Aula 03/10

Quick sort

length1 (L, N):- length2(L,0,N).

length2 ([], Acc Acc).

length2 ([| T], Acc, N):-

Acc1 is Acc+1

lenght2(T, Acc1, N).

partition1 (3,[7,1,0,2,8,-1,4], [1,0,2,-1], [7,8,4]).

partition1 (N,L, Me, Ma):- partition2 (N, L, Acc1, Acc2, Me, Ma).

partition2 (\_, [], Acc1, Acc2, Me, Ma):- reverse ( Acce1, me), reverse( Acc2, Ma).

partition2 (N, [H | T], Acc1, Acc2, Me, Ma) :- H =<N,

partition2 (N,T, [H|Acc1], Acc2, Me, Ma).

partition2( N, [H|T], Acc1, Acc2, Me, Ma) -: H > N,

partition2 (N, T, Acc1, [H | Acc2], Me,Ma).

quicksort([],[]).

quicksort ([H|T], R) :-

partition1 ( H, T, Me, Ma),

quicksort (Me, Me\_ord),

quicksort (Ma, Ma\_ord),

append (Me\_ord, [H| Ma\_ord].

p(a).

p(b).

q(c).

q(d).

q(e).

r(a,d).

r(b,c).

r(b,e).

p(x), q(y), r(x,y).

back-tracking

**Backtracking** é um tipo de algoritmo que representa um refinamento da busca por força bruta, em que múltiplas soluções podem ser eliminadas sem serem explicitamente examinadas. O termo foi cunhado pelo matemático estado-unidense D. H. Lehmer na década de 1950.

p(X),q(Y), r(X,Y).

X = a,

Y = d ;

X = b,

Y = c ;

X = b,

Y = e.

p(X),q(Y),!, r(X,Y). // !corte

false.

p(X),q(Y),!. // Ele viu a e c , dai ele cortou!! Não tem nenhuma escolha.

X = a,

Y = c.

p(X),!,q(Y).

X = a,

Y = c ;

X = a,

Y = d ;

X = a,

Y = e

r(X,Y,Z) :- X >= Y, Z = X.

r(X,Y,Z) :- X < Y, Z = Y.

?- r(5,4,M).

M = 5 ;

false.

rc(X,Y,Z) :- X >= Y,!, Z = X.

rc(X,Y,Z) :- X < Y, Z = Y.

?- rc(4,5,M).

M = 5.

?- rc(5,4,M).

M = 5.

r2(X,Y,Z) :- X >= Y,!, Z = X.

r2(\_,Y,Z) :- Z = Y. // Nao necessita da comparação, porque? Ele só vai entrar aqui se e somente falhar em cima, !!!

?- r2(5,4,M).

M = 5.

?- r2(4,5,M).

M = 5.

re(X,Y,Z) :- X >= Y, Z = X.

re(\_,Y,Z) :- Z = Y.

?- re(5,4,M).

M = 5 ;

M = 4.

?- re(4,5,M).

M = 5.

NOT

p(a).

p(b).

q(c).

q(d).

?- not(p(a)).

false.

?- not(p(c)).

true.

not(X):- X, !, fail.

not(X).

Not é uma uma armadilha ( trap)!!!

Se x for verdade eu deixo ele passar pela porta, dai eu ponho uma parede (corte) sem saída.

→ QUESTÃO DA PROVA

ELE DA AS RELAÇÕES, E OS CORTES!!!

60% = recurção

20% → brack traking

20% →última aula!!!

//Base de dados

homem(tom\_hanks).

homem(pierce\_brosnan).

mulher(meryl\_streep).

mulher(anne\_hathaway).

prof\_teatro(tom\_hanks).

prof\_teatro(pierce\_brosnan).

prof\_teatro(meryl\_streep).

prof\_teatro(anne\_hathaway).

produtor(tom\_hanks).

aatua(naufrago, tom\_hanks).

atua(forest\_gump,tom\_hanks).

atua(o\_diabo\_veste\_prada,meryl\_streep).

atua(o\_diabo\_veste\_prada,anne\_hathaway).

atua(mama\_mia,pierce\_brosnan).

atua(mama\_mia,meryl\_streep).

atua(zerozerosete\_goldeneye, pierce\_brosnan).

produziu(mama\_mia,tom\_hanks).

?-

| atua(mama\_mia,X),homem(X).

X = pierce\_brosnan ;

false.

?- atua(mama\_mia,X),mulher(X).

X = meryl\_streep.

?- atua(X,meryl\_streep),atua(X,Y), not(Y==meryl\_streep).

X = o\_diabo\_veste\_prada,

Y = anne\_hathaway ;

X = mama\_mia,

Y = pierce\_brosnan ;

false.

?- atua(X,Y), mulher(Y), atua(X,W),mulher(W),not(Y==W).

X = o\_diabo\_veste\_prada,

Y = meryl\_streep,

W = anne\_hathaway ;

X = o\_diabo\_veste\_prada,

Y = anne\_hathaway,

W = meryl\_streep ;

dois atores no msm filme:

dois\_atores\_filme(X,Y) :-

homem(X), homem(Y), atua(W,X),

atua(W,Y), not(X==Y).

uma questão com acumulador, e uma questão sem acumulador--- em prolog, haskell

4 de recurção

2 back traking

2 banco de dados

Recurção

insert(3, [0,2,6,9], [0,2,3,6,9]).

insert(N,[],[N]).

insert(N,[H|T],[H|T1]):-

N >= H, !, insert(N,T,T1).

insert(N,L,[N|L]).

insertion\_sort([],[]).

insertion\_sort([H|T],L):-

insertion\_sort(T,T\_ord),

insert(H,T\_ord,L).

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

delete(N,[H|T],[H|T1]):-

N \= H, delete(N,T,T1).

delete(N,[N|T],T).

COM ACUMULADOR

diff([0,1,3,4,7,8],[0,1,2,3,7,8,9],[4],[2,9]).

Diff([0,1,3,4,7,8],[0,1,2,3,7,8,9],[menos(2), mais(4), menos(9)]).