Starting with Assignment 2

Step 1: Develop your solutions locally:

Like with Assignment 1, you shall solve the queries locally by using
the **SWI-Prolog-Interpreter** -
either by double-clicking the file a2.pl (or by opening it from command line, i.e.
using **swipl-win** -s a2.pl in **Windows**
or using **./swipl** -s a2.pl in **MacOS** or **Linux**)

and the **SWI-Prolog-Editor .**

Step 2: Complete the assignments before you upload your solutions:

Upload your commented solutions files in time, but not earlier than when you have done all assignments and commented your solutions as described below.

Before you upload your solutions, do a double check using the following tester2 program.
The version of the test program differs depending on the operating system:

1. for Windows use **tester2**.exe
2. for MacOS use **tester2**
3. for OpenSuse Linux use **tester2suse**

1. Assuming that your program is called **a2.pl** and that you want to check
your solution to query **1** (i.e. lastE(…,…)), in **Windows** use a CMD-Shell to call:
> tester**2** a**2** **1**
(here, you shall use the Windows version tester**2**.exe)

2. Assuming that your program is called **a2.pl** and that you want to check
your solution to query **1** (i.e. lastE(…,…)), in **MacOS** use a shell to call:
> **./**tester**2** a**2** **1**
(here, you shall use the MacOS version tester**2**)

3. Assuming that your program is called **a2.pl** and that you want to check
your solution to query **1** (i.e. lastE(…,…)), in **OpenSuse Linux** use a shell to call:
> **./**tester**2suse** a**2** **1**
(here, you shall use the OpenSuse Linux version tester2suse)

Actions to be taken depending on the feedback of the program tester2:

If you get an error message query q1/1 is undefined, make sure that you have named your procedures like in Assignment 2, i.e. secondLastE(…), … and that the number of parameters is correct.
If the program tester2 tells you that your solution is different from the wanted solution, look at the counter example presented to show the difference - and rewrite your query.
May be, you have to do this multiple times until you have a correct solution.

If the program tester2 tells you that your solution is equivalent to the wanted solution, but it takes more clauses and/or more goals, try to improve your solution (i.e., find a shorter solution). May be, you have to do this multiple times until you have a short solution.

If program tester2 tells you that your solution is most likely correct, the tester could not detect any difference. Then, you should prepare for explaining your solution.
You are requested to write a short (2-3 lines) comment about your solution below your code as part of the file that you upload. Your comment shall contain one sentence about the recursion termination and one or two sentences about the recursive problem reduction.

For example:

secondLastE(…) :- < here is your implementation of secondLastE >

% In the recursion termination, I check … because …
% The recursive problem reduction is … because …

Finally, before you upload your solution to PANDA:

1. Doublecheck that you have commented all your solution queries.
2. Thereafter (!), doublecheck that your program still compiles correctly.
3. Doublecheck, that you have solved all queries as required by typing
   > tester2 a2 -1 (on Windows)
   > ./tester2 a2 -1 (on MacOS)
   >. ./tester2suse a2 -1 (on Linux in the University Pool)
   and you get messages your solution is most likely correct for subtasks 1 to 15 .