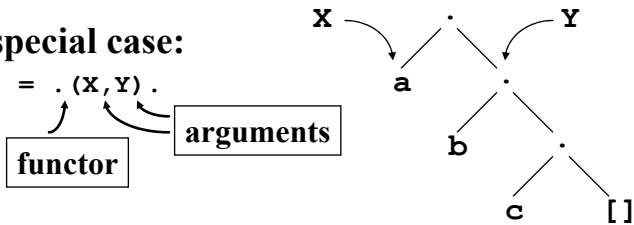


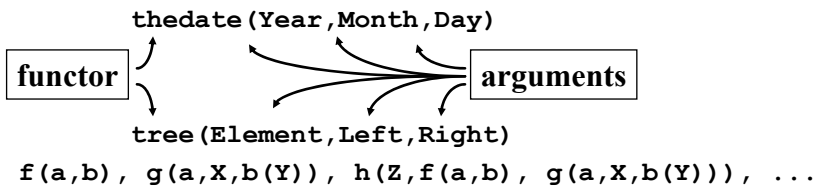
# Data Structures: Terms

- Lists are a special case:

```
| ?- [a,b,c] = .(X,Y).
X = a,
Y = [b,c] ?
```



- Syntax for arbitrary terms:



1

# Predicates on Terms

- Accessing fields of a record:

```
datehas(year, thedate(Y,_,_), Y).
datehas(month, thedate(_,M,_), M).
datehas(day, thedate(_,_,D), D).
| ?- A=thedata(2004,4,14), datehas(year, A, Y).
D = thedate(2004, 4, 14)
Y = 2004 ;
| ?- A=thedata(2004,4,D), datehas(year,A,Y),
datehas(month,A,M), datehas(day,A,7).
A = thedate(2004,4,7), D = 7, M = 4, Y = 2004 ?
```

- Standard order on terms:

```
| ?- thedate(2004,3,25) @< thedate(2004,4,7).
yes
| ?- thedate(2004,3,25) @> thedate(2004,4,7).
no
```

2

# Unification

- A *substitution*,  $\sigma$ , is a finite map from variables to terms, for example:

$\sigma$ :  $A=a$ ,  $B=[b]$ ,  $Y=[c]$ ,  $W=[a|Z]$

- A term  $U$  is an *instance* of another term  $T$  if there is a substitution  $\sigma$  such that  $T\sigma = U$ , for example:

$\text{append}([A|B], Y, [A|Z])\sigma = \text{append}([a,b], [c], [a|Z])$

$\text{append}([a,b], [c], W)\sigma = \text{append}([a,b], [c], [a|Z])$

- Two terms  $S$  and  $T$  are *unifiable* if there exists a substitution  $\sigma$  such that  $S\sigma = T\sigma = U$ , in which case we say that  $\sigma$  is a *unifier* of  $S$  and  $T$ .

3

# Unification

- There may be more than one unifier. For example:

$\sigma_1$ :  $Z=4$ ,  $Y=\text{plus}(3,5)$ ,  $W=\text{times}(\text{plus}(3,5),7)$

$\sigma_2$ :  $Z=4$ ,  $W=\text{times}(Y,7)$

are both unifiers for

$S = \text{times}(Z, \text{times}(Y, 7))$  and  $T = \text{times}(4, W)$

but  $\sigma_2$  is *more general* than  $\sigma_1$ , since

$S\sigma_1 = T\sigma_1 = \text{times}(4, \text{times}(\text{plus}(3,5), 7))$

$S\sigma_2 = T\sigma_2 = \text{times}(4, \text{times}(Y, 7))$

and thus  $S\sigma_1$  can be obtained from  $S\sigma_2$  by applying the further substitution  $\delta$ :  $Y=\text{plus}(3,5)$ .

4

# Unification

- **Theorem:** There exists a *most general unifier* for every pair of unifiable terms S and T.

- **And this is what Prolog computes:**

```
| ?- append([A|B],Y,[A|Z]) = append([a,b],[c],W).  
A = a, B = [b], W = [a|Z], Y = [c] ?
```

```
| ?- times(Z,times(Y,7)) = times(4,W).  
W = times(Y,7), Z = 4 ?
```

5

# List minimum

```
findmin([], Sofar, Sofar).
```

```
findmin([H|T], Sofar, M) :-  
    H < Sofar, findmin(T, H, M).
```

```
findmin([H|T], Sofar, M) :-  
    H >= Sofar, findmin(T, Sofar, M).
```

```
min([H|T], M) :-  
    findmin(T, H, M).
```

```
?- min([5,1,4,2],X).  
X = 1 ;
```

6

# Extra-Logical Predicates

- **Arithmetic:**

| ?- X is 2+3.

evaluated

X = 5 ? yes

not evaluated (it's just a term)

| ?- X is 2+3, X = 2+3.

no

| ?- X is 2+3, Y is 4\*X.

X = 5, Y = 20 ? yes

| ?- X is 2+3, Y is 4\*X, X > Y.

no