```
:- op(100,yf,!).

expr(S,T) :-
    constrain(S,S2,[],[S1,S2],["+","-"],[],[]),
    !,subexpr(S1,T1,O1), term(S2,T2),
    build(T,T1,T2,[O1,T1,T2]).

subexpr("",nil,nil) :- !.
subexpr(S,T,Op) :-
    constrain(S,S2,[O],[S1,S2,[O]],["+","-"],[],[]),
    char_code(Op,O),
    !,subexpr(S1,T1,O1),term(S2,T2),
    build(T,T1,T2,[O1,T1,T2]).

term(S,T) :-
    constrain(S,S2,[],[S1,S2],["*","/"],[],[]),
    !,subterm(S1,T1,O1), factor(S2,T2),
    build(T,T1,T2,[O1,T1,T2]).

subterm("",nil,nil) :- !.
subterm(S,T,Op):-
    constrain(S,S2,[O],[S1,S2,[O]],["*","/"],[],[]),
    !,subterm(S1,T1,O1),factor(S2,T2), char_code(Op,O),
    build(T,T1,T2,[O1,T1,T2]).

factor(S,T) :-
    constrain(S,S1,[],[S1,S2],["^"],S2,["!"]),
    !,base(S1,T1), restexp(S2,T2,O2),
    build(T,T2,T1,[O2,T1,T2]).

restexp("",nil,nil) :- !.
restexp(S,T,^) :-
    constrain(S,S1,"^",["^",S1,S2],["^"],S2,["!"]),
    !, base(S1,T1), restexp(S2,T2,O2),
    build(T,T2,T1,[O2,T1,T2]).

base(S,T!) :-
    constrain(S,S1,[],[S1,"!"],[],[],[]),!,
        factarg(S1,T).
base(S,T) :- factarg(S,T).

% this part used to be the <base>
factarg(S,T) :- append(["(",S1,")"],S), !, expr(S1,T).
factarg([S],A) :- 97 =< S, S =< 122, char_code(A,S).
```

```
%build(_,_,_,_) :- !.
build(T,nil,T,_) :- !.
build(T,_,_,L) :- T =.. L .

constrain(S,S1,O,L,OL,S2,NL) :-
    S1 = [_|_], append(L,S), balanced(S1,R1),
    findall(X,(member([X],OL),member(X,R1)),[]),
    (   S2 = [H|_] -> \+ member([H],NL) ; true ),
    (   O \= [] -> member(O,OL) ; true ).

balanced("","") :- !.
balanced(S,"") :-
    append(["(",S1,")"],S),balanced(S1,_),!.
balanced(S,R) :-
    append([X],S1,S), \+ member([X],["(",")"]),!,
    balanced(S1,R1), append([X],R1,R).
balanced(S,R) :-
    append(S1,[X],S), \+ member([X],["(",")"]),!,
    balanced(S1,R1), append(R1,[X],R).
balanced(S,R) :-
    append(["(",S1,")",S2,"(",S3,")"],S),
    balanced(S1,_),balanced(S2,R),balanced(S3,_).
```