```
:-dynamic pos / 2,
          notalllowed / 2,
          size / 2.


melem(X,[X|_]).
melem(X,[_|T]) :- melem(X,T).

diff([],_,[]).
diff([H|T],S,R) :-
  melem(H,S),!,
  diff(T,S,R).
diff([H|T],S,[H|R]):-
  diff(T,S,R).

union([],R,R).
union([H|T],S,R) :-
  melem(H,S),!,
  union(T,S,R).
union([H|T],S,[H|R]) :-
  union(T,S,R).

xor(S1,S2,R) :-
  diff(S1,S2,D1),
  diff(S2,S1,D2),
  union(D1,D2,R).


/* ---------------------------------- */


paths(X1,Y1,X2,Y2,XS,YS,X,R) :-
  within(X1,XS), within(X2,XS),
  within(Y1,YS), within(Y2,YS),
  not((X1 = X2,Y1 = Y2)),
  retractall(size(_,_)),
  retractall(pos(_,_)),
  assertz(size(XS,YS)),
  setof(F,lookp(X1,Y1,X2,Y2,X,F),RES),
  shortest(RES,R).

within(X,S) :-
  X > 0, X =< S.

% pripadne shortest([],[]) :- !,fail.
shortest([],[]) :- !.
shortest([H|T],R) :- shr(H,T,R).

shr(M,[],M).
shr(M,[H|T],R) :-
  shr(M,T,W),
  length(H,LH),
  length(W,LW),
  (LH<LW -> R=H ; R=W).

lookp(X,Y,X,Y,_,[X:Y]).
lookp(X,Y,X,Y,_,_) :- !,fail.
lookp(X,Y,XX,YY,F,[X:Y|PP]) :-
  assertz(pos(X,Y)),
  nxt(X,Y,NX,NY),
  not(pos(NX,NY)),
  Tst =.. [F,X,Y], not(call(Tst)),
  lookp(NX,NY,XX,YY,F,PP).
lookp(X,Y,_,_,_,_) :-
  pos(X,Y),
  retract(pos(X,Y)),
  !,fail.


nxt(X,Y,X,YY) :-
  YY is Y+1,
  size(_,S),
  within(YY,S).
nxt(X,Y,X,YY) :-
  YY is Y-1,
  size(_,S),
  within(YY,S).
nxt(X,Y,XX,Y) :-
  XX is X+1,
  size(S,_),
  within(XX,S).
nxt(X,Y,XX,Y) :-
  XX is X-1,
  size(S,_),
  within(XX,S).


/* ---------------------------------- */

search(T,_,_) :-
  var(T),
  !, fail.
search(_,K,V) :-
  var(K), var(V),
  !, fail.
search(T,K,V) :-
  var(K), nonvar(V),
  getKeys(T,V,[X|XS]), !,
  K=[X|XS].
```

```
search(_,K,V) :-
  var(K), nonvar(V),
  !, fail.
search(T,K,V) :-
  nonvar(K),
  getVal(T,K,V).

getVal(leaf,_) :- !, fail.
getVal(node(K,V,_,_),K,V) :- !.
getVal(node(KK,_,L,_),K,V) :-
  K < KK,
  getVal(L,K,V).
getVal(node(KK,_,_,R),K,V) :-
  K > KK,
  getVal(R,K,V).

getKeys(leaf,_,[]).
getKeys(node(K,V,L,R),V,[K|A]) :-
  !,getKeys(L,V,LR),
  getKeys(R,V,RR),
  append(LR,RR,A).
getKeys(node(_,_,L,R),V,A) :-
  getKeys(L,V,LR),
  getKeys(R,V,RR),
  append(LR,RR,A).


/* ------------------------------------ */

splitWith(_,[],[],[]).
splitWith(P,[H|T],R1,R2) :-
  splitWith(P,T,X1,X2),
  C =.. [P,H],
  ( (call(C),!,R1=[H|X1],R2=X2);
    (R1=X1,R2=[H|X2])
  ).

tst(X) :- X<5.

/* ------------------------------------ */
```