```
rodzic(a,b).
rodzic(a,c).
rodzic(a,d).
rodzic(b,e).
rodzic(b,f).
rodzic(c,g).
rodzic(c,h).
rodzic(c,i).
rodzic(d,j).
rodzic(f,k).
rodzic(f,l).
```

Zadanie 1

Zdefiniuj następujące predykaty:

1. `rodzenstwo(A,B)` – element A jest rodzeństwem B
2. `tenSamPoziom(A,B)` – element A jest na tym samym poziomie co element B
3. `poziom(X,N)` – element X jest na poziomie N (korzeń – poziom 0).
4. `przodek(X,Y)` – element X jest przodkiem* Y.
5. `wspPrzodek(X,Y,Z)` – element Z jest wspólnym przodkiem X i Y.
6. `doPrzodka(X,Y)` – wypisujemy wszystkie wierzchołki od X do przodka Y.
7. `odPrzodka(X,Y)` – wypisujemy wszystkie wierzchołki od przodka Y od X.
8. `sciezka(X)` – wypisana jest droga pozwalająca dotrzeć od korzenia do elementu X.
9. `sciezka(X,Y)` – wypisana jest droga między elementem X i Y
10. `sciezkaMin(X,Y)` – wypisana jest najkrótsza droga między wierzchołkami X i Y.
11. `doKorzeniaL(X,L)` – L to tablica wierzchołków przez które trzeba przejść aby dotrzeć do korzenia.
12. `sciezkaL(X,Y,L)` – L to tablica wierzchołków przez które trzeba przejść między wierzchołkami X i Y. *przodek to wierzchołek znajdujący się na wyższym poziomie drzewa.

```
rodzenstwo(A,B) :- rodzic(C,A),rodzic(C,B),A\==B.

ten_sam_poziom(X,X).
ten_sam_poziom(X,Y):-rodzic(A,X),rodzic(B,Y),X\==Y,ten_sam_poziom(A,B).

poziom(X,0):-not(rodzic(_,X)),!.
poziom(X,N):-rodzic(Y,X),poziom(Y,M),N is M+1.

przodek(X,Y):-rodzic(X,Y).
przodek(X,Y):-rodzic(RodzicY,Y),przodek(X,RodzicY).

wspPrzodek(X,Y,Z) :- przodek(Z,X),przodek(Z,Y).
```

```
?- przodek(X,f).
```

```
X = b
X = a
false
```

```
?- wspPrzodek(e,k,X).
```

```
X = b
X = a
false
```

```
doPrzodka(X,X) :- write(X).
doPrzodka(X,Y) :- write(X),write('->'),rodzic(RodzicX,X),doPrzodka(RodzicX,Y),!.
```

Create a Program Query Markdown HTML cell here

```
?- doPrzodka(k,a).
```

```
k->f->b->a
true
```

1

```
odPrzodka(X,X) :- write(X).
odPrzodka(X,Y) :- rodzic(RodzicX,X),odPrzodka(RodzicX,Y),write('->'),write(X).
```

```
?- odPrzodka(k,a).
```

```
a->b->f->k
true
false
```

1

```
doPrzodkaL(X,X,[X]):-write(X).
doPrzodkaL(X,Y,[X|L]):-write(X),
                write('->'),
                rodzic(R,X),
```

```
5                   doPrzodkaL(R,Y,L),!.
```

```
?- doPrzodkaL(k,a,X).
```

k->f->b->a
X = [k, f, b, a]

Create a Program | Query | Markdown | HTML **cell here**

```
1 sciezka(X):-poziom(X,0).
2 sciezka(X):-rodzic(A,X),sciezka(A),write('->'),write(A),!.
```

```
?- sciezka(k).
```

->a->b->f
true *1*

```
?- sciezka(f).
```

->a->b
true *1*

```
1 sciezka(X,Y):-wspPrzodek(X,Y,R),rodzic(R,R1),przodek(R1,Y),not(R==R1),doPrzodka(X,R),write('->'),odPrzodka(Y,R1),!.
```

```
?- sciezka(e,j)
```

e->b->a->d->j
true *1*

```
?- sciezka(e,l).
```

e->b->f->l
true *1*

```
1 sciezkaMin(X,Y):sciezka(X,Y).
```

```
1 doKorzeniaL(X,L):-poziom(YC,1),rodzic(Y,YC),doPrzodkaL(X,Y,L),!.
```

```
?- doKorzeniaL(k,L).
```

k->f->b->a
L = [k, f, b, a]

```
1 suma([],L,L).
2 suma([H|T1],L2,[H|T2]):-suma(T1,L2,T2).
3 odPrzodkaL(X,X,[X]) :- write(X).
4 odPrzodkaL(X,Y,L) :- rodzic(RodzicX,X),odPrzodkaL(RodzicX,Y,L1),write('->'),write(X),!, suma(L1,[X],L).
5 sciezkaL(X,Y,L):-wspPrzodek(X,Y,R),rodzic(R,R1),przodek(R1,Y),not(R==R1),doPrzodkaL(X,R,L1),write('->'),odPrzodkaL(Y,R1,L2)
```

```
?- sciezkaL(e,j,L)
```

e->b->a->d->j
L = [e, b, a, d, j]

```
?- odPrzodkaL(k,a,L).
```

a->b->f->k
L = [a, b, f, k]