

Overview

Imperative Languages

- sequence of instructions, executed after each other

■ Procedural Languages

- variables, assignments, control structures

■ Object-Oriented Languages

- objects and classes
- ADT and inheritance

Declarative Languages

- specify *what* should be computed
- compiler determines *how* the computation works

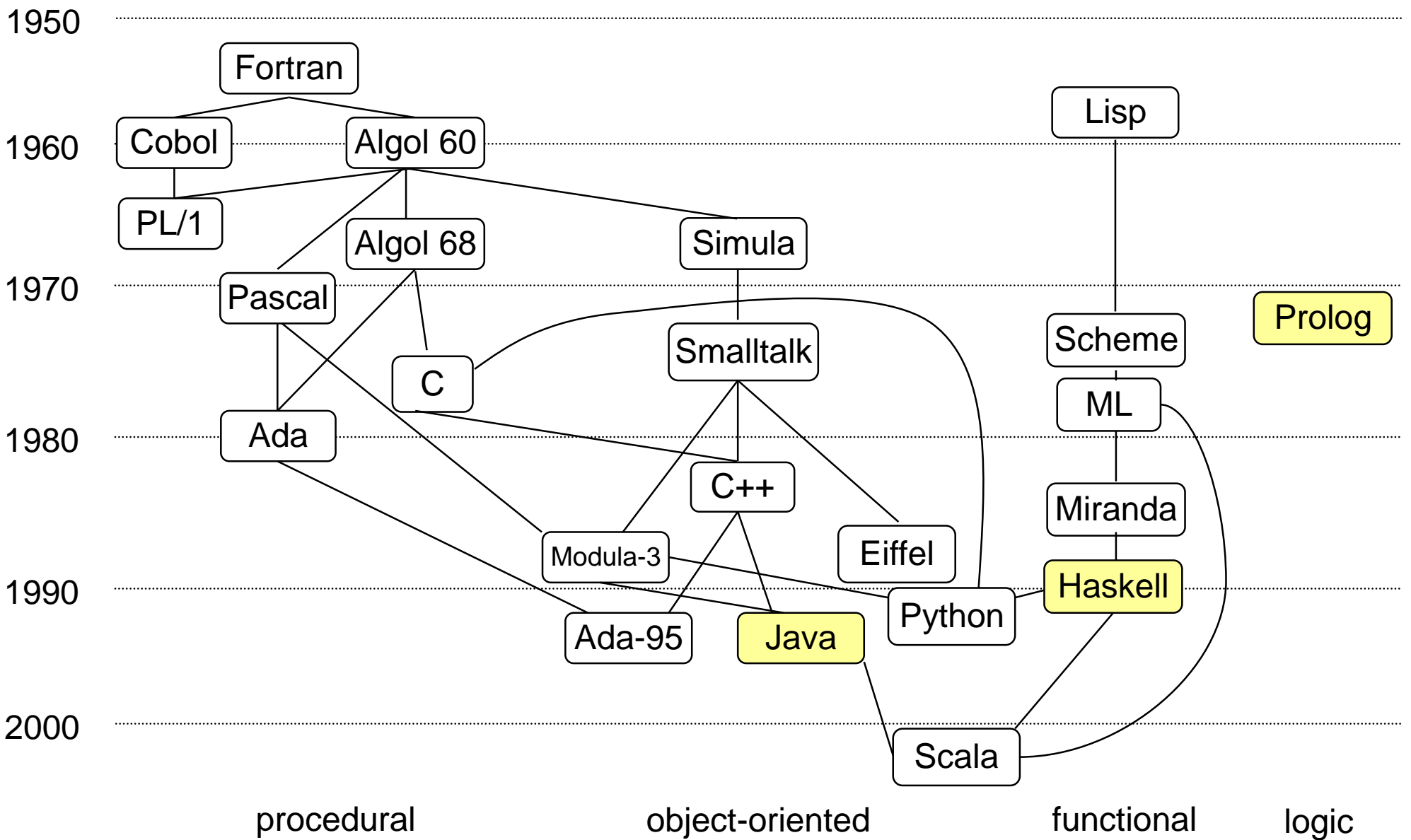
■ Functional Languages

- no side-effects
- recursion

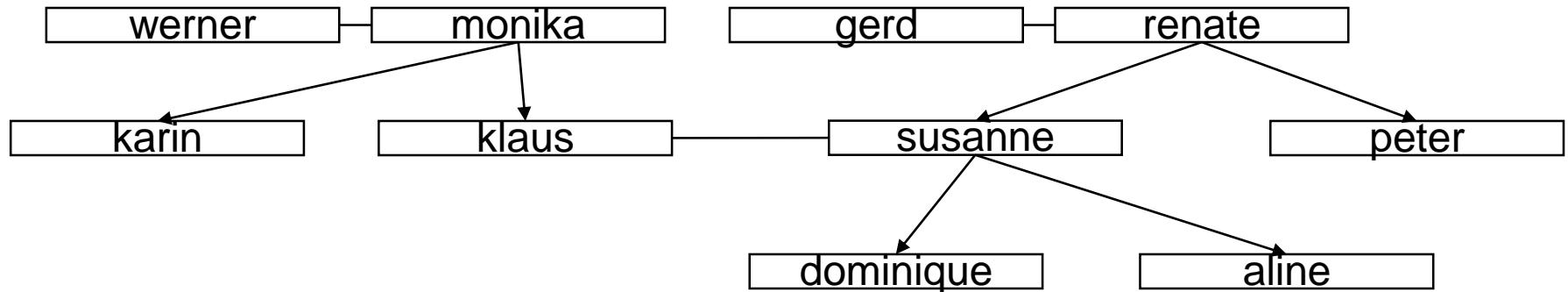
■ Logic Languages

- rules to define relations

Important Programming Languages



Facts and Queries



Program:

```
female(monika).
female(karin).
female(renate).
female(susanne).
female(aline).

male(werner).
male(klaus).
male(gerd).
male(peter).
male(dominique).

married(werner, monika).
married(gerd, renafe).
married(klaus, susanne).

human(X).

motherOf(monika, karin).
motherOf(monika, klaus).
motherOf(renate, susanne).
motherOf(renate, peter).
motherOf(susanne, aline).
motherOf(susanne, dominique).
```

?- male(gerd).

true

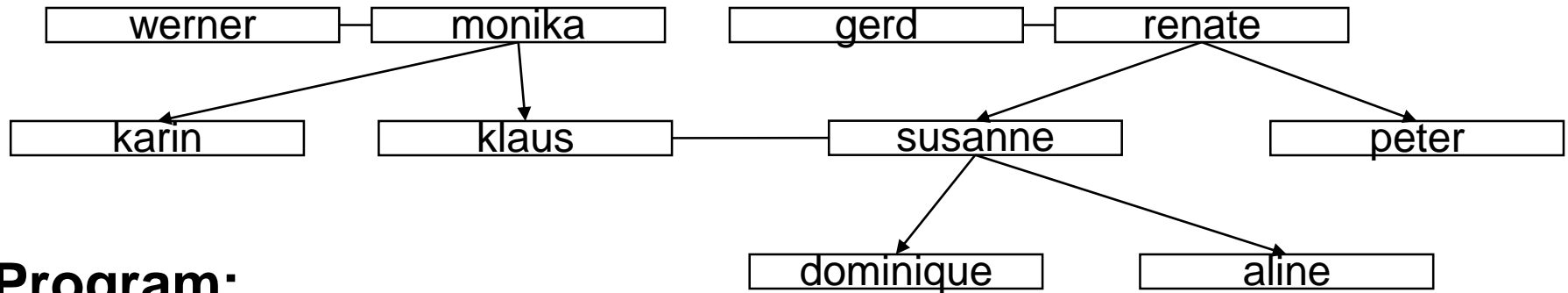
?- married(gerd, monika).

false

?- human(gerd).

true

Variables in Queries



Program:

```
female(monika).
```

```
female(aline).
```

```
married(werner, monika).
```

```
married(klaus, susanne).
```

```
male(werner).
```

```
male(dominique).
```

```
motherOf(monika, karin).
```

```
motherOf(susanne, dominique).
```

```
?- motherOf(X, susanne).
```

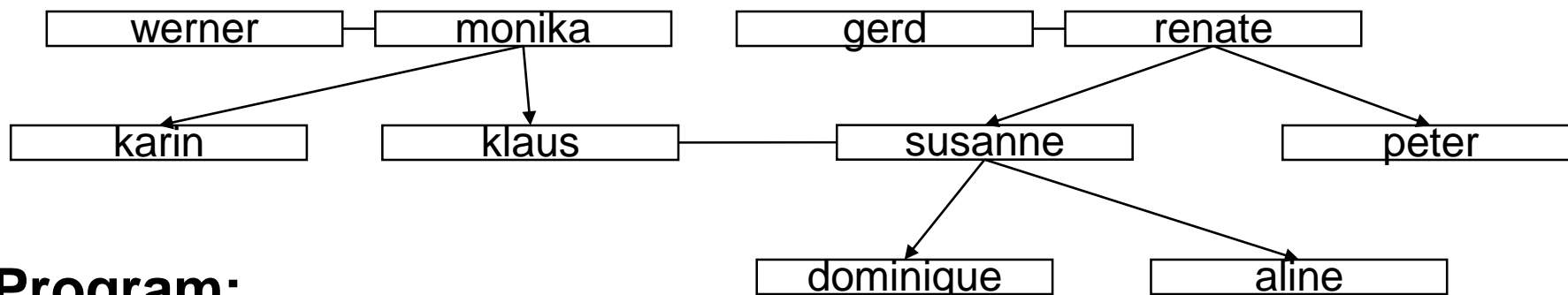
```
X = reate
```

```
?- motherOf(renate, Y).
```

```
Y = susanne ;
```

```
Y = peter
```

Combined Queries



Program:

```
female(monika).
```

```
female(aline).
```

```
married(werner, monika).
```

```
married(klaus, susanne).
```

```
male(werner).
```

```
male(dominique).
```

```
motherOf(monika, karin).
```

```
motherOf(susanne, dominique).
```

```
?- married(gerd,W), motherOf(W,susanne).
```

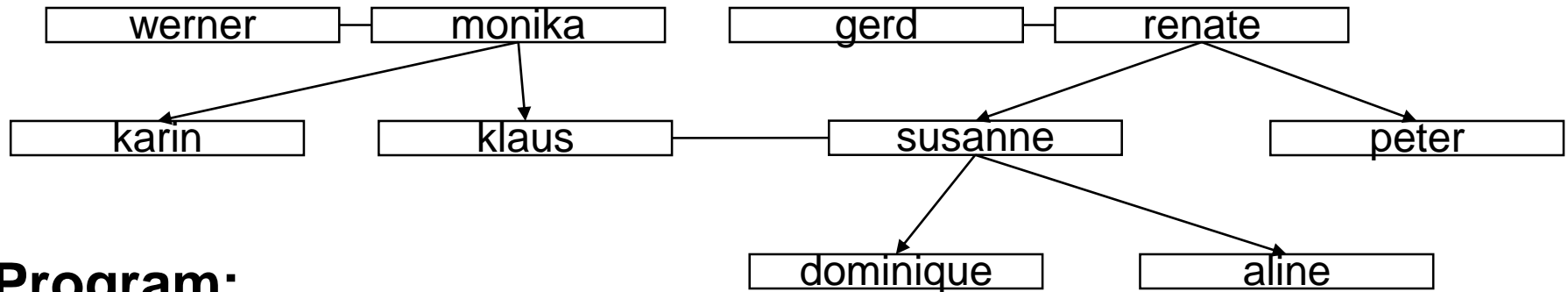
```
W = renafe
```

```
?- motherOf(Grandma,Mom), motherOf(Mom,aline).
```

```
Grandma = renafe
```

```
Mom = susanne
```

Rules



Program:

```
female(monika).
female(aline).
married(werner, monika).
married(klaus, susanne).

male(werner).
male(dominique).
motherOf(monika, karin).
motherOf(susanne, dominique).

fatherOf(F,C) :- married(F,W), motherOf(W,C).
```

?- fatherOf(gerd, susanne).

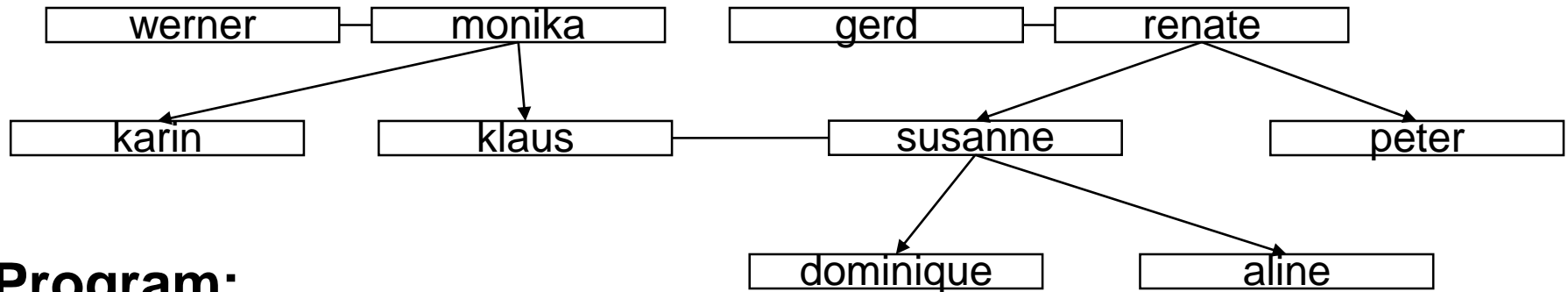
true

?- fatherOf(gerd, Y).

Y = susanne ;

Y = peter

Several Rules for one Predicate



Program:

```
female(monika).
female(äline).
married(werner, monika).
married(klaus, susanne).
fatherOf(F,C) :- married(F,W), motherOf(W,C).

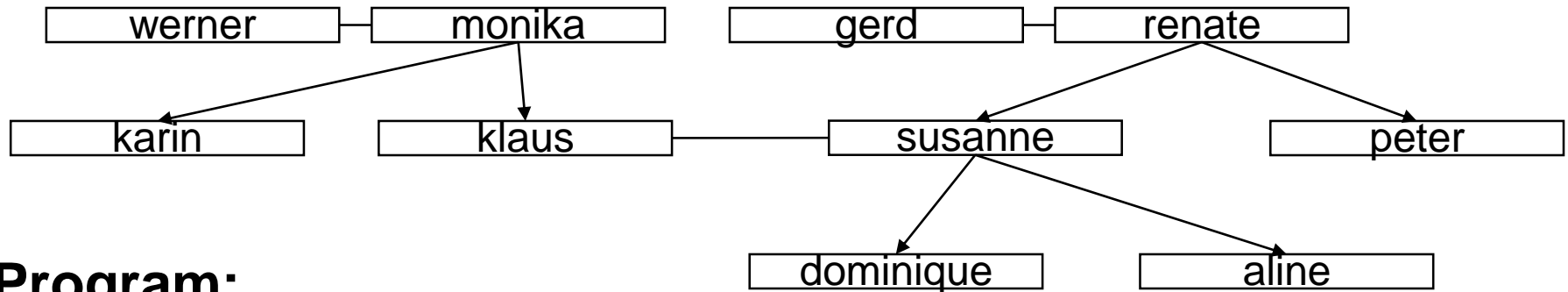
parent(X, Y) :- motherOf(X,Y).
parent(X, Y) :- fatherOf(X,Y).
```

```
?- parent(X, susanne).
```

```
X = reate ;
```

```
X = gerd
```

Recursive Rules



Program:

```
female(monika).           male(werner).
married(klaus, susanne).  motherOf(susanne, dominique).
fatherOf(F,C) :- married(F,W), motherOf(W,C).
parent(X, Y) :- motherOf(X,Y).
parent(X, Y) :- fatherOf(X,Y).

ancestor(V,X) :- parent(V,X).
ancestor(V,X) :- parent(V,Y), ancestor(Y,X).
```

```
?- ancestor(X, aline).
```

```
X = susanne ; X = klaus ;
```

```
X = monika ; X = renafe ; X = werner ; X = gerd
```