Heinrich Heine Universität Düsseldorf

Software Engineering and Programming Languages
Prof. Dr. Michael Leuschel
David Geleßus, Fabian Vu

Introduction to Logic Programming – WS 2023
Exercise Sheet 9

1 Exercises

Exercise 1 (Lecture – Inverted Classroom)
Watch the lecture videos *16 Minimax*[1] and *17 Minimax Chess*[2] in the HHU Mediathek. The corresponding slides are uploaded in ILIAS: 10_Minimax.pdf (slides 1-82)

The complete playlist is available at: `https://mediathek.hhu.de/playlist/691`.

Note: you have to log in with your HHU account (Uni-Kennung) to see the lecture videos!

The exercises will be discussed on 19th December 2023.

Exercise 2 (Higher-Order Predicates)
In general, a higher-order predicate is a predicate which receives another predicate as its argument.

`call/?` is a predicate which receives a predicate call in the first argument and adds all remaining arguments to this predicate before actually calling it.

For instance, the following calls are the same as calling `member(X, [1])`.

```
?- call(member, X, [1]).
X = 1.
?- call(member(X), [1]).
X = 1.
```

In Prolog, we can also (de-)construct terms using `=../2` (so called Univ operator).

For instance:

```
?- Pred =.. [member,X,[1]], Pred.
Pred = member(1, [1]),
X = 1.
```

a) Implement a predicate `mymaplist/3` which behaves like the corresponding implementation in SWI-Prolog (see `maplist/3`).

[1] `https://mediathek.hhu.de/watch/6c079f4e-994e-4c0b-9e37-ff0d3f6914cd`
[2] `https://mediathek.hhu.de/watch/8e329df6-9e55-4eef-a7c1-abcfe0a7f28f`

`mymaplist/3` receives a predicate in the first argument, a list in the second argument, and returns a new list in the third argument which contains all elements after applying the predicate to the list's elements.

Examples:

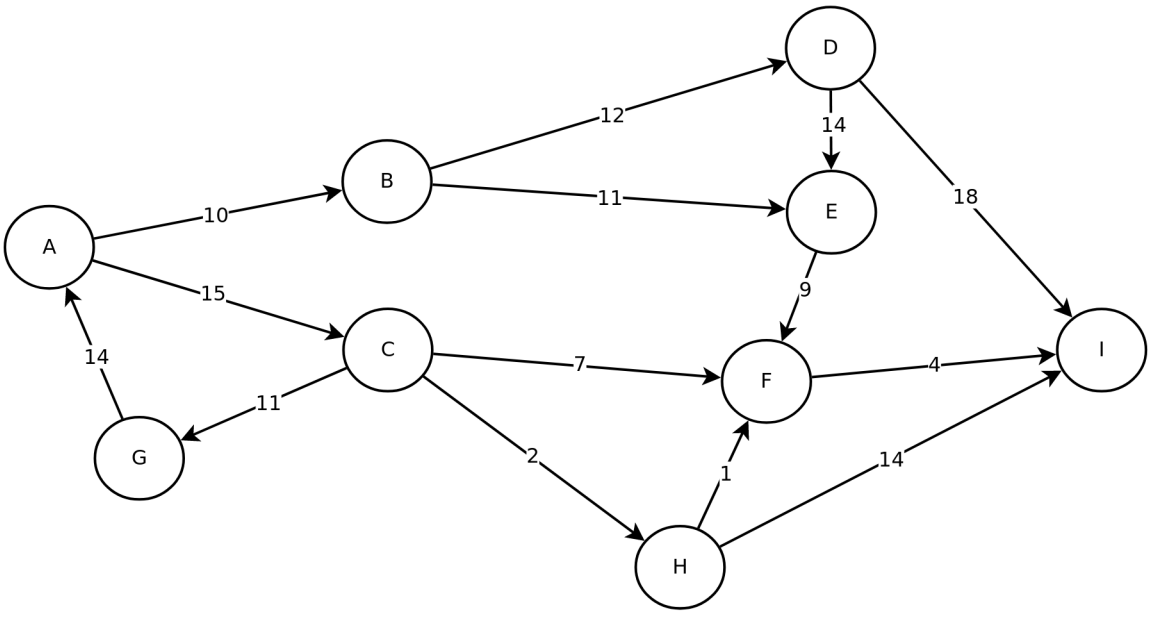
```
?- mymaplist(atom_concat(z),[a,b,c],Mapped).
Mapped = [za,zb,zc].
?- mymaplist(append([1]),[[2],[3]],Mapped).
Mapped = [[1,2],[1,3]].
```

b) Implement a predicate `myinclude/3` which receives a predicate in the first argument and a list in the second argument. In the third argument, the predicate should return a new list of all elements of the list provided in the second argument for which the predicate is true. The order of elements should be kept.

Example:

```
?- myinclude(ground,[a,_,b,c,_,_,_,_],Filtered).
Filtered = [a,b,c].
?- myinclude(integer,[2,a,1,b,c],Filtered).
Filtered = [2,1].
```

Exercise 3 (Informed Search - A*)
The following figure shows a directed graph with weighted edges describing the actual costs between two nodes.



The heuristic function `h` is defined as follows:
`h(A) = 22, h(B) = 20, h(C) = 3, h(D) = 14, h(E) = 9, h(F) = 1, h(G) = 32, h(H) = 2, h(I) = 0`

Find the shortest path from the node `A` to the node `I` by applying the A* algorithm.

State all computed f-values and the queue in each step.

Exercise 4 (Prolog Lists)
Implement a predicate `greater_nrs_only(+Nr, +L, -NL)` which removes all elements of `L` that are *not a number* greater than `Nr`.

a) implement `greater_nrs_only(+Nr, +L, -NL)` by iterating over the list

b) implement `greater_nrs_only(+Nr, +L, -NL)` by using `include/3` from the prior exercise

Examples:

```
1 ?- greater_nrs_only(0, [1,1,5,-3,-6,-7,4], X).
2 X = [1,1,5,4]
3
4 ?- greater_nrs_only(1, [1,a,1,f(-5),5,-3,4], X).
5 X = [5,4]
6
7 ?- greater_nrs_only(6, [3,1,k,4], X).
8 X = []
```