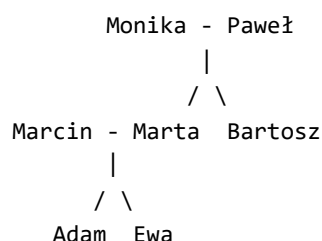


# Zadanie 1

Wykorzystując predykaty:


- ojciec(X,Y) - znaczenie: X jest ojcem Y
- matka(X,Y) - znaczenie: X jest matką Y
- mezczyzna(X) - znaczenie: X jest mężczyzną
- kobieta(X) - znaczenie: X jest kobietą Zdefiniuj predykaty:
- rodzic(X,Y) - znaczenie: X jest rodzicem Y
- rodzenstwo(X,Y) - znaczenie: X i Y są rodzeństwem
- siostra(X,Y) - znaczenie: X jest siostrą Y
- brat(X,Y) - znaczenie: X jest bratem Y
- dziadek(X,Y) - znaczenie: X jest dziadkiem Y
- babcia(X,Y) - znaczenie: X jest babcią Y
- jest\_ojcem(X) - znaczenie: X jest ojcem
- jest\_matka(X) - znaczenie: X jest matką




```

1 :-style_check(-discontiguous).
2
3 kobieta(monika).
4
5 mezczyzna(pawel).
6
7 kobieta(marta).
8
9 kobieta(ewa).
10
11 mezczyzna(marcin).
12
13 mezczyzna(bartosz).
14
15 mezczyzna(adam).
16
17 matka(monika,marta).
18
19 matka(monika,bartosz).
20
21 ojciec(pawel,marta).
22
23 ojciec(pawel,bartosz).
24
25 ojciec(marcin,adam).
26
27 ojciec(marcin,ewa).
  
```


28

 ?- rodzenstwo(marta,X).



X = marta  
 X = bartosz  
 X = marta  
 X = bartosz  
**false**

 ?- rodzic(X,marta).


X = pawel  
 X = monika

 ?- rodzic(marta,X).


X = adam  
 X = ewa

 ?- jest\_ojcem(adam).


**false**

 ?- jest\_ojcem(pawel).



**true**  
**true**

1

2

 ?- dziadek(X,adam).


X = pawel  
**false**

 ?- babcia(X,Y).


X = monika,  
 Y = adam  
 X = monika,  
 Y = ewa  
**false**

## Zadanie 2

Wykorzystując predykat: `wiek(X,Y)` (znaczenie: wiek X to Y) zdefiniuj predykaty:

- `rowiesnik(X,Y)`
- `pelnoletni(X)`
- `emeryt(X)`
- `starszy0(X,Y,Z)` - znaczenie: X jest starszy od Y o Z lat. („operator przypisania” po prawej stronie reguły to is)

```
1 rowiesnik(X,Y) :- wiek(X,WX),wiek(Y,WY),WX = WY.
2
3 pelnoletni(X) :- wiek(X,WX),WX>18.
4
5 starszy0(X,Y,Z):- wiek(X,WX),wiek(Y,WY),Z is (WX-WY).
6
```



```

7 wiek(adam, 15).
8 wiek(pawel, 15).
9 wiek(ula, 70).

```

```
≡ ?- rowiesnik(adam,pawel).
```



```
true
```

```
1
```

```
≡ ?- rowiesnik(adam,X).
```



```
X = adam
```

```
X = pawel
```

```
false
```

```
≡ ?- pelnoletni(ula).
```



```
true
```

```
1
```

```
≡ ?- pelnoletni(adam).
```



```
false
```

```
≡ ?- starszy0(ula,adam,X).
```



```
X = 55
```

## Zadanie 3

Zdefiniuj predykaty:

- `nalezy(X,L)` –spełniony jeżeli Xjest elementem listy L.
- `usun(X,L)` -elementX jest usunięty z listy L.

```

1 nalezy(_,[]) :- false.
2 nalezy(X, [Y|Yogon]) :- X=Y;nalezy(X,Yogon).
3
4 usun(X,[X|L1], L1).
5 usun(X, [Y|L2], [Y|L1]) :- usun(X,L2,L1).

```



```
≡ ?- nalezy(b, [a,b,c]).
```



```
true
```

```
false
```

```
1
```

```
≡ ?- nalezy(m, [a,b,c]).
```



```
false
```

```
≡ ?- usun(b, [a,b,c], X).
```



```
X = [a, c]
```

```
false
```

```
≡ ?- usun(b, [a,b,b,a,c], X).
```



```
X = [a, b, a, c]
```

**X** = [a, b, a, c]  
**false**

## Zadanie 4

Rozważmy reprezentacje zbiorów za pomocą list. Wykorzystując predykaty z Zadania 3 zdefiniuj następujące predykaty:

- `element(X,L)` -X jest elementem zbioru L.
- `podzbior(L,K)` -L jest podzbiorem zbioru K.
- `rozlaczny(L,K)` -L jest rozłączny ze zbiorem K.
- `suma(L,K,M)` -M jest sumą zbiorów L i K.
- `przeciecie(L,K,M)` -M jest przecięciem zbiorów L i K.
- `roznica(L,K,M)` -M=L\K

```

1 element(X,L) :- налеzy(X,L).
2
3 podzbior([],_).
4
5 podzbior([LH|LT],K) :- налеzy(LH,K), podzbior(LT,K).
6
7 rozlaczny([],_).
8
9 rozlaczny([LH|LT],K) :- not(налеzy(LH,K)), rozlaczny(LT,K).
10
11 dodaj(X,L1,[X|L1]).
12
13 suma([],L,L).
14
15 suma([H|T1],L2,[H|T2]) :- suma(T1,L2,T2).
16
17
18 przeciecie([], _, []).
19
20 przeciecie([LH|LT], K, [LH|OUT]) :- налеzy(LH, K), przeciecie(LT, K, OUT).
21
22 przeciecie([_|TL], L2, OUT) :- przeciecie(TL, L2, OUT).
23
24 usun_wszystkie(L,[],L).
25 usun_wszystkie(L,[X|Sub],Rem) :- usun(X,L,Rem0), usun_wszystkie(Rem0,Sub,Rem).
26
27 roznica(L,K,OUT) :- przeciecie(L,K,PRZE),!,suma(L, K, SUM),usun_wszystkie(SUM, P
28

```

≡ ?- suma([a,b],[c,d],X).



**X** = [a, b, c, d]

≡ ?- element(a,[a,c,d]).



**true**  
**false**

1

≡ ?-



```
podzbior([a,b],[a,b,c]).
```

**true**

1

**false**

```
≡ ?- podzbior([a,p],[a,b,c]).
```

**false**

```
≡ ?- podzbior(X,[a,b,c]).
```

**X = []****X = [a]****X = [a, a]****X = [a, a, a]****X = [a, a, a, a]****X = [a, a, a, a, a]****X = [a, a, a, a, a, a]****X = [a, a, a, a, a, a, a]**

```
≡ ?- rozlaczny([1,p],[a,b,c]).
```

**true**

1

```
≡ ?- rozlaczny([b,p],[a,b,c]).
```

**false**

```
≡ ?- suma([a,b,c],[b,c,d],X).
```

**X = [a, b, c, b, c, d]**

```
≡ ?- nalezy(_, [a,b]).
```

**true**

1

**true**

2

**false**

```
≡ ?- suma([a,b,c],[b,c,d],X).
```

**X = [a, b, c, b, c, d]**

```
≡ ?- przeciecie([a,b,c,d],[g,b,c,h,j,l],X).
```

**X = [b, c]****X = [b]****X = [c]****X = []**

```
≡ ?- przeciecie([a,b,c,d],[a,b,e,f],X), !.
```

**X = [a, b]**

```
≡ ?- usun_wszystkie([a,b,c,d,e],[a,c,e],X).
```

**X = [b, d]****false**

≡ ?- roznica([a,b,c,d],[a,b,f],X).

X = [c, d, a, b, f]

## Zadanie 5

Zdefiniuj predykaty:

- ostatni(X,L) -zmienna X jest ukonkretniona ostatnim elementem listy L.
- suma(L,N) oznaczający, że suma liczb całkowitych będących na liście L wynosi N.
- predykat wyst\_poz(X,L,N) oznaczający, że element X występuje w liście L na pozycji N.
- predykat wyst\_ile(X,L,N) oznaczający, że element X występuje w liście L N razy.
- max(L,N) oznaczający, że elementem maksymalnym na liście L jest N.
- znajdz(N,L,X) –zmienna X jest ukonkretniona N-tym elementem listy L.
- zastap(X,L1,Y,L2) pozwalający zastąpić element X na liście L1 elementem Y. Wynikiem zamiany jest lista L2.

```

1 ostatni(X,[_,X]).
2 ostatni(X,[_|LT]) :- ostatni(X,LT).
3
4 suma_num3([],0).
5 suma_num3([G|O],M) :- suma_num3(O,N),0 is mod(G,2),M is N+G.
6 suma_num3([G|O],M) :- suma_num3(O,N),not(0 is mod(G,2)),M is N.
7
8 wyst_poz(X,[X|_],N,NCurr):- N=NCurr.
9 wyst_poz(X,[_|LT],N,NCurr):- NewCurr is NCurr+1, wyst_poz(X,LT,N,NewCurr).
10 wyst_poz(X,[LH|LT],N):- wyst_poz(X,[LH|LT],N,0).
11
12 wyst_ile(_,[],0).
13 wyst_ile(X,[X|LT],N) :- wyst_ile(X,LT,C),N is C+1,!.
14 wyst_ile(X,[_|LT],N) :- wyst_ile(X,LT,C),N is C,!.
15
16 max([],0).
17 max([LH|LT],N) :- max(LT,MX),MX>LH,N is MX.
18 max([LH|LT],N) :- max(LT,MX),MX <= LH,N is LH.
19
20 znajdz(N,[X|_],X,N).
21 znajdz(N,[_|LT],X,NCurr) :- NP is Ncurr+1, znajdz(N,LT,X,NP).
22 znajdz(N,L,X) :- znajdz(N,L,X,0).
23
24
25 zastap(_, [], _, []).
26 zastap(X, [X|L1T], Y, [Y|L1T]).
27 zastap(X, [L1H|L1T], Y, [L1H|L2T]):-zastap(X, L1T, Y, L2T).
```

≡ ?- ostatni(X,[a,b,c,d]).

X = d  
false

≡ ?- suma\_num3([1,2,2,3,4],L).

L = 8  
false

≡

```
?- suma_num3([1,2,2,1,1,3,3,5,7],L).
```

```
L = 4
```

```
false
```

```
≡ ?- wyst_poz(a,[b,a,c],1).
```

```
true
```

```
false
```

1

Create a

Program

Query

Markdown

HTML

cell here

```
≡ ?- wyst_poz(a,[a,b,c],1)
```

```
false
```

```
≡ ?- wyst_ile(a,[b,a,c,a,a,1,1,a,a],X).
```

```
X = 5
```

```
≡ ?- wyst_ile(g,[b,a,c,a,a,1,1,a,a],X).
```

```
X = 0
```

```
≡ ?- max([1,2,3,4],X).
```

```
X = 4
```

```
false
```

```
≡ ?- znajdz(2,[a,b,c,d,e,f],X)
```

```
X = c
```

```
false
```

```
≡ ?- zastap(a,[b,c,a,d],z,X).
```

```
X = [b, c, z, d]
```

```
X = [b, c, a, d]
```

```
false
```

## Zadanie 6

Zdefiniuj predykat `wypiszListe(L)` wypisujący elementy listy w następujący sposób:

- Każdy element listy powinien być wypisany w oddzielnej linii.
- Jeżeli element listy `L` jest listą `M` wówczas elementy listy `M` wypisywane są z „wcięciem”.

```
1 wypiszListe([],_).
2 wypiszListe([LH|LT], OFFSET) :- not(nalezy(_,LT)),nl,write(OFFSET),write(LH),wypiszListe(LT,OFFSET).
3 wypiszListe([LH|LT], OFFSET) :- nalezy(_,LT),nl,write(OFFSET),write(LH),wypiszListe(LT,OFFSET).
4 wypiszListe(L) :- wypiszListe(L,' ').
```

```
≡ ?- wypiszListe(['a','b','c']).
```

```
a
```

```
b
```

```
c
```

**true**

1

≡ ?- wypiszListe(['a','b','c', ['c','d'] ]).



a  
b  
c  
[c, d]  
**true**

1