,Y2,Y3,Y4], [] ),
     difference( [1,2,3,4,5,6,7,8,9], [X1,X2,X3,X4,Y1,Y2,Y3,Y4], [W] ),
     0 is M mod W,
     0 is N mod W.

intersection( [], _, [] ).
intersection( [X|XS], YS, [X|RS] )
  :- member( X, YS ),
     intersection( XS, YS, RS ).
intersection( [X|XS], YS, RS )
  :- \+ member( X, YS ),
     intersection( XS, YS, RS ).

difference( [], _, [] ).
difference( [X|XS], YS, [X|RS] )
```

```
      :- \+ member( X, YS ),
         difference( XS, YS, RS ).
   difference( [_|XS], YS, RS )
      :- difference( XS, YS, RS ).
```

(30 marks)

Question 2.4

Show a Prolog predicate "`partys`" that may be used to find all PARTYs, so that

```
   partys( PARTYS ).
   PARTYS = [ [3618, 5427], ... [9632, 5418] ]
```

Note that the PARTYs may appear in any order you like. Note also that given these PARTYs, it is easy to find the answer to Teaser 3026.

```
   partys( PARTYS )
     :- pars( PARS ),
        loop2( PARS, PARS, PARTYS ).

   loop2( [], _, [] ).
   loop2( [P|PS], PARS, RS3 )
      :- loop3( P,  PARS, RS1 ),
         loop2( PS, PARS, RS2 ),
         append( RS1, RS2, RS3 ).

   loop3( _, [], [] ).
   loop3( P, [Q|QS], [ [P,Q] | RS ] )
     :- party( P, Q ),
        loop3( P, QS, RS ).
   loop3( P, [_|QS], RS )
     :- loop3( P, QS, RS ).
```

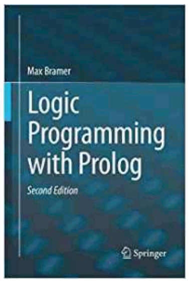
(20 marks)

Assessment Materials and Criteria

You should submit a ".zip" file of Prolog source code to the EBART system.

Your work will be assessed on the grounds of clarity (is what it computes obvious?) and correctness (is what it computes right?). Clarity is all about the appropriate use of meaningful predicate and variable names (upto half the marks for each part, with partial marks possible), and predicates built-in to the Prolog system. Correctness is all about computing the outputs from inputs, as described in each question (upto half the marks for each part, with partial marks possible).

Readings



M. Bramer, *Logic Programming with Prolog (Second Edition)*, Springer, 2013, ISBN 978-1447154860.