

### Μεταβλητή Κλήση

- Η μεταβλητή κλήση μας δίνει τη δυνατότητα να τοποθετήσουμε μια μεταβλητή στη θέση μιας κλήσης στο σώμα ενός κανόνα ή ερώτησης.
- Η μεταβλητή αυτή παίρνει τιμή και εκτελείται κατά τη στιγμή της εκτέλεσης.
- Η ιδιότητα αυτή προκύπτει από την ομοιότητα που παρουσιάζουν οι σύνθετοι όροι (terms) και οι ατομικοί τύποι (atomic formula), στη σύνταξή τους.
- Η διαφορά τους ουσιαστικά έγκειται στη θέση τους μέσα στη πρόταση, καθώς και στην ερμηνεία τους :
  - Ο ατομικός τύπος παριστάνει μια σχέση μεταξύ των όρων του, που ισχύει ή δεν ισχύει ( δηλ. true ή false)
  - Ο σύνθετος όρος παριστάνει μια συνάρτηση των όρων του, χωρίς τιμή αλήθειας.

Π.χ. **father(nick)** .

- Ατομικός τύπος: Ο Nick είναι πατέρας. **true** = ισχύει, **false** = δεν ισχύει
- Σύνθετος τύπος: Ο πατέρας του Nick. (δεν συνδέουμε κάποια τιμή αλήθειας).

35

### Παράδειγμα μεταβλητής κλήσης

**b(3)** .

**a(X) :- X.**

ερώτηση (π.χ.)

?- **a(b(3))** .

**Yes**

?- **a(b(K))** .

**K = 3**

- Η ερώτηση ανάγεται στην ?- **b(3)** . , με αποτέλεσμα **true** ή **false**.

ο Δηλαδή ο σύνθετος όρος γίνεται ατομικός τύπος.

- Σε πολλές εκδόσεις της Prolog υποστηρίζεται η διαδικασία **call(X)** η οποία κάνει την ίδια δουλειά με την σκέτη μεταβλητή.

**a(X) :- call(X)** .

- Ποιο είναι το αποτέλεσμα του προγράμματος, αν δώσουμε σαν είσοδο: **a(K)** .

**a(5)** .

?- **read(X)** , **X** .

Απάντηση: **X=a(5)**

36

### Παράδειγμα μεταβλητής κλήσης

- Να γραφεί πρόγραμμα το οποίο να διαβάζει εντολές από το πληκτρολόγιο (**add, sub,...**) και να τις εκτελεί (διερμηνέας εντολών). Το πρόγραμμα θα σταματά όταν θα διαβάσει μια συγκεκριμένη εντολή (π.χ. **stop**) ή μια εντολή που δεν υπάρχει.

**add(X,Y) :- Z is X+Y, write('Result = '), write(Z), nl.**

**sub(X,Y) :- Z is X-Y, write('Result = '), write(Z), nl.**

**div(X,Y) :- Z is X/Y, write('Result = '), write(Z), nl.**

**mul(X,Y) :- Z is X\*Y, write('Result = '), write(Z), nl.**

**run :- write('Entoli = '), read(E), E, nl, run.**

**run.** (ή **run:- write ('End of Program'), nl.**)

π.χ. ?-**run.**

**Entoli =** (έστω ότι πληκτρολογούμε :) **add(5,7)** .

**Result = 12**

37

### Προηγούμενο παράδειγμα χωρίς μεταβλητή κλήση

**run :-**

**write('Entoli = '), read(E),**

**write('First number: '), read(X), write('Second number: '), read(Y),**

**operation(E,X,Y),**

**nl, run.**

**run.**

**operation(add,X,Y) :- Z is X+Y, write('Result = '), write(Z), nl.**

**operation(sub,X,Y) :- Z is X-Y, write('Result = '), write(Z), nl.**

**operation(div,X,Y) :- Z is X/Y, write('Result = '), write(Z), nl.**

**operation(mul,X,Y) :- Z is X\*Y, write('Result = '), write(Z), nl.**

π.χ. ?-**run.**

**Entoli = add.**

**First number: 5.**

**Second number: 7.**

**Result = 12**

38

### Έλεγχος Τύπου: Παράδειγμα

Να οριστεί το κατηγορήμα **print\_type**, το οποίο να ζητάει από το χρήστη να πληκτρολογήσει έναν όρο (term) και να τυπώνει στην οθόνη τον τύπο του όρου (π.χ. ακέραιος, άτομο, κλπ). Δηλαδή:

```
?- print_type.
```

```
Give a term: (έστω ότι πληκτρολογούμε :) 5.
```

```
The term 5 is of type: integer
```

```
yes
```

```
?- print_type.
```

```
Give a term: (έστω ότι πληκτρολογούμε:) a.
```

```
The term a is of type: atom
```

```
yes
```

```
?- print_type.
```

```
Give a term: (έστω ότι πληκτρολογούμε:) A.
```

```
The term _312234 is of type: variable
```

```
yes
```

39

### Έλεγχος Τύπου: Παράδειγμα

```
type :-
```

```
write('Give a term: '),
```

```
read(X),
```

```
find_type(X,Type),
```

```
write('The term '), write(X),
```

```
write(' is of type: '), write(Type), nl.
```

```
find_type(X,atom) :- atom(X).
```

```
find_type(X,integer) :- integer(X).
```

```
find_type(X,float) :- float(X).
```

```
find_type(X,compound) :- compound(X).
```

```
find_type(X,variable) :- var(X).
```

40

### Έλεγχος Τύπου & Σύνθεση/Διάσπαση Όρων: Παράδειγμα

Να ορισθεί η διαδικασία **subs (ST,T,ST1,T1)**, η οποία αληθεύει αν ο όρος **T1** προκύπτει από τον όρο **T** με αντικατάσταση του υπο-όρου **ST** με **ST1**.

```
?- subs(a+b, f(a+b), new, T).
```

```
T = f(new)
```

```
subs(T,T,T1,T1). /* Αντικατάστησε όλο τον όρο T=T1 */
```

```
subs(_,T,_,T) :-
```

```
atomic(T). /* Αν ο T δεν είναι σύνθετος όρος, μην αντικαθιστάς τίποτα */
```

```
subs(Sub,Term,Sub1,Term1) :-
```

```
Term =.. [F|Args], /* πάρε τα ορίσματα */
```

```
sublist(Sub,Args,Sub1,Args1), /* αντικατέστησε σε αυτά */
```

```
Term1 =.. [F|Args1]. /* Σύνθεση */
```

```
sublist(_,[],_,[]).
```

```
sublist(Sub,[Term|Terms],Sub1,[Term1|Terms1]): -
```

```
subs(Sub,Term,Sub1,Term1),
```

```
sublist(Sub,Terms,Sub1,Terms1).
```

41