

## Why Logic?

**Knights and Liars:** Knights always tell the truth; Liars always lie.

Zoe: “Mel is a liar”

Mel: “Neither I nor Zoe are liars”

Who’s lying?

## Why!? Logic!!

Zoe: “Mel is a liar”

Mel: “Neither I nor Zoe are liars”

Assume **Mel** is a knight.

Then what **Zoe** said is false;

Hence **Zoe** has to be a liar;

But **Mel** said she is not!

This is a contradiction; so **Mel** has to be a liar, and, consequently, **Zoe** a knight.

## Why Programming?

*It was a dark and stormy night. Three men, Alex, Bob & Carl, taking refuge from the rains, came to a hotel, one after another. One was a “knight”: he always spoke the truth; another was a “liar”: he always lied; and the third was a “knave”: he alternated lying with speaking the truth.*

*When they arrived at the hotel, the manager was not at the desk. When she came to the front, she said she had only one room available, and will give it to the person who arrived first; the other two have to make do with the chairs in the lobby.*

...

## Why Programming?

(Contd.)

...

*The conversation went like this:*

*Alex: I came first.*

*Bob: No, he did not! I came first.*

*Carl: No, he did not! I came first.*

*Alex: No, he did not! I came first.*

*Carl: Well, Bob did not come second.*

*Bob: That's true!*

Who came first? Who came second? Who came third?

# Logic and Programming

- Logic forms a formal foundation for describing relationships between entities.
- In many cases, we can infer interesting consequences from these relationships.
- The procedure used for inference *turns descriptions into programs*
- The same set of relationships can be described in many ways: each resulting in a different “program”.
- Logic Programming: a framework for describing relationships such that inferences can be done *efficiently*.

## Relations

- `parent(X, Y)`: X is a parent of Y.

```
parent(pam, bob).      parent(bob, ann).
parent(tom, bob).     parent(bob, pat).
parent(tom, liz).     parent(pat, jim).
```

- `male(X)`: X is a male.  
`female(X)`: X is a female.

```
female(pam).          male(tom).
female(pat).          male(bob).
female(ann).          male(jim).
female(liz).
```

- `mother(X, Y)`: X is the mother of Y.

$$\forall X, Y. \text{parent}(X, Y) \wedge \text{female}(X) \Rightarrow \text{mother}(X, Y)$$

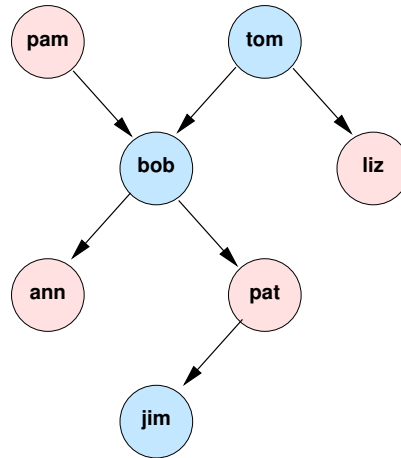
- **In Prolog:** `mother(X,Y) :- parent(X,Y), female(X).`

# Representing relations

```
parent(pam, bob).
parent(tom, bob).
parent(tom, liz).
parent(bob, ann).
parent(bob, pat).
parent(pat, jim).

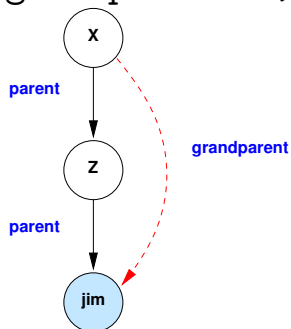
female(pam).
female(pat).
female(ann).
female(liz).

male(tom).
male(bob).
male(jim).
```

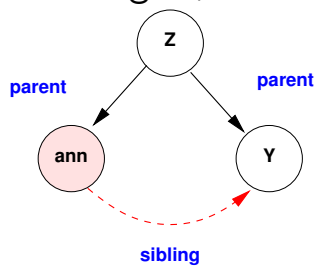


## More Relations

- $\text{grandparent}(X, Y) \text{ :- } \text{parent}(X, Z), \text{parent}(Z, Y).$



- $\text{sibling}(X, Y) \text{ :- } \text{parent}(Z, X), \text{parent}(Z, Y), X \neq Y.$



## More Relations

- `cousin(X,Y) :- .....`
- `greatgrandparent(X,Y) :- .....`
- `greatgreatgrandparent(X,Y) :- .....`
- `ancestor(X,Y) :- ...`