Exercise 10.1

Three students quarrel over who has the best result in an exam.

- Bruno says: “I have the best result.”
- Christina says: “I have the best result.”
- Frank says: “I have the best result.”
- Christina says: “Bruno does not have the best result.”
- Frank says: “Christina does not have the best result.”

You know that only the student with the best exam tells the truth. Write a Prolog program that gives as answer the name of the person who tells the truth.

Hints for realization in Prolog:
1) Define facts for a predicate one_says_truth(S1, S2, S3) which means that only one of the students S1, S2, and S3 tells the truth. If a student Si tells the truth then Si should get the value true, false otherwise.
2) For the assertions by students, referring to other students, define some facts using a binary predicate says_not_best(S1, S2), where the values true and false are used as in 1).
3) Define a predicate who_has_the_best_result(Bruno, Christina, Frank) which gives the answer to the question who has the best result in the exam and that uses the predicates one_says_truth and says_not_best.

Exercise 10.2

a. Define a predicate expand(Term, Arg, NewTerm) which expands the given Term with the new argument Arg.

Hint: Use the Univ-operator ’=.,’.

Example: ?- expand(p(a(X), b, c), f(Y), NewTerm). yields NewTerm = p(a(X), b, c, f(Y)).

b. Define a predicate inst(Term) which has the result true if each variable in the term Term is instantiated.

Example: ?- X = d, inst(p(a(X), b, c)). yields X = d.

Hint: Use the built-in predicates atomic, and var but not ground.

c. Define a predicate count_type(Term, Type, Number) which counts how many terms of the given Type are contained in the given Term. Take into account the types atom, variable, and number.

Example: ?-count_type(p(a(X), X, [b, Y]), variable, Number). yields Number = 3.

Hints for realization in Prolog:
Read about term inspection facilities in chapter 11 of book K. Apt: From Logic Programming to Prolog, and use the help system of SWI-Prolog.
Define a rule \( \text{func}(F, X) \), where \( X \) is the argument of a function \( f \) given as arithmetic expression \( F \).

The rule repeats the following steps:

1) ask for the value of a variable \( X \) (the value should be a number or 'stop').
2) if you input 'stop' then the end is reached.
3) if you don’t input a number or 'stop', the rule requests to improve the value.
4) computes the value of the function \( f(X) \) (defined by \( F \)).

Example: \( ?- \text{func}(3*X+1, X). \)

\[ \begin{align*}
X &= \\
|: & 5.3. \\
\text{Value of } f(X): & 16.9 \\
X &= \\
|: & 7.8. \\
\text{Not a correct value for } X! \\
X &= \\
|: & \text{stop.} \\
X &= \text{stop}
\end{align*} \]

Hints for realization in Prolog:

Use the built-in predicates \( \text{repeat}, \text{fail}, \text{read}, \text{write}, \text{nl}, \text{number} \), and the cut. (\( |: \) is the automatic prompt for read.) Use \( \text{nl} \) (newline) after \( \text{write} \) and before \( \text{read} \).