Úloha 1

vytvořte predikát **soucet_plus**(L, N), který do N sečte všechna pozitivní čísla ze seznamu L

Řešení 1

```
soucet_plus([], 0).

soucet_plus([Hlava|Ocas], Vysledek) :-
        Hlava > 0,
        soucet_plus(Ocas, Mezivysledek),
        Vysledek is Mezivysledek + Hlava.

soucet_plus([Hlava|Ocas], Vysledek) :-
        Hlava =< 0,
        soucet_plus(Ocas, Vysledek).
```

Úloha 2

vytvořte predikát **rozdel(L, Cast1, Cast2)**, který rozdělí seznam hodnot L na dvě části tak, aby se jejich počty prvků lišily nanejvýš o 1; na pořadí prvků nezáleží, části nemusí být souvislé

např. rozdel([1, 2, 3, 4, 5, 6, 7], [1, 3, 5, 7], [2, 4, 6])

Řešení 2

```
rozdel([], [], []).

rozdel([A], [A], []).

rozdel([A, B | Ocas], [A|Cast1], [B|Cast2]) :-
    rozdel(Ocas, Cast1, Cast2).
```

Alternativní řešení 2

- odebíráme po jednom prvku a střídáme pořadí částí, aby se prvky přidávaly střídavě oběma

```
rozdel([], [], []).

rozdel([Hlava | Ocas], [Hlava | Cast1], Cast2) :-
  rozdel(Ocas, Cast2, Cast1).
```

Alternativní řešení 2

- s využitím knihovních funkcí

```
r2(L, Cast1, Cast2) :-
   length(L, Delka),
   Polovina is Delka // 2,
   length(Cast1, Polovina),  %vytvoří seznam dané délky
   append(Cast1, Cast2, L).
```

Úloha 3

vytvořte determinovanou verzi predikátu
odstran(co, *odkud*, *vysledek*)
který ze seznamu *odkud* odstraní první výskyt atomu *co*; při pokusu o redo musí selhat

pokračování: vytvořte odstranvse(), který odstraní všechny výskyty; lze realizovat řezem i bez řezu

Řešení 3a s řezem

```
odstran(_, [], []).

odstran(Hlava, [Hlava|Ocas], Ocas) :- !.

odstran(X, [Hlava|Ocas], [Hlava|OcasBez]) :-
        odstran(X, Ocas, OcasBez).
```

Řešení 3a bez řezu

```
odstran(_, [], []).

odstran(Hlava, [Hlava|Ocas], Ocas).

odstran(X, [Hlava|Ocas], [Hlava|OcasBez]) :-
        X \= Hlava,
        odstran(X, Ocas, OcasBez).
```

Řešení 3b s řezem

```
odstranvse(_, [], []).

odstranvse(Hlava, [Hlava|Ocas], Vysledek) :-
        odstranvse(Hlava, Ocas, Vysledek), !.

odstranvse(X, [Hlava|Ocas], [Hlava|OcasBez]) :-
        odstranvse(X, Ocas, OcasBez).
```

Řešení 3b bez řezu

```
odstranvse(_, [], []).

odstranvse(Hlava, [Hlava|Ocas], Vysledek) :-
        odstranvse(Hlava, Ocas, Vysledek).

odstranvse(X, [Hlava|Ocas], [Hlava|OcasBez]) :-
        X \= Hlava,
        odstranvse(X, Ocas, OcasBez).
```

Úloha 4

vytvořte predikát **unikatni**(L), který vydá true, pokud se žádný prvek L neopakuje, fail jinak

měl by být determinován

Řešení 4

```
unikatni([]).

unikatni([Hlava|Ocas]) :-
        \+ member(Hlava, Ocas),
        unikatni(Ocas).
```

Úloha 5

vytvořte predikát **linearizuj**(L1, L2), kde L2 je lienarizovaná verze L1 – bez vnořených seznamů, všechny prvky z nich jsou na stejné úrovni

tip: pro spojení seznamů použijte merge

linearizuj([a, [b, [c], d], [e, f]], [a, b, c, d, e, f]).

Řešení 5

```
linearizuj([], []).

linearizuj([Hlava|Ocas], [Hlava|LinOcas]) :-
        \+ is_list(Hlava),
        linearizuj(Ocas, LinOcas).

linearizuj([Hlava|Ocas], Vysledek) :-
        is_list(Hlava),
        linearizuj(Hlava, LinHlava),
        linearizuj(Ocas, LinOcas),
        merge(LinHlava, LinOcas, Vysledek).
```