distance((X1,Y1),(X2,Y2),D) :- D is sqrt(((X1-X2)\*\*2)+((Y1-Y2)\*\*2)).

fib(0,1).

fib(1,1).

fib(X,Y) :- X>1, A is X-1, B is X-2,

fib(A,M), fib(B,N),

Y is M+N.

fibg(0,0,1).

fibg(1,1,1).

fibg(X,Y,Z) :- X>1, A is X-1, fibg(A, M, Z), Y is Z+M.

fibb(X,Y) :- fibg(X,\_,Y).

linie(0,\_) :- nl.

linie(B,X) :- B>0, write(X), A is B-1, linie(A,X).

patrat(N,X) :- N>0, B is N, linie(B,X), write('\n'), A is N-1, patrat(A,X).

Listele in prolog sunt de doua feluri – vida sau nevida.

[]

[Head|Tail]

H== orice

T== lista

[7|[]]

[5|[7|[]]]=[5,7]

Lg([],0)

Lg([\_|T],X):- lg(T,Y), X is Y+1.

all\_a([]).

all\_a([a|B]) :-all\_a(B).

rev([],[]).

rev([H|T], M) :- rev(T,U), append(U,[H],M).

palindrom(L) :- rev(L,L).

remove\_duplicates([],[]).

remove\_duplicates([H|T], U) :- remove\_duplicates(T,U), member(H,U).

remove\_duplicates([H|T], [H|U]) :- remove\_duplicates(T,U), not(member(H,U)).\*/

atimes(\_,[],0).

atimes(N,[N|T],W) :- atimes(N,T,Y), W is Y+1.

atimes(N,[H|T],Y) :- atimes(N,T,Y), N\==H.

Sortari

L este o lista sortata

insert(X,L,M) <-> M este obtinuta prin inserarea lui X in poz corespunzatoare in L

insert(X,[],[X]).

insert(X,[H|T], [X|[H|T]]) :- X<H.

insert(X,[H|T],[H|L]) :- X>=H, insert(X,T,L).

quicksort

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

H- ------- T ------------

----A-----|H|-----B------

Split(H,T,A,B)

split(\_,[],[],[]).

split(H,[X|T],A,[X|B]) :- split(H,T,A,B), H<X.

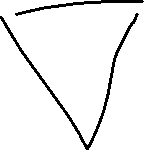
split(H,[X|T],[X|A],B) :- split(H,T,A,B), H>=X.

quicksort([],[]).

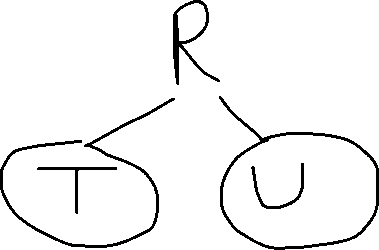
quicksort([H|T],L) :-split(H,T,A,B), quicksort(A,M), quicksort(B,N),

append(M,[H|N],L).

Arbori



Arbori:



1.vid

2. arb(R,T,U)

bt\_ins(X,vid,arb(X,vid,vid)).

bt\_ins(X,arb(R,T,U),arb(R,V,U)) :- X<R, bt\_ins(X,T,V).

bt\_ins(X,arb(R,T,U),arb(R,T,V)) :- X>=R , bt\_ins(X,U,V).

bt\_list([],vid).

bt\_list([H\T],V) :- bt\_list(T,W), bt\_ins(H,W,V).

srd(vid,[]).

srd(arb(R,T,U),L) :- srd(T,L1), srd(U,L2), append(L1,[R|L2],L).

bt\_ins(X,vid,arb(X,vid,vid)).

bt\_ins(X,arb(R,T,U),arb(R,V,U)) :- X<R, bt\_ins(X,T,V).

bt\_ins(X,arb(R,T,U),arb(R,T,V)) :- X>=R , bt\_ins(X,U,V).

bt\_list([],vid).

bt\_list([H|T],V) :- bt\_list(T,W), bt\_ins(H,W,V).

bt\_sort(L,X) :- bt\_list(L,T), srd(T,X).

/\*

length([],0).

length([H|T],N) :- length([T],A), A is N-1.

listaNelem(\_,0,[]).

listaNelem(L,N,[H|T]) :- N>0, member(H,L), M is N-1, listaNelem(L,M,T).

listeNelem(L,N,LL) :- bagof(M, listaNelem(L,N,M), LL).

word(abalone,a,b,a,l,o,n,e).

word(abandon,a,b,a,n,d,o,n).

word(anagram,a,n,a,g,r,a,m).

word(connect,c,o,n,n,e,c,t).

word(elegant,e,l,e,g,a,n,t).

word(enhance,e,n,h,a,n,c,e).

crossword(V1,V2,V3,H1,H2,H3) :- word(V1,\_,L1,\_,L4,\_,L7,\_), word(V2,\_,L2,\_,L5,\_,L8,\_),

word(V3,\_,L3,\_,L6,\_,L9,\_), word(H1,\_,L1,\_,L2,\_,L3,\_),

word(H2,\_,L4,\_,L5,\_,L6,\_), word(H3,\_,L7,\_,L8,\_,L9,\_).

\*/

connected(1,2).

connected(3,4).

connected(5,6).

connected(7,8).

connected(9,10).

connected(12,13).

connected(13,14).

connected(15,16).

connected(17,18).

connected(19,20).

connected(4,1).

connected(6,3).

connected(4,7).

connected(6,11).

connected(14,9).

connected(11,15).

connected(16,12).

connected(14,17).

connected(16,19).

path(X,X,[X]).

path(X,Y,[X|L]) :- connected(X,Z), path(Z,Y,L).

pathc(X,Y) :- path(X,Y,L), L\==[].

pathc(X,Y) :- path(X,Y,\_).

