Θέματα Ιανουάριος 2011-12:

%a)

remove(X,[Head|Tail],L):-

X=\=Head,remove(X,Tail,Y),append([Head],Y,L),!.

remove(X,[Head|Tail],L):-

L=Tail.

%remove(2,[1,2,2,2,2,3,4,4,4,2,2,5],L).

%b)

p(A,B):-

p(A,[],B).

p([],A,A).

p([A|B],C,D):-

p(B,[A|C],D).

% p Reverses a list, or checks if a list is the reversed version of another given list

%p([1,2,3],X). Output: X = [3, 2, 1]

%p(X,[1,2,3]). First output: X = [3, 2, 1], if ; second output: error

%c)

file\_to\_list(File,List):-

setup\_call\_cleanup(

open(File, read, In),

readData(In, List),

close(In)

).

readData(In, L):-

read\_term(In, H, []),

( H == end\_of\_file

-> L = []

; L = [H|T],

readData(In,T)

).

/\*file\_to\_list("testfile.abc",List).\*/

Θέματα Σεπτέμβριος 2011-12:

%a)

adjacent(A,B,[La,Lb|L]):-

A=La,B=Lb,!.

adjacent(A,B,[La,Lb|L]):-

adjacent(A,B,[Lb|L]).

/\*adjacent(2,3,[1,2,3]).

adjacent(2,3,[1,3,2]).\*/

%b)

median(Xs,X):-

add(Xs,MO),length(Xs,Length),

X is MO/Length.

add([],MO):-

MO is 0.

add([X|Y],MO):-

add(Y,NMO),MO is NMO+X.

%median([4,5,6],X).

%c)

guess([]).

guess([Head|Tail]):-

write("Guess letter: "),

read(Letter),Head==Letter,

write("Ok"),nl,guess(Tail).

guess(X):-

write("Fail. Try again!"),nl,

guess(X).

%guess([a,p,p,l,e]).

Θέματα 2013:

%a)

adjacent(A,B,[La,Lb|L]):-

A=La,B=Lb,!.

adjacent(A,B,[La,Lb|L]):-

adjacent(A,B,[Lb|L]).

/\*adjacent(2,3,[1,2,3]).

adjacent(2,3,[1,3,2]).\*/

%b)

insert(X,[Head|Tail],L):-

X>Head,insert(X,Tail,Y),append([Head],Y,L),!.

insert(X,List,L):-

append([X],List,L).

%insert(3,[1,4,9],L).

%c)

change(you,i).

change(are,[am, not]).

change(french, german).

change(do, no).

change(X,X).

alter([Head|Tail],L):-

alter(Tail,Y),change(Head,Res),is\_list(Res),append(Res,Y,L),!.

alter([Head|Tail],L):-

alter(Tail,Y),change(Head,Res),append([Res],Y,L),!.

alter([Head],L):-

change(Head,Res),append([Res],[],L).

/\*alter([do,you,know,french],L).

alter([you,are,a,computer],L).\*/

Θέματα 2017:

%a)

add([Head1],[Head2],NewList):-

append([Head1], [Head2], NewList).

add([Head1|Tail1], [Head2|Tail2], A) :-

add(Tail1,Tail2,X),

append([Head1,Head2],X,A).

%add([10,30,50],[20,40,60],L).

%b)

lay([],[]).

lay([Head|Tail],NewList):-

is\_list(Head),append(Head,Tail,CheckedList),

lay(CheckedList,NewList).

lay([Head|Tail],NewList):-

lay(Tail,List),append([Head],List,NewList),!.

%lay([1,[2,3],[[4]],[[]]],X).

%c)

cluster([A|B],X,Y):-

A=\=32,cluster(B,X1,Y),append([A],X1,X).

cluster([A|B],X,Y):-

A==32,Y=B,X=[].

/\*cluster([12,26,44,32,65,13],A,B).\*/

cluster2(S):-

name(S,L),cluster(L,X,Y),name(String1,X),

write("Next Item is "),write(String1),write("\n"),

name(String2,Y),cluster2(String2).

cluster2(S):-

write("Last Item is "),write(S).

/\*cluster2("This is a sentence").\*/

Θέματα 2020:

%a) between\_(1,4,Y). X=2, X=3

between\_(A,B,X):-

A<B-1,X is A + 1.

between\_(A,B,Y):-

A<B-1,L is A + 1,between\_(L,B,Y).

%between\_(1,4,Y).

%c)

maximum([Head],B):-

B=Head.

maximum([Head|Tail],B):-

maximum(Tail,Y),Y>Head,B=Y,!.

maximum([Head|Tail],B):-

B=Head,!.

%maximum([4,8,12,6,12,5],X).

grade(john,9).

grade(mary,8).

grade(michael,10).

grade(henry,7).

grade(george,10).

grade(bill,9).

grade(martin,7).

best\_student(X):-

findall(B,grade(\_,B),Bag),

maximum(Bag,Best),findall(Name,grade(Name,Best),X).

% same as bagof(Name,grade(Name,Best),X)

%best\_student(X).

Θέματα 2021:

%Δ-Ζ

insert(X,[Head|Tail],L):-

X>Head,insert(X,Tail,Y),append([Head],Y,L),!.

insert(X,List,L):-

append([X],List,L).

/\*insert(0,[1,4,9],L).

insert(10,[1,4,9],L).

insert(3,[1,4,9],L).\*/

extract(\_,0,[]).

extract(Xs,N,[X|Zs]) :- N > 0,

length(Xs,L),

I is random(L) + 1,

remove\_at(X,Xs,I,Ys),

N1 is N - 1,

extract(Ys,N1,Zs).

remove\_at(X,[X|Xs],1,Xs).

remove\_at(X,[Y|Xs],K,[Y|Ys]) :- K > 1,

K1 is K - 1, remove\_at(X,Xs,K1,Ys).

%extract([1,2,3,4,5,6,7,8,9],3,X).

%Μ-Ν

part([X1,X2|\_],X1,X2,R):-

R=[X1,X2].

part([X1,X2|T],X1,Y,R):-

part([X2|T],X2,Y,Rt),R=[X1|Rt],!.

part([H|T],X,Y,R):-

H\=X,part(T,X,Y,R).

%part([1,2,3,4,5,6,7,8,9],3,6,X).

%Ξ-Π

check([H],Y):-

append([],[H],Y).

check([H1,H2|T],Y):-

append([H2],T,CH),

check(CH,Y1),H1=:=H2,append([H1],[=],Yadd),append(Yadd,Y1,Y).

check([H1,H2|T],Y):-

append([H2],T,CH),

check(CH,Y1),H1<H2,append([H1],[<],Yadd),append(Yadd,Y1,Y).

check([H1,H2|T],Y):-

append([H2],T,CH),

check(CH,Y1),H1>H2,append([H1],[>],Yadd),append(Yadd,Y1,Y).

%check([7,9,9,2],X).

part([X1,X2|\_],X1,X2,R):-

R=[X1,X2].

part([X1,X2|T],X1,Y,R):-

part([X2|T],X2,Y,Rt),R=[X1|Rt],!.

part([H|T],X,Y,R):-

H\=X,part(T,X,Y,R).

%part([1,2,3,4,5,6,7,8,9],3,6,X).