Type-Checker

```
type(N,entier):- integer(N).
type(R,reel):- real(R).
type(vrai,booleen).
type(faux,booleen).
type(Expression,Type):-
       Expression =.. [Op,E1,E2],
       type(E1,T1),
       type(E2,T2),
       type(Op,T1,T2,Type).
        /* remarquez l'utilisation de l'arite variable des predicats Prolog */
type(_,erreur,erreur,erreur):- !. /* le seul cut que je m'autorise car */
                                  /* devoir ecrire dans tous les autre cas */
                                 /* T \== erreur deviendrait vite fastidieux */
type(Op,T,T,booleen):-
       egal_ou_different(Op),
       T\==erreur.
type(Op,T1,T2,erreur):-
       egal_ou_different(Op),
       T1\==T2.
type(Op,erreur,erreur,erreur):- egal_ou_different(Op).
type(Op,booleen,booleen,booleen):- et_ou_ou(Op).
type(Op,T,_,erreur):-
       et_ou_ou(Op),
       T\==booleen.
type(Op,_,T,erreur):-
       et_ou_ou(Op),
       T\==booleen.
type(Op,entier,entier,entier):- plus_ou_moins_ou_mult(Op).
type(Op,reel,reel,reel):- plus_ou_moins_ou_mult(Op).
type(Op,reel,entier,reel):- plus_ou_moins_ou_mult(Op).
type(Op,entier,reel,reel):- plus_ou_moins_ou_mult(Op).
type(Op,T,_,erreur):-
       plus_ou_moins_ou_mult(Op),
       hors_de(T,[reel,entier]).
type(Op,_,T,erreur):-
       plus_ou_moins_ou_mult(Op),
       hors_de(T,[reel,entier]).
type(div,T1,T2,reel):-
       element(T1,[entier,reel]),
       element(T2,[entier,reel]).
type(div,T,_,erreur):-
       hors_de(T,[entier,reel]).
type(div,_,T,erreur):-
       hors_de(T,[entier,reel]).
type(divent,entier,entier,entier).
type(divent,T,_,erreur):- T\==entier.
type(divent,_,T,erreur):- T\==entier.
egal_ou_different(egal).
egal_ou_different(different).
et_ou_ou(et).
et_ou_ou(ou).
plus_ou_moins_ou_mult(plus).
plus_ou_moins_ou_mult(moins).
plus_ou_moins_ou_mult(mult).
/* un cas tres utile de l'utilisation de la negation: la specification */
/* differentielle. */
real(R):-
       number(R),
       \+ integer(R).
element(X,[X|_]).
element(X,[Y|L]):- element(X,L).
hors_de(X,[]).
hors_de(X,[Y|L]):-
       X\==Y,
       hors_de(X,L).
```