Artificial intelligence - Project 2
- First Order Logic Problems -

Strujan Florentina

7/12/2020

1 Problem 1 - Movies Night

Four boys are at home to watch some movies. Figure out what each boy's favorite kind of movie is.

- Joshua is in one of the ends.
- The boy wearing the Black shirt is somewhere to the left of the youngest boy.
- Joshua likes Horror movies.
- The 14-year-old boy is at the third position.
- The boy wearing the Red shirt is somewhere between the 13-year-old boy and the one who likes Action movies, in that order.
- Daniel likes Thriller movies.
- The boy who is going to eat Cookies is in one of the ends.
- The boy wearing the Black shirt is exactly to the left of the one who likes Thriller movies.
- The boy who is going to eat Crackers is exactly to the right of the boy who likes Comedy movies.
- The boy wearing the Red shirt is somewhere between the boy who is going to eat Popcorn and Nicholas, in that order.
- In one of the ends is the boy who likes Thriller movies.
- Nicholas is somewhere between Joshua and Daniel, in that order.
- At the first position is the boy wearing the Green shirt.

1.1 Input file

```
assign(report_stderr, 2).
set(ignore_option_dependencies). % GUI handles dependencies

if(Prover9). % Options for Prover9
  assign(max_seconds, 60).
end_if.

if(Mace4).   % Options for Mace4
  assign(start_size, 4).
  assign(max_seconds, 60).
end_if.

formulas(assumptions).

%Four boys are at home to watch some movies.
%Figure out what each boy's favorite kind of movie is.
%
%1. Joshua is in one of the ends.
%
%2. The boy wearing the Black shirt is somewhere to
%   the left of the youngest boy.
%
%3. Joshua likes Horror movies.
%
%4. The 14-year-old boy is at the third position.
%
%5. The boy wearing the Red shirt is somewhere between the 13-year-old
%   boy and the one who likes Action movies, in that order.
```

```
%
%6. Daniel likes Thriller movies.
%
%7. The boy who is going to eat Cookies is in one of the ends.
%
%8. The boy wearing the Black shirt is exactly to the left of the one who
%   likes Thriller movies.
%
%9. The boy who is going to eat Crackers is exactly to the right of the
%   boy who likes Comedy movies.
%
%10.The boy wearing the Red shirt is somewhere between the boy who is
%   going to eat Popcorn and Nicholas, in that order.
%
%11.In one of the ends is the boy who likes Thriller movies.
%
%12.Nicholas is somewhere between Joshua and Daniel, in that order.
%
%13.At the first position is the boy wearing the Green shirt.
%
%
% Constants: boy1, boy2, boy3, boy4
%
% Predicate: DifferentFrom(x,y): x is different from y
% Predicate: RightNeighbor(x,y): y is immediately to the right of x
% Predicate: Between(x,y,z): y is somewhere between x and z, in this order
% Predicate: SomewhereLeft(x,y): x is somewhere to the left of y
%
% Predicate: Daniel(x): the name of the boy x is Daniel
% Predicate: Joshua(x): the name of the boy x is Joshua
% Predicate: Nicholas(x): the name of the boy x is Nicholas
% Predicate: Ryan(x): the name of the boy x is Ryan
%
% Predicate: 11yo(x) : the boy x is 11 yers old
% Predicate: 12yo(x) : the boy x is 12 yers old
% Predicate: 13yo(x) : the boy x is 13 yers old
% Predicate: 14yo(x) : the boy x is 14 yers old
%
% Predicate: black(x): the boy x has a black shirt
% Predicate: blue(x): the boy x has a blue shirt
% Predicate: green(x): the boy x has a green shirt
% Predicate: red(x):the boy x has a red shirt
%
% Predicate: chips(x): the boy x eats chips
% Predicate: cookies(x): the boy x eats cookies
% Predicate: crackers(x): the boy x eats crackers
% Predicate: popcorn(x): the boy x eats popcorn
%
% PRedicate: action(x): the boy's favourite kind of movie is action
% Predicate: comedy(x): the boy's favourite kind of movie is comedy
% Predicate: thriller(x): the boy's favourite kind of movie is thriller
% Predicate: horror(x): the boy's favourite kind of movie is horror

DifferentFrom(boy1,boy2).
```

```
DifferentFrom(boy1,boy3).
DifferentFrom(boy1,boy4).
DifferentFrom(boy2,boy3).
DifferentFrom(boy2,boy4).
DifferentFrom(boy3,boy4).

DifferentFrom(x,y) -> DifferentFrom(y,x).

RightNeighbor(boy1,boy2).
RightNeighbor(boy2,boy3).
RightNeighbor(boy3,boy4).

-RightNeighbor(boy1,boy1).
-RightNeighbor(boy1,boy3).
-RightNeighbor(boy1,boy4).

-RightNeighbor(boy2,boy1).
-RightNeighbor(boy2,boy2).
-RightNeighbor(boy2,boy4).

-RightNeighbor(boy3,boy1).
-RightNeighbor(boy3,boy2).
-RightNeighbor(boy3,boy3).

-RightNeighbor(boy4,boy1).
-RightNeighbor(boy4,boy2).
-RightNeighbor(boy4,boy3).
-RightNeighbor(boy4,boy4).

Between(boy1,boy2,boy3).
Between(boy1,boy2,boy4).
Between(boy1,boy3,boy4).
Between(boy2,boy3,boy4).

-Between(boy1,boy1,boy1).
-Between(boy1,boy1,boy2).
-Between(boy1,boy1,boy3).
-Between(boy1,boy1,boy4).
-Between(boy1,boy2,boy1).
-Between(boy1,boy2,boy2).
-Between(boy1,boy3,boy1).
-Between(boy1,boy3,boy2).
-Between(boy1,boy3,boy3).
-Between(boy1,boy4,boy1).
-Between(boy1,boy4,boy2).
-Between(boy1,boy4,boy3).
-Between(boy1,boy4,boy4).

-Between(boy2,boy1,boy1).
-Between(boy2,boy1,boy2).
-Between(boy2,boy1,boy3).
-Between(boy2,boy1,boy4).
-Between(boy2,boy2,boy1).
-Between(boy2,boy2,boy2).
```

```
-Between(boy2,boy2,boy3).
-Between(boy2,boy2,boy4).
-Between(boy2,boy3,boy1).
-Between(boy2,boy3,boy2).
-Between(boy2,boy3,boy3).
-Between(boy2,boy4,boy1).
-Between(boy2,boy4,boy2).
-Between(boy2,boy4,boy3).
-Between(boy2,boy4,boy4).

-Between(boy3,boy1,boy1).
-Between(boy3,boy1,boy2).
-Between(boy3,boy1,boy3).
-Between(boy3,boy1,boy4).
-Between(boy3,boy2,boy1).
-Between(boy3,boy2,boy2).
-Between(boy3,boy2,boy3).
-Between(boy3,boy2,boy4).
-Between(boy3,boy3,boy1).
-Between(boy3,boy3,boy2).
-Between(boy3,boy3,boy3).
-Between(boy3,boy3,boy4).
-Between(boy3,boy4,boy1).
-Between(boy3,boy4,boy2).
-Between(boy3,boy4,boy3).
-Between(boy3,boy4,boy4).

-Between(boy4,boy1,boy1).
-Between(boy4,boy1,boy2).
-Between(boy4,boy1,boy3).
-Between(boy4,boy1,boy4).
-Between(boy4,boy2,boy1).
-Between(boy4,boy2,boy2).
-Between(boy4,boy2,boy3).
-Between(boy4,boy2,boy4).
-Between(boy4,boy3,boy1).
-Between(boy4,boy3,boy2).
-Between(boy4,boy3,boy3).
-Between(boy4,boy3,boy4).
-Between(boy4,boy4,boy1).
-Between(boy4,boy4,boy2).
-Between(boy4,boy4,boy3).
-Between(boy4,boy4,boy4).

SomewhereLeft(boy1,boy2).
SomewhereLeft(boy1,boy3).
SomewhereLeft(boy1,boy4).
SomewhereLeft(boy2,boy3).
SomewhereLeft(boy2,boy4).
SomewhereLeft(boy3,boy4).

-SomewhereLeft(boy1,boy1).

-SomewhereLeft(boy2,boy1).
```

```
-SomewhereLeft(boy2,boy2).

-SomewhereLeft(boy3,boy1).
-SomewhereLeft(boy3,boy2).
-SomewhereLeft(boy3,boy3).

-SomewhereLeft(boy4,boy1).
-SomewhereLeft(boy4,boy2).
-SomewhereLeft(boy4,boy3).
-SomewhereLeft(boy4,boy4).

%Each boy has a unique name

Daniel(x) | Joshua(x) | Nicholas(x) | Ryan(x).

Daniel(x) & Daniel(y) -> -DifferentFrom(x,y).
Joshua(x) & Joshua(y) -> -DifferentFrom(x,y).
Nicholas(x) & Nicholas(y) -> -DifferentFrom(x,y).
Ryan(x) & Ryan(y) -> -DifferentFrom(x,y).

%Each boy has a unique age

11yo(x) | 12yo(x) | 13yo(x) | 14yo(x).

11yo(x) & 11yo(y) -> -DifferentFrom(x,y).
12yo(x) & 12yo(y) -> -DifferentFrom(x,y).
13yo(x) & 13yo(y) -> -DifferentFrom(x,y).
14yo(x) & 14yo(y) -> -DifferentFrom(x,y).

%Each boy has a unique shirt

black(x) | blue(x) | green(x) | red(x).

black(x) & black(y) -> -DifferentFrom(x,y).
blue(x) & blue(y) -> -DifferentFrom(x,y).
green(x) & green(y) -> -DifferentFrom(x,y).
red(x) & red(y) -> -DifferentFrom(x,y).

%Each boy eats a unique snack

chips(x) | cookies(x) | crackers(x) | popcorn(x).

chips(x) & chips(y) -> -DifferentFrom(x,y).
cookies(x) & cookies(y) -> -DifferentFrom(x,y).
crackers(x) & crackers(y) -> -DifferentFrom(x,y).
popcorn(x) & popcorn(y) -> -DifferentFrom(x,y).

%Each boy has a unique favorite kind of movie

action(x) | comedy(x) | horror(x) | thriller(x).

action(x) & action(y) -> -DifferentFrom(x,y).
comedy(x) & comedy(y) -> -DifferentFrom(x,y).
horror(x) & horror(y) -> -DifferentFrom(x,y).
```

```
thriller(x) & thriller(y) -> -DifferentFrom(x,y).

%1.
Joshua(boy1) | Joshua(boy4).
%2.
SomewhereLeft(x,y) <- black(x) & 11yo(y).
%3.
Joshua(x) <-> horror(x).
%4.
14yo(boy3).
%5.
Between(x,y,z) <- 13yo(x) & red(y) & action(z).
%6.
Daniel(x) <-> thriller(x).
%7.
cookies(boy1) | cookies(boy4).
%8.
RightNeighbor(x,y) <- black(x) & thriller(y).
%9.
RightNeighbor(x,y) <- comedy(x) & crackers(y).
%10.
Between(x,y,z) <- popcorn(x) & red(y) & Nicholas(z).
%11.
thriller(boy1) | thriller(boy4).
%12.
Between(x,y,z) <- Joshua(x) & Nicholas(y) & Daniel(z).
%13.
green(boy1).

end_of_list.

formulas(goals).

end_of_list.
```

1.2 Output file

```
interpretation( 4, [number = 1,seconds = 0], [
    function(boy1, [0]),
    function(boy2, [1]),
    function(boy3, [2]),
    function(boy4, [3]),
    relation(11yo(_), [0,0,0,1]),
    relation(12yo(_), [0,1,0,0]),
    relation(13yo(_), [1,0,0,0]),
    relation(14yo(_), [0,0,1,0]),
    relation(Daniel(_), [0,0,0,1]),
    relation(Joshua(_), [1,0,0,0]),
    relation(Nicholas(_), [0,0,1,0]),
    relation(Ryan(_), [0,1,0,0]),
    relation(action(_), [0,0,1,0]),
    relation(black(_), [0,0,1,0]),
    relation(blue(_), [0,0,0,1]),
    relation(chips(_), [0,1,0,0]),
```

```
    relation(comedy(_), [0,1,0,0]),
    relation(cookies(_), [0,0,0,1]),
    relation(crackers(_), [0,0,1,0]),
    relation(green(_), [1,0,0,0]),
    relation(horror(_), [1,0,0,0]),
    relation(popcorn(_), [1,0,0,0]),
    relation(red(_), [0,1,0,0]),
    relation(thriller(_), [0,0,0,1]),
    relation(DifferentFrom(_,_), [
        0,1,1,1,
        1,0,1,1,
        1,1,0,1,
        1,1,1,0]),
    relation(RightNeighbor(_,_), [
        0,1,0,0,
        0,0,1,0,
        0,0,0,1,
        0,0,0,0]),
    relation(SomewhereLeft(_,_), [
        0,1,1,1,
        0,0,1,1,
        0,0,0,1,
        0,0,0,0]),
    relation(Between(_,_,_), [0,0,0,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0
```

Explanation: The output contains an interpretation consisting of a set of functions and relations, which actually describe a model produced by Mace4. For example, function(boy3,[2]) tells us "The boy with number 3 has been assigned the number 2 " (which is actually the third boy, because the order matters). The relation(13yo( ), [1,0,0,0]) tells us "The first boy is 13 years old."

Interpreting the output, we obtain the following results, which respect the conditions

- The first boy is boy1, who is 13 years old. His name is Joshua, he wears a green shirt, eats popcorn and love horror movies.
- The second boy is boy2, who is 12 years old. His name is Ryan, he wears a red shirt, eats chips and loves comedy movies.
- The third boy is boy3, who is 14 years old. His name is Nicholas, he wears a black shirt, eats crackers and love action movies.
- The fourth boy is boy4, who is 11 years old. His name is Daniel, he wears a blue shirt, eats cookies and love thriller movies.

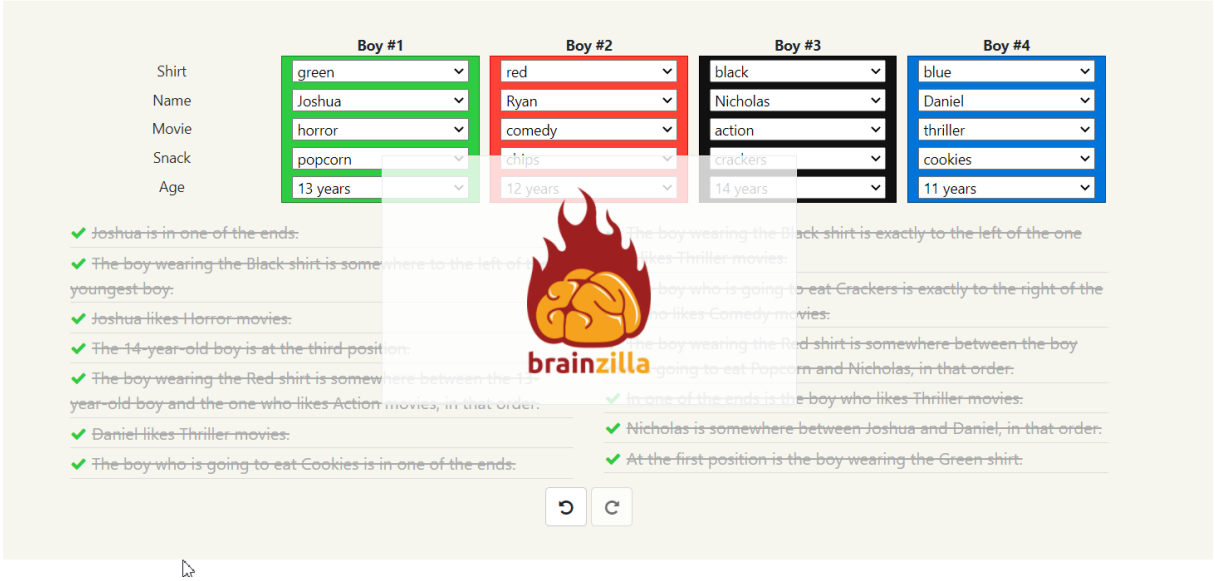

Figure 1: Correct answer

2 Problem 2 - Dracula and Friends

Various vampires lived in distinct regions of Romania over different centuries.Which plant did each of them fear the most?

- One, and only one, of the vampires had the same initial of his name and of his birthplace.
- Matei wasn't from Dobrogea. He hated onions or ivy.
- The vampire from Muntenia lived in a century without thornbush.
- 100 years after Dorian's death, another vampire rised in Bucovina, but this wasn't Bogdan.
- Octavian either lived in the XVI century or hated thornbush.
- If Bogdan hated wolfsbane, then Matei lived in Bucovina.
- The vampire from XIV century wasn't Octavian nor Bogdan.
- Villagers didn't grow thornbush against Dorian.
- Chronicles of XVII century claimed that ivy was ineffective, and that Dobrogea was free from vampires.

2.1 Input file

```
assign(report_stderr, 2).
set(ignore_option_dependencies). % GUI handles dependencies

if(Prover9). % Options for Prover9
  assign(max_seconds, 60).
end_if.

if(Mace4).   % Options for Mace4
  assign(max_seconds, 60).
end_if.

formulas(assumptions).

%Various vampires lived in distinct regions of Romania over different centuries.
%Which plant did each of them fear the most?
%
%1. One, and only one, of the vampires had the same initial of his name
% and of his birthplace.
%
%2. Matei wasn't from Dobrogea. He hated onions or ivy.
%
%3. The vampire from Muntenia lived in a century without thornbush.
%
%4. 100 years after Dorian's death, another vampire rised in Bucovina,
% but this wasn't Bogdan.
%
%5. Octavian either lived in the XVI century or hated thornbush.
%
%6. If Bogdan hated wolfsbane, then Matei lived in Bucovina.
%
%7. The vampire from XIV century wasn't Octavian nor Bogdan.
%
%8. Villagers didn't grow thornbush against Dorian.
%
%9. Chronicles of XVII century claimed that ivy was ineffective,
```

```
% and that Dobrogea was free from vampires.
%
%
% Constants: Matei, Dorian, Bogdan, Octavian
%
% Predicate: DifferentFrom(x,y): x is different from y
%
% Predicate: XIV(x) : the vampire x lived in XIVth century
% Predicate: XV(x) : the vampire x lived in XVth century
% Predicate: XVI(x) : the vampire x lived in XVIth century
% Predicate: XVII(x) : the vampire x lived in XVIIth century
%
% Predicate: Muntenia(x): the vampire x lived in Muntenia
% Predicate: Dobrogea(x): the vampire x lived in Dobrogea
% Predicate: Bucovina(x): the vampire x lived in Bucovina
% Predicate: Oltenia(x):the vampire x lived in Oltenia
%
% Predicate: ivy(x): the vampire x fears ivy
% Predicate: onion(x): the vampire x fears onion
% Predicate: thornbush(x): the vampire x fears thornbush
% Predicate: wolfsbane(x): the vampire x fears wolfsbane
%

DifferentFrom(Matei,Dorian).
DifferentFrom(Matei,Bogdan).
DifferentFrom(Matei,Octavian).
DifferentFrom(Dorian,Bogdan).
DifferentFrom(Dorian,Octavian).
DifferentFrom(Bogdan,Octavian).

DifferentFrom(x,y) -> DifferentFrom(y,x).

%Each vampire lived in a unique century

XIV(x) | XV(x) | XVI(x) | XVII(x).

XIV(x) & XIV(y) -> -DifferentFrom(x,y).
XV(x) & XV(y) -> -DifferentFrom(x,y).
XVI(x) & XVI(y) -> -DifferentFrom(x,y).
XVII(x) & XVII(y) -> -DifferentFrom(x,y).

%Each vampire lived in a unique region

Muntenia(x) | Dobrogea(x) | Bucovina(x) | Oltenia(x).

Muntenia(x) & Muntenia(y) -> -DifferentFrom(x,y).
Dobrogea(x) & Dobrogea(y) -> -DifferentFrom(x,y).
Bucovina(x) & Bucovina(y) -> -DifferentFrom(x,y).
Oltenia(x) & Oltenia(y) -> -DifferentFrom(x,y).

%Each vampire fears a unique plant

ivy(x) | onion(x) | thornbush(x) | wolfsbane(x).

```

```
ivy(x) & ivy(y) -> -DifferentFrom(x,y).
onion(x) & onion(y) -> -DifferentFrom(x,y).
thornbush(x) & thornbush(y) -> -DifferentFrom(x,y).
wolfsbane(x) & wolfsbane(y) -> -DifferentFrom(x,y).

%1.
Bucovina(Bogdan) | Dobrogea(Dorian) |
Muntenia(Matei) | Oltenia(Octavian).

Bucovina(Bogdan) -> -Dobrogea(Dorian) &
-Muntenia(Matei) & -Oltenia(Octavian).

Dobrogea(Dorian) -> -Bucovina(Bogdan) &
-Muntenia(Matei) & -Oltenia(Octavian).

Muntenia(Matei) -> -Dobrogea(Dorian) &
-Bucovina(Bogdan) & -Oltenia(Octavian).

Oltenia(Octavian) -> -Dobrogea(Dorian) &
-Muntenia(Matei) & -Bucovina(Bogdan).

%2.
-Dobrogea(Matei).
onion(Matei) | ivy(Matei).

%3.
Muntenia(x) -> -thornbush(x).

%4.
XIV(Dorian) -> (Bucovina(Matei) & XV(Matei)) | (Bucovina(Octavian) & XV(Octavian)).
XV(Dorian) -> (Bucovina(Matei) & XVI(Matei)) | (Bucovina(Octavian) & XVI(Octavian)).
XVI(Dorian) -> (Bucovina(Matei) & XVII(Matei)) | (Bucovina(Octavian) & XVII(Octavian)).
-XVII(Dorian).

%5.
XVI(Octavian) | thornbush(Octavian).

%6.
wolfsbane(Bogdan) -> Bucovina(Matei).

%7.
-XIV(Octavian).
-XIV(Bogdan).

%8.
-thornbush(Dorian).

%9.
XVII(x) -> -ivy(x) & -Dobrogea(x).

end_of_list.

formulas(goals).

```

```
end_of_list.
```

2.2 Output file

```
interpretation( 4, [number = 1,seconds = 0], [
    function(Bogdan, [0]),
    function(Dorian, [1]),
    function(Matei, [2]),
    function(Octavian, [3]),
    relation(Bucovina(_), [0,0,1,0]),
    relation(Dobrogea(_), [0,1,0,0]),
    relation(Muntenia(_), [0,0,0,1]),
    relation(Oltenia(_), [1,0,0,0]),
    relation(XIV(_), [0,1,0,0]),
    relation(XV(_), [0,0,1,0]),
    relation(XVI(_), [0,0,0,1]),
    relation(XVII(_), [1,0,0,0]),
    relation(ivy(_), [0,0,0,1]),
    relation(onion(_), [0,0,1,0]),
    relation(thornbush(_), [1,0,0,0]),
    relation(wolfsbane(_), [0,1,0,0]),
    relation(DifferentFrom(_,_), [
        0,1,1,1,
        1,0,1,1,
        1,1,0,1,
        1,1,1,0])]).
```

Explanation: The output contains an interpretation consisting of a set of functions and relations, which actually describe a model produced by Mace4. For example, function(Matei,[2]) tells us ”Matei has been assigned the number 2 ” (which is actually the third vampire). The relation(ivy( ), [0,1,0,0]) tells us "The second vampire(wich is Dorian if we look above.) fears ivy".

Interpreting the output, we obtain the following results, which respect the conditions

- Bogdan lived in Oltenia in the XVII-th century and he feared thornbush.
- Dorian lived in Dobrogea in the XIV-th century and he feared wolfbush.
- Matei lived in Bucovina in the XV-th century and he feared onion.
- Octavian lived in Muntenia in the XVI-th century and he feared ivy.

3 Problem 3 - Ancient Gods

There are four Greek gods in this logic problem. You have to associate them with their respective Roman gods, weekday and domain to answer this question: Who was the Greek god of poetry?

- Neither Aphrodite nor the god of War were called Mercury.
- Zeus was worshipped the day after Hermes and the day before Venus.
- Either Ares' domain was War, or he was known as Jupiter.
- The god of poetry was either called Aphrodite or was worshipped on Wednesday.
- Friday and Tuesday weren't good days for poetry.
- Jupiter was worshipped Thursday or Friday.
- The god of War was worshipped Tuesday or Wednesday.
- Mercury was worshipped in one of the first three days.
- The god of Love is either called Aphrodite or was worshipped on Friday

3.1 Input file

```
assign(report_stderr, 2).
set(ignore_option_dependencies). % GUI handles dependencies

if(Prover9). % Options for Prover9
  assign(max_seconds, 60).
end_if.

if(Mace4).   % Options for Mace4
  assign(max_seconds, 60).
end_if.

formulas(assumptions).

%There are four Greek gods in this logic problem.
%You have to associate them with their respective Roman gods,
%weekday and domain to answer this question:
%Who was the Greek god of poetry?
%
%
%1. Neither Aphrodite nor the god of War were called Mercury.
%2. Zeus was worshipped the day after Hermes and the day before Venus.
%3. Either Ares' domain was War, or he was known as Jupiter.
%4. The god of poetry was either called Aphrodite or was worshipped on Wednesday.
%5. Friday and Tuesday weren't good days for poetry.
%6. Jupiter was worshipped Thursday or Friday.
%7. The god of War was worshipped Tuesday or Wednesday.
%8. Mercury was worshipped in one of the first three days.
%9. The god of Love is either called Aphrodite or was worshipped on Friday
%
% Constants: Aphrodite, Ares, Hermes, Zeus
%
% Predicate: DifferentFrom(x,y): x is different from y
%
% Predicate: Tuesday(x) : Roman weekday of the Greek god x is Tuesday
```

```
% Predicate: Wednesday(x) : Roman weekday of the Greek god x is Wednesday
% Predicate: Thursday(x) : Roman weekday of the Greek god x is Thursday
% Predicate: Friday(x) : Roman weekday of the Greek god x is Friday
%
% Predicate: love(x): Roman domain of the Greek god x is love
% Predicate: poetry(x): Roman domain of the Greek god x is love
% Predicate: thunder(x): Roman domain of the Greek god x is love
% Predicate: war(x): Roman domain of the Greek god x is love
%
% Predicate: Mars(x): Roman god name of the Greek god x is Mars
% Predicate: Mercury(x): Roman god name of the Greek god x is Mercury
% Predicate: Jupiter(x): Roman god name of the Greek god x is Jupiter
% Predicate: Venus(x):  Roman god name of the Greek god x is Venus
%

DifferentFrom(Aphrodite,Ares).
DifferentFrom(Aphrodite,Hermes).
DifferentFrom(Aphrodite,Zeus).
DifferentFrom(Ares,Hermes).
DifferentFrom(Ares,Zeus).
DifferentFrom(Hermes,Zeus).

DifferentFrom(x,y) -> DifferentFrom(y,x).

%Each god has a unique Roman weekday

Tuesday(x) | Wednesday(x) | Thursday(x) | Friday(x).

Tuesday(x) & Tuesday(y) -> -DifferentFrom(x,y).
Wednesday(x) & Wednesday(y) -> -DifferentFrom(x,y).
Thursday(x) & Thursday(y) -> -DifferentFrom(x,y).
Friday(x) & Friday(y) -> -DifferentFrom(x,y).

%Each god has a unique Roman domain

love(x) | poetry(x) | thunder(x) | war(x).

love(x) & love(y) -> -DifferentFrom(x,y).
poetry(x) & poetry(y) -> -DifferentFrom(x,y).
thunder(x) & thunder(y) -> -DifferentFrom(x,y).
war(x) & war(y) -> -DifferentFrom(x,y).

%Each Greek god has a Roman god name

Mars(x) | Mercury(x) | Jupiter(x) | Venus(x).

Mars(x) & Mars(y) -> -DifferentFrom(x,y).
Mercury(x) & Mercury(y) -> -DifferentFrom(x,y).
Jupiter(x) & Jupiter(y) -> -DifferentFrom(x,y).
Venus(x) & Venus(y) -> -DifferentFrom(x,y).

%1.
-Mercury(Aphrodite).
war(x) -> Mars(x) | Jupiter(x) | Venus(x) .
```

```
89
90  %2.
91  Tuesday(Hermes) & (Venus(x) & Thursday(x)) -> Wednesday(Zeus).
92  Wednesday(Hermes) & (Venus(x) & Friday(x)) -> Thursday(Zeus).
93
94  -Thursday(Hermes).
95  -Friday(Hermes).
96
97  -Friday(Zeus).
98  -Tuesday(Zeus).
99
100 Venus(x)-> Thursday(x) | Friday(x).
101
102 %3.
103 war(Ares) | Jupiter(Ares).
104
105 %4.
106 poetry(Aphrodite) | (Wednesday(x) & poetry(x)).
107
108 %5.
109 poetry(x) -> -Friday(x).
110 poetry(x) -> -Tuesday(x).
111
112 %6.
113 Jupiter(x) -> Thursday(x) | Friday(x).
114
115 %7.
116 war(x) -> Tuesday(x) | Wednesday(x).
117 war(x) -> -Jupiter(x).
118
119 %8.
120 thunder(x) -> Tuesday(x) | Wednesday(x) | Thursday(x).
121
122 %9
123 love(Aphrodite) -> -Friday(x) & love(x).
124 Friday(x) & love(x) -> -love(Aphrodite).
125
126 end_of_list.
127
128 formulas(goals).
129
130 end_of_list.
```

3.2 Output file

```
1  interpretation( 4, [number = 1,seconds = 0], [
2      function(Aphrodite, [0]),
3      function(Ares, [1]),
4      function(Hermes, [2]),
5      function(Zeus, [3]),
6      relation(Friday(_), [0,1,0,0]),
7      relation(Jupiter(_), [0,1,0,0]),
8      relation(Mars(_), [0,0,0,1]),
9      relation(Mercury(_), [0,0,1,0]),
```

```
    relation(Thursday(_), [1,0,0,0]),
    relation(Tuesday(_), [0,0,1,0]),
    relation(Venus(_), [1,0,0,0]),
    relation(Wednesday(_), [0,0,0,1]),
    relation(love(_), [0,1,0,0]),
    relation(poetry(_), [1,0,0,0]),
    relation(thunder(_), [0,0,1,0]),
    relation(war(_), [0,0,0,1]),
    relation(DifferentFrom(_,_), [
        0,1,1,1,
        1,0,1,1,
        1,1,0,1,
        1,1,1,0])]).
```

Explanation: The output contains an interpretation consisting of a set of functions and relations, which actually describe a model produced by Mace4. For example, function(Hermes,[2]) tells us "Hermes has been assigned the number 2 " (which is actually the third god). The relation(love( ), [0,1,0,0]) tells us "The second god(wich is Arens if we look above.) fears is the god of love".

The results satisfy the conditions, but there are false in mythology. I suppose that one of the statement might be wrong.

Interpreting the output, we obtain the following results, which respect the conditions

- Aphrodite, the god of poetry, was called Venus, was worshipped Thursday.
- Ares, the god of love, was called Jupiter, was worshipped Friday.
- Hermes, the god of thunder, was called Mercury, was worshipped Tuesday.
- Zeus, the god of war, was called Mars, was worshipped Wednesday.

4 Bibliography

- https://www.brainzilla.com/logic/logic-grid/
- https://www.brainzilla.com/logic/zebra/