Prolog 3-1

Tento týden je test z Prologu !!

opakování a rozšíření

Počet výskytů daného prvku v seznamu:

```
pocet(_,[],0).
pocet(A,[A|L],N) :-
  !,
  pocet(A,L,N1),
  N is N1 + 1.
pocet(A,[_|L],N) :-
  pocet(A,L,N).
```

příklad:
  pocet(a, [a,b,c,a,d,a], X). => X = 3
  pocet(a, [a,b,c,C,d,E], X). => C = a, E = a, X = 3

Počet výskytů daného prvku v seznamu (verze s lepší kontrolou prvků seznamu):

```
pocet2(_,[],0).
pocet2(A,[B|L],N) :-
  atom(B),
  A = B,
  !,
  pocet2(A,L,N1),
  N is N1 + 1.
pocet2(A,[_|L],N) :-
  pocet2(A,L,N).
```

příklad:
  pocet2(a, [a,b,c,a,d,a], X). => X = 3
  pocet2(a, [a,b,c,C,d,E], X). => X = 1

Výběr prvků v seznamu, které dohromady dají požadovaný součet:

```
vyber([],0,[]).
vyber([N|L1],S,[N|L2]) :-
  N =< S,
  S1 is S - N,
  vyber(L1,S1,L2).
vyber([N|L1],S,L2) :-
  vyber(L1,S,L2).
```

příklad:
   vyber([1,2,3,4,5,6],7,X). =>
      X = [1,2,4]
      X = [1,6]
      X = [2,5]
      X = [3,4]
      no

Výběr prvků v seznamu, které dohromady dají požadovaný součet (verze s opakováním prvků):

```
vyber2([],0,[]).
vyber2([N|L1],S,[N|L2]) :-
  N =< S,
  S1 is S - N,
  vyber2([N|L1],S1,L2).          % N posílame do hry znovu
vyber2([N|L1],S,L2) :-
  vyber2(L1,S,L2).
```

příklad:

```
   vyber2([1,2,3,4,5,6],7,X). =>
      X = [1,1,1,1,1,1,1]
      X = [1,1,1,1,1,2]
      X = [1,1,1,1,3]
      X = [1,1,1,2,2]
      X = [1,1,1,4]
      X = [1,1,2,3]
      X = [1,1,5]
      X = [1,2,2,2]
      X = [1,2,4]
      X = [1,3,3]
      X = [1,6]
      X = [2,2,3]
      X = [2,5]
      X = [3,4]
      no
```

Rozdělení seznamu na dvě části:

```
rozdeleni(Cele,Cast1,Cast2) :-
  append(Cast1,Cast2,Cele).
```

příklad:

```
   rozdeleni([1,2,3,4],X,Y). =>
      X = [], Y = [1,2,3,4]
      X = [1,2,3,4], Y = []
      X = [1], Y = [2,3,4]
      X = [1,2,3,4], Y = []
      X = [1,2], Y = [3,4]
      X = [1,2,3,4], Y = []
      X = [1,2,3], Y = [4]
      X = [1,2,3,4], Y = []
      X = [1,2,3,4], Y = []
      X = [1,2,3,4], Y = []
      X = [1,2,3,4], Y = []
```

Prolog 3-3

Predikáty pro hromadné vyhodnocení

bagof(X, P, L)
- vytvoří se seznam L všech objektů X takových, že je cíl P splněn

?- bagof(X,vyber([1,2,3,4,5,6],7,X),L).
X = _5612 % ukazatel ve vyhodnocovacím prostředí, nevšímáme si
L = [[1,2,4],[1,6],[2,5],[3,4]]

setof - funguje stejně jako bagof, ale eliminuje případné duplikáty výsledků
findall - funguje stejně jako bagof, ale volné proměnné ve vnitřním cíly jsou existenčně kvantifikovány

Třídění seznamů v Prologu

MergeSort:

```
mergesort([],[]).       % triviálně setřídený seznam
mergesort([A],[A]).     % taktéž triviálně setříděný seznam
mergesort(Nesetridene,Setridene) :-
  rozpul(Nesetridene,Levy,Pravy),    % nesetříděný seznam rozdělíme na dvě části
  mergesort(Levy,Levy_setrideny),    % obě části setřídíme
  mergesort(Pravy,Pravy_setrideny),
  merger(Levy_setrideny,Pravy_setrideny,Setridene). % setříděné části spojíme do setříděného celku

rozpul(Seznam,Levy,Pravy) :-
  append(Levy,Pravy,Seznam),   % obě části musí dohromady dát původní seznam
  stejne_dlouhe(Levy,Pravy).     % chceme aby byli stejně dlouhé

stejne_dlouhe([],[]).    % původní seznam byl sudé délky a povedlo se rozdělit ho na ideání poloviny
stejne_dlouhe([],[_]).   % původní seznam byl sudé délky, jedna část je větší o jeden prvek
stejne_dlouhe([_],[]).
stejne_dlouhe([_|Zbytek1],[_|Zbytek2]) :  % odebíráme postupně první prvky
  stejne_dlouhe(Zbytek1,Zbytek2).

merger([],Seznam,Seznam).   % triviální sloučení dvou seznamů
merger(Seznam,[],Seznam).
merger([X|ZbytekX],[Y|ZbytekY],[X|Spojene]) :-   % X bylo menší a dáme ho do výstupu
  X < Y,
  merger(ZbytekX,[Y|ZbytekY],Spojene).
merger([X|ZbytekX],[Y|ZbytekY],[Y|Spojene]) :-   % Y bylo menší a dáme ho do výstupu
  X >= Y,
  merger([X|ZbytekX],ZbytekY,Spojene).
```

příklad:
mergesort([7,2,3,2,4,0,9],X). => X = [0,2,2,3,4,7,9]

Prolog 3-4

QuickSort:

```
quicksort([],[]). % triviální setřízení
quicksort([Pivot|Zbytek],Setridene) :-        % vybereme Pivota a jdem na to
  rozdel(Zbytek,Pivot,Mensi,Vetsi),           % získáme dvě části, v jedné prvky menší než pivot a ve druhé větší
  quicksort(Mensi,Mensi_setridene),           % obě části setřídíme
  quicksort(Vetsi,Vetsi_setridene),
  append(Mensi_setridene,[Pivot|Vetsi_setridene],Setridene).  % a spojíme

rozdel([],_,[],[]).
rozdel([X|Zbytek],Pivot,[X|Mensi],Vetsi) :-  % X zařadíme do Menší části
  X =< Pivot,
  rozdel(Zbytek,Pivot,Mensi,Vetsi).
rozdel([X|Zbytek],Pivot,Mensi,[X|Vetsi]) :- % X zařadíme do Větší části
  X >= Pivot,
  rozdel(Zbytek,Pivot,Mensi,Vetsi).
```

příklad:

quicksort([7,2,3,2,4,0,9],X). => X = [0,2,2,3,4,7,9]