

## Ασκήσεις - Είσοδος/Εξοδος

1. Να γραφεί πρόγραμμα το οποίο να τυπώνει τα στοιχεία μιας λίστας σε διαφορετικές σειρές.

```
writelist([]).
writelist([X|L]) :-
    write(X), nl,
    writelist(L).
```

2. Να γραφεί πρόγραμμα που να τυπώνει τις υπολίστες μιας λίστας σε διαφορετικές σειρές.

```
writelist2([]).
writelist2([X|T]) :-
    writeline(X), nl,
    writelist2(T).
```

```
writeline([]).
writeline([X|T]) :-
    write(X), write(' '),
    writeline(T).
```

Τυπώνει τα στοιχεία μιας λίστας σε μια σειρά.

16

## Παράδειγμα (βιβλίου)

Να γραφεί διαδικασία που να γράφει τα στοιχεία μιας λίστας κυκλικά στα αρχεία : **test1**, **test2** και στην οθόνη.

```
out(X) :- send(X, test1).
```

*Κυκλική Εναλλαγή*

```
next_out(test1, test2).
```

```
send([], _) :- told.
```

```
next_out(test2, user).
```

```
send([X|T], F) :-
```

```
next_out(user, test1).
```

```
    tell(F),
```

```
    write(X),
```

```
    next_out(F, F1),
```

```
    send(T, F1).
```

Π.χ. ?- out([1,2,3,4,5,6]).

Θα δούμε στην οθόνη το 3 και το 6 και τα υπόλοιπα στα αρχεία **test1** και **test2** (ASCII files).

17

## Επεξήγηση name/2

- ?- name('Prolog', X), write(X).

output: [80,114,...]

X = [80,114,...]

- ?- name(X, [80,114,111,...]), write(X).

output: Prolog

X = 'Prolog'

Αν ήταν με μικρό γράμμα: X=prolog.  
Δηλαδή χωρίς εισαγωγικά.

- Ένα άτομο (ή αριθμός) μέσα σε διπλά εισαγωγικά ισοδυναμεί με λίστα ASCII κωδικών.

Π.χ. "Prolog" ↔ [80,114,111, ...]

"a" ↔ [97]

" " ↔ [34]

**Συμπέρασμα:** Οπουδήποτε απαιτείται λίστα ASCII κωδικών μπορούμε να βάζουμε string σε διπλά εισαγωγικά.

18

## Παράδειγμα name/2

Συνένωση 2 strings

```
sconc(N1, N2, N) :-
```

```
    name(N1, Str1),
```

```
    name(N2, Str2),
```

```
    append(Str1, Str2, Str),
```

```
    name(N, Str).
```

Π.χ. ?- sconc(prolog, isgood, X).

X= prologisgood

Δεν χρειάζονται εισαγωγικά.

19

### Ασκήσεις Εισόδου/Εξόδου

Να γραφεί πρόγραμμα που να διαβάζει ένα κείμενο από ένα αρχείο και να το γράφει σ' ένα άλλο αρχείο.

```
file_copy(F1, F2) :-
    see(F1),
    tell(F2),
    copytext.
copytext :-
    get(Ch),
    copy1(Ch).
copy1(46) :-
    put(46),
    seen,
    told.
copy1(Ch) :-
    put(Ch),
    copytext.
```

Π.χ. ?- copy(test1,test2).

Πρόβλημα: αγνοεί τα κενά.

20

### Ασκήσεις Χειρισμού Συμβολοσειρών

Να γραφεί πρόγραμμα που θα διαγράφει από μια συμβολοσειρά κάποια συγκεκριμένη ακολουθία χαρακτήρων, όσες φορές εμφανίζεται.

Π.χ. ?- remove(cd, abcdec dx, X).

X = abex

```
remove(S1,S,NewS) :-
    name(S1,S1L),
    name(S,SL),
    del_all(S1L,SL,NewSL),
    name(NewS,NewSL).
del_all(X,L,NewL) :-
    delstring(X,L,SubL),
    del_all(X,SubL,NewL),
    del_all(X,L,L).
```

```
dstring(X,L,NewL) :-
    append(L1,L2,L),
    append(X,L3,L2),
    append(L1,L3,NewL).
```

→ χωρίζει την L σε L1 και L2  
→ χωρίζει την L2 σε X και L3  
→ συνδέει την L1 και L3 (σβήνει το X)

21

### Ασκήσεις Χειρισμού Συμβολοσειρών

- Να ορισθεί η σχέση **starts (Atom,Char)**, η οποία θα ελέγχει αν το **Atom** αρχίζει με τον χαρακτήρα **Char**.

```
starts(Atom,Char) :-
    name(Atom,[X|T]),
    name(Char,[X]).
```

ή [X|\_]

- Να ορισθεί η σχέση **plural** που να μετατρέπει ονόματα στον πληθυντικό.

Π.χ. ?- plural (table, X).

```
X= tables.
plural(N,Ns) :-
    name(N,List),
    name(s,CodeS),
    append(List,CodeS,NewList),
    name(Ns,NewList).
```

Μετατρέπει το χαρακτήρα s σε λίστα με έναν ASCII κωδικό

append(List,"s",NewList)

22

### Ασκήσεις Χειρισμού Συμβολοσειρών

- Να ορισθεί μια διαδικασία που να ελέγχει αν ένα string είναι παλινδρομικό.

```
palindstring(S) :-
    name(S,NS),
    palindrome(NS).
```

όπου **palindrome/1** είναι το κατηγορημα που ελέγχει αν μία λίστα είναι παλινδρομική:

```
palindrome(L) :-
    reverse(L,L).
```

Άλλη λύση:

```
palindstring(S) :-
    revstring(S,S).
revstring(S,RS) :-
    name(S,L),
    reverse(L,RL),
    name(RS,RL).
```

23