Εργασία Μαθήματος «ΛΟΓΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ»

ΕΡΓΑΣΙΑ 2	
Όνομα φοιτητή – Αρ. Μητρώου	**ΕΥΘΥΜΙΟΣ-ΠΑΤΡΟΚΛΟΣ ΓΕΩΡΓΙΑΔΗΣ – Π19031**
Προθεσμία παράδοσης	31/12/2021

Περιεχόμενα

Κώδικας προγράμματος

```
precede_list([Head1],[Head2|_]):-
    Head1==Head2,!.

precede_list([Head1|Tail1],[Head2|Tail2]):-
    Head1==Head2,
    precede_list(Tail1,Tail2).
```

Κύριο Θέμα

Να γραφούν τα ακόλουθα κατηγορήματα: precede_list(X, Y) που αληθεύει όταν η X λίστα προηγείται της Y λίστας, δηλ. αν η X είναι υπολίστα με κάποια από τα πρώτα από αριστερά στοιχεία της Y.

Για να επιτύχω τον παραπάνω στόχο, θα πρέπει να ελέγξω με συγκεκριμένη σειρά, τα πρώτα στοιχεία των 2 λιστών μεταξύ τους, έπειτα τα δεύτερα στοιχεία των 2 λιστών μεταξύ τους, κ.λ.π, μέχρι η πρώτη λίστα να μην έχει άλλα στοιχεία. Οποιαδήποτε άλλη περίπτωση π.χ. η δεύτερη λίστα να έχει λιγότερα στοιχεία από την πρώτη, ή κάποια λίστα να είναι κενή, είναι εσφαλμένη. Για να ελέγξω τα στοιχεία των 2 λιστών, θα πρέπει αρχικά να ελέγξω αν η κεφαλή της πρώτης λίστας (Head1), δηλαδή το πρώτο στοιχείο της πρώτης λίστας, είναι ίσο με την κεφαλή της δεύτερης λίστας (Head2), δηλαδή το πρώτο στοιχείο της δεύτερης λίστας. Έπειτα θα πρέπει να ελέγξω τα υπόλοιπα στοιχεία των λιστών μεταξύ τους, με τον ίδιο τρόπο. Άρα, θα χρησιμοποιήσω την διαδικασία της αναδρομής, για να ελέγξω και τα υπόλοιπα στοιχεία μεταξύ τους.

Ο πρώτος αναδρομικός κανόνας, θα πρέπει να είναι ο τερματικός κανόνας. Όμως, παρόλο που συνήθως ο τερματικός κανόνας υπάρχει για την περίπτωση της κενής λίστας, έχω κάνει την παραδοχή πως η κενή λίστα δεν είναι υπολίστα μιας άλλης λίστας. Άρα, ο τερματικός κανόνας θα πρέπει να ελέγχει το τελευταίο στοιχείο της πρώτης λίστας με το αντίστοιχο στοιχείο (μπορεί να είναι το τελευταίο στοιχείο της λίστας, μπορεί και όχι) της δεύτερης λίστας (Head1==Head2). Επομένως, εάν η κεφαλή της πρώτης λίστας (Head1), δηλαδή το τελευταίο στοιχείο της αρχικής πρώτης λίστας, είναι ίσο με την κεφαλή της δεύτερης λίστας (Head2), δηλαδή το αντίστοιχο στοιχείο της δεύτερης λίστας, το κατηγόρημα precede_list(X,Y) θα αληθεύει.
Για να τελειώσει η διαδικασία ελέγχου, και να μην ελεγχθεί εάν ισχύει ο δεύτερος αναδρομικός κανόνας αφού έχει αληθεύσει ο τερματικός, χρησιμοποιώ την εντολή cut !, ώστε να παύσει η αναδρομή.
Ο πρώτος αναδρομικός κανόνας, ο τερματικός κανόνας, διαμορφώνεται ως εξής:

```
precede_list([Head1],[Head2|_]):-
  Head1==Head2,!.
```

Ο δεύτερος αναδρομικός κανόνας, θα πρέπει να κόβει τις λίστες σε 2 κομμάτια, την κεφαλή και την ουρά της κάθε λίστας, να ελέγχει εάν οι δύο κεφαλές των λιστών είναι ίσες (Head1==Head2), και να καλεί αναδρομικά το κατηγόρημα precede_list(X.Y), όπου το X

ισούται με την ουρά της πρώτης λίστας (Tail1), και το Υ ισούται με την ουρά της δεύτερης λίστας (Tail2).
Ο δεύτερος αναδρομικός κανόνας διαμορφώνεται ως εξής:

```
precede_list([Head1|Tail1],[Head2|Tail2]):-
   Head1==Head2,
   precede_list(Tail1,Tail2).
```

Παραδείγματα εκτέλεσης κώδικα:

```
?- precede_list([1,2,3],[1,2,3,4,5,6]).
true.

?- precede_list([1,2,3,4,5],[1,2,3,4,5]).
true.

?- precede_list([1,2,3,x,5],[1,2,3,x,5,k]).
true.

?- precede_list([1,2,3,4],[1,2,3]).
false.

?- precede_list([1,y,3],[1,2,3,4,5]).
false.

?- precede_list([],[1,2,3,4,5]).
false.
```