CS_205 Declarative Programming 30 November 2020
Ulrich Berger/Monika Seisenberger

Assignment 2: Due 14 December 2020

1. This coursework will be submitted in **groups of three** (at most two groups may have a size of two). As a group, you are asked to get together, discuss your ideas, plan the solutions, compare your solutions, etc. Everyone needs to be able to explain the main ideas in the submission.

2. Please register in groups of three on CANVAS by Friday 4th December.

3. Please submit **exactly one Prolog file for the whole coursework** and add the answers to additional questions as comments. The file must be named `<your groupnumber>.pl` (for example `12.pl`).

4. The university takes plagiarism very seriously. With the submission you take responsibility that this is the group's own solution, and that the group has implemented the code, rather than someone having taken it from somewhere else.

5. To avoid confusion please **put your group name, your names and student numbers at the beginning of the file**. If somebody has not contributed to, say, one question in your solution, you can state this at the beginning of the file, and they will not be awarded the marks for it. However, please keep this simple.

6. The required Prolog predicates **must be named as prescribed in the questions**.

7. **The submitted file must compile**. If you cannot get certain programs to work, then comment them out and (very briefly) explain the problem.

Rules for awarding marks:

- Full marks are awarded for a solution that is correct, complete, well-structured, makes good use of functional programming concepts, and, in case of a complex solution, contains explanations of the overall strategy and the roles of the helper functions. In addition, marks will be set aside for neat presentation and conforming to the rules which include to register in a group in time.

Question 1.

Write the following program and compile it:

% Program: ROYAL

```
parent(mary,georgeVI).
parent(mary,henry).
parent(mary,george).
parent(georgeV,georgeVI).
parent(georgeV,henry).
parent(georgeV,george).
parent(elizabeth,elizabethII).
parent(georgeVI,elizabethII).
parent(alice,richard).
parent(henry,richard).
parent(elizabethII,charles).
parent(elizabethII,andrew).
parent(elizabethII,anne).
parent(elizabethII,edward).
parent(philip,charles).
parent(philip,andrew).
parent(philip,anne).
parent(philip,edward).
parent(diana,william).
parent(diana,harry).
parent(charles,william).
parent(charles,harry).
parent(sarah,beatrice).
parent(sarah,eugenie).
parent(andrew,beatrice).
parent(andrew,eugenie).
parent(anne,peter).
parent(anne,zara).
parent(mark,peter).
parent(mark,zara).
parent(kate,georgejun).
parent(kate,charlotte).
parent(kate,louis).
parent(william,georgejun).
parent(william,charlotte).
parent(william,louis).
parent(meghan,archie).
parent(harry,archie).
```

The following two predicates define the lists of all female respectively male members of the Royal Family:

```
the_royal_females([mary,elizabeth,elizabethII,alice,anne,diana,sarah,
                   beatrice,zara,eugenie,charlotte,kate,meghan]).
the_royal_males([georgeV,georgeVI,george,philip,charles,andrew,edward,
     richard,henry, william,harry,peter,georgejun,mark,louis,archie]).
```

Define the following predicates on the persons in the program `ROYAL`.

(1) `the_royal_family/1` (This predicate should hold for a list of all members of the Royal Family. Use the predicates `the_royal_females/1` and `the_royal_males/1` defined above)

(2) `mother/2`

(3) `grandma/2`

(4) `has_child/1`.

(5) `ancestor/2`

(6) `sibling/2`

(7) `sister/2`

Translate the following questions into Prolog queries and try them out; include the query and answers as comments (please leave out repetitions though). [You may want to define additional predicates.]

(8) Who is a grandchild of George V?

(9) Who has a child (one or more)?

(10) Who is an ancestor of Archie?

(11) Who is a cousin of Eugenie? (Cousins are persons who have parents that are siblings.)

(12) Who has a cousin who is grandma?
(Define a predicate `has_cousin_who_is_grandma`).

(13) Who has a brother who is a grandfather?

[35 marks]

Question 2.

Consider the following database for train connections:

```
train(swansea, cardiff, [3,5,8,15,17,18,19,20,23],1,[4,5,6,7,10,14,18,22,23],2).
train(cardiff, manchester, [7,11,16],4,[8,14,19],5).
train(cardiff, bristol, [3,5,7,11,15,18,19,20],2,[5,6,7,10,14,16,18,22],2).
train(manchester, bristol, [5,6,7,8,11,15,18,19,20],4,[5,6,7,10,14,16,18,22],5).
train(manchester, swansea, [7,11,16],5,[8,14,19],6).
train(manchester, london, [6,7,11,16],4,[7,8,14,19],5).
train(cardiff, london, [5,6,7,11,18,19,20],3,[8,9,17,18,19,20,21],3).
train(london, brussels, [6,7,8,11,13,17,18,20],5,[9,11,13,16,17,18,19,23],5).
train(london, paris, [7,11,13,17,18,20],5,[9,11,13,16,18,20],6).
train(paris, brussels, [7,11,17],4,[9,13,19],3).
train(paris, munich, [7,11,13,17,22],8,[5,9,13,19,23],7).
train(munich, vienna, [8,9,11,13,17,19],6,[9,10,12,16,18,23],5).
train(vienna, venice, [5,7,8,10,13,16,12,23],8,[2,4,7,9,12,20,21,23],9).
train(venice, paris, [4,11,20],11,[9,12,21],10).
```

For example, the fact

```
train(cardiff, manchester, [7,11,16], 4, [8,14,19], 5).
```

means that there are

- 3 trains daily from Cardiff to Manchester, departing at 7 am, 11 am and 4 pm with a duration of 4 hours each,
- 3 trains daily from Manchester to Cardif, departing at 8 am, 2 pm, and 7 pm with a duration of 5 hours each.

It is your task to write a Prolog program `connection/5` that finds connections between cities, including a list of intermediate stops and all arrival and departure times.

A city may occur at most once as an intermediate stop.

The rule for intermediate stops is that there must be at least 1 and at most 4 hours waiting time between connecting trains.

The list of intermediate stops must consist of entries of the form `via(CArr,C,CDep)` where `C` is a city, `CArr` and `CDep` are the arrival and departure times at `C`.

Times must be given as numbers between 0 and 23.

`connection(X,Dep,V,Arr,Y)` should mean that `X` and `Y` are origin and destination of a journey departing at time `Dep` and arriving at time `Arr`, and `V` is the list of intermediate stops.

For example, the query

```
?- connection(swansea,Dep,V,Arr,munich).
```

should give as one of many answers:

```
Dep = 3,
V = [via(4,cardiff,5),via(8,london,11),via(16,paris,17)],
Arr = 1
```

This particular answer means:

> Departure from Swansea to Cardiff at 3 am. Arrival Cardiff at 4 am.
>
> Departure from Cardiff to London at 5 am. Arrival London at 8 am.
>
> Departure from London to Paris at 11 am. Arrival Paris at 4 pm.
>
> Departure from Paris to Munich at 5 pm. Arrival Munich at 1 am on the next day.

Develop your program step by step as follows:

(a) Define a predicate `direct/4` such that `direct(X,Y,XDeps,XDur)` means that there is are direct trains from city `X` to city `Y` at times `XDeps` with duration `Dur`. For example, `direct(london,manchester,[7,8,14,19],5)` should hold.

[10 marks]

(b) Define a predicate `con/6` such that `con(X,Dep,V,Arr,Y,Cs)` means the same as `connection(X,Dep,V,Arr,Y)` however with the restriction that `Y` and all cities occurring in `V` must be drawn from the list of cities `Cs`.

[35 marks]

(c) Define the final predicate `connection/5` using the predicate `con/6` where `Cs` is the list of all cities minus the city where the journey starts.

[15 marks]

Hint: You may use the predicates `between/3` and `mem_rem/3` from the last lab sheet.

Write a well-structured document complying with the rules as described at the beginning.

[5 marks]

Overall available: [100 marks]