Contents

1 Pre midterm stuff

1.1 SECD virtual machine

1.2 Lambda calculus

1.3 Lisp

2 Post midterm stuff

2.1 Prolog

2.1.1 Code examples

```
% append(L1,L2,L3): append L1 and L2 to get L3
append([],L,L).
append([A|L],L1,[A|L2]) :- append(L,L1,L2).

% member(A,L): A is in list L
member(A,[A|_]).
member(A,[B|L]) :- A \== B, member(A,L).

cartesian([], _, []).
cartesian([A|N], L, M) :-
     pair(A,L,M1),
     cartesian(N, L, M2),
     append(M1, M2, M).
cartesian([a,b], [d,e], [[a,d], [a,e], [b,d], [b,e]]).

pair(_, [], []).
pair(A, [B|L], [[A,B]|N] ) :- pair(A, L, N).

% reverse(X,Y): Y is the reverse of input list X
reverse([], []).
reverse([A|L1], L2) :- reverse(L1, N), append(N, [A], L2).

% sum(L,N) will have N bound to the sum of the numbers in L.
sum([],0).
sum(N,N) :- number(N).
sum([A|L],S) :- sum(A,S1), sum(L,S2), S is S1 + S2.

% flatten(L,L1): flatten a list of atoms (atoms and numbers) L to a flat list L1.
flatten([],[]).
flatten([A|L],[A|L1]) :-
     xatom(A), flatten(L,L1).
flatten([A|L],R) :-
     flatten(A,A1), flatten(L,L1), append(A1,L1,R).

xatom(A) :- atom(A).
xatom(A) :- number(A).
```

2.2 Constraint logic programming

2.3 Constraint satisfaction problem

2.4 Answer set Programming

2.4.1 Another one found in notes todo look for

2.4.2 Ferry problem