ΕΡΓΑΣΤΗΡΙΟ VIII
ΔΙΑΔΙΚΑΣΙΕΣ ΕΙΣΟΔΟΥ ΕΞΟΔΟΥ

Στην Prolog, η επικοινωνία μεταξύ χρήστη και προγράμματος γίνεται συνήθως με τη μορφή ερωτήσεων και απαντήσεων. Σε πολλές περιπτώσεις όμως, απαιτείται μεγαλύτερη διαλογικότητα μεταξύ χρήστη και προγράμματος. Αυτό μπορεί να επιτευχθεί με ειδικές ενσωματωμένες διαδικασίες, οι οποίες διαβάζουν όρους ή χαρακτήρες από το προκαθορισμένο κανάλι εισόδου (πληκτρολόγιο ή αρχείο) και επιστρέφουν το αποτέλεσμα στο προκαθορισμένο κανάλι εξόδου (οθόνη ή αρχείο).

ΜΕΡΟΣ Ι

Είσοδος/έξοδος από το πληκτρολόγιο

Για την είσοδο και έξοδο χαρακτήρων και όρων από το πληκτρολόγιο υπάρχουν οι παρακάτω διαδικασίες :

Είσοδος

get(X) : διαβάζει τον επόμενο χαρακτήρα (ASCII) που πληκτρολογεί ο χρήστης.

read(X) : διαβάζει τον επόμενο όρο που εισάγεται από το πληκτρολόγιο

Έξοδος

put(X) : τυπώνει στην οθόνη τον χαρακτήρα (ASCII) X.
write(X) : τυπώνει στην οθόνη τον όρο X
nl : προκαλεί αλλαγή γραμμής.

Σημαντικές παρατηρήσεις:

- Στις read και write οι όροι πρέπει πάντα να τελειώνουν με τελεία και αν περιέχει κενά πρέπει να τοποθετείται σε αποστρόφους.
- Το **write** δεν επηρεάζει το πρόγραμμα, το **read** όμως το επηρεάζει. Τα αποτελέσματα των **read** και **write** δεν αλλάζουν με την οπισθοδρόμηση.

Παραδείγματα χρήσης

Ερώτηση	Δεδομένα Χρήστη	Αποτέλεσμα
?- get(X), get(Y).	**cd**	**X = 99** **Y = 100**
?- read(X).	**xyz.**	**X = xyz**
?- read(X).	**'hello'.**	**X = hello**
?- put(70).		**F**
?- write(abc),nl ,write([1,2,3]), nl, write(f(k(1,2))).		**abc** **[1, 2, 3]** **f(k(1, 2))**

ΑΣΚΗΣΗ

Συμπληρώστε τον παρακάτω πίνακα:

Ερώτηση	**Δεδομένα Χρήστη**	**Αποτέλεσμα**
?- read(X).	**f.**	**X = f**
?- read(X).	**[2,6,k].**	
?- get(X).	**r**	**X = 114**
?- get(X),get(Y),get(Z).	**[a]**	
?- put(35).		
?- nl, write(today), nl, write([cat,2,dog]), nl, nl, write(odos(arithmos(12))).		

Είσοδος/έξοδος σε αρχεία

Για την είσοδο/έξοδο όρων και χαρακτήρων σε αρχεία χρησιμοποιούνται τα ίδια κατηγορήματα που χρησιμοποιούνται για είσοδο όρων/χαρακτήρων από το πληκτρολόγιο και έξοδο όρων/χαρακτήρων στην οθόνη, τα οποία παρουσιάστηκαν παραπάνω. Το μόνο επιπλέον που απαιτείται είναι η ανακατεύθυνση των καναλιών εισόδου/εξόδου, έτσι ώστε να χρησιμοποιούνται αρχεία και όχι το πληκτρολόγιο και η οθόνη για αυτή τη δουλειά.

Οι διαδικασίες που χρησιμοποιούνται για το σκοπό αυτό είναι οι εξής:

- **see(Όνομα_Αρχείου)** : Ορίζει το κανάλι εισόδου. Μετά την εκτέλεση της πρότασης αυτής, όλες οι εντολές εισόδου αναφέρονται στο αρχείο με όνομα **Όνομα_Αρχείου** (κανάλι εισόδου).
- **seen** : Ακυρώνει το κανάλι επικοινωνίας που ορίστηκε με την διαδικασία **see** και κλείνει όλα τα αρχεία εισόδου.
- **seeing(**Όνομα_Αρχείου**):**Επιστρέφει το τρέχον κανάλι εισόδου(χρησιμοποιήστε μεταβλητή).
- **tell(Όνομα_Αρχείου)** : Ορίζει το κανάλι εξόδου. Μετά την εκτέλεση της πρότασης αυτής, όλες οι εντολές εξόδου αναφέρονται στο αρχείο με όνομα **Όνομα_Αρχείου** (κανάλι εξόδου).
- **told** : Ακυρώνει το κανάλι επικοινωνίας που ορίστηκε με την διαδικασία **tell** και κλείνει όλα τα αρχεία εξόδου.
- **telling(**Όνομα_Αρχείου**)**:Επιστρέφει το τρέχον κανάλι εξόδου (χρησιμοποιήστε μεταβλητή).

Επανάληψη χωρίς αναδρομή

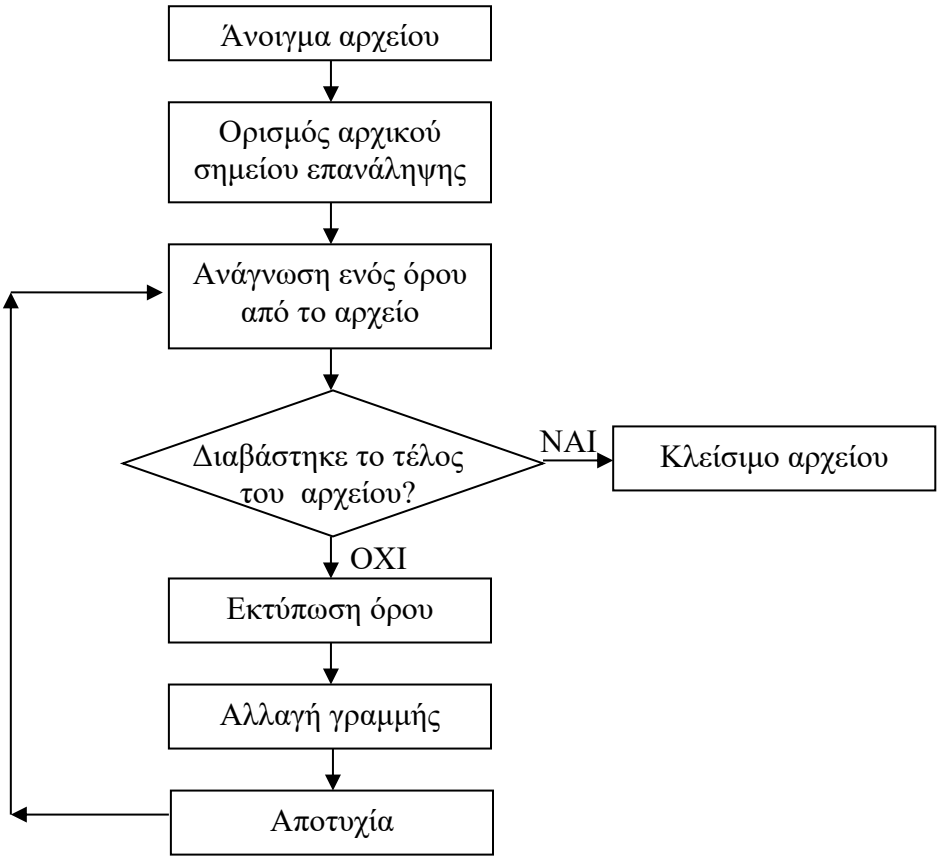
Μέχρι τώρα ο βασικός τρόπος υλοποίησης επαναληπτικών διαδικασιών ήταν μέσω αναδρομικών κλήσεων. Η μέθοδος αυτή είναι η πλέον συνηθισμένη στην Prolog. Μια εναλλακτική μέθοδος επαναληπτικών δομών είναι με τη χρήση του

συνδυασμού repeat-fail. Το **repeat/0** είναι ένα κατηγόρημα συστήματος της Prolog. Η ιδιαίτερη του ιδιότητα, είναι ότι κατά την οπισθοδρόμηση πετυχαίνει πάντα δημιουργώντας ουσιαστικά ένα σημείο πριν από το οποίο δεν μπορεί να επιστρέψει η διαδικασία οπισθοδρόμησης.

Το επόμενο πρόγραμμα διαβάζει τα περιεχόμενα ενός αρχείου και τα τυπώνει στην οθόνη:

```
print_file(File):- see(File), repeat, read(X),(X=end_of_file; write(X),nl, fail), seen.
```

Προσέξτε ότι στην τέταρτη πρόταση του κανόνα, οι ατομικοί τύποι συνδέονται με τον χαρακτήρα ";", ο οποίος αντιπροσωπεύει το ***λογικό ή***. Έτσι, αν ο ατομικός τύπος που διαβάστηκε είναι ο χαρακτήρας τέλους του αρχείου (**X=end_of_file),** η πρόταση θα επιτύχει και η εκτέλεση θα προχωρήσει στην επόμενη πρόταση (seen) χωρίς να εκτελέσει το δεύτερο σκέλος της πρότασης που συνδέεται με το λογικό ή. Σε αντίθετη περίπτωση θα τυπώσει στην οθόνη ότι έχει διαβάσει από το αρχείο **(write(X))**, θα αλλάξει γραμμή (**nl**), θα αποτύχει λόγω του κατηγορήματος fail και η εκτέλεση θα οπισθοδρομήσει μέχρι το κατηγόρημα repeat, από όπου θα επαναληφθεί η εκτέλεση. Η παραπάνω διαδικασία μπορεί να απεικονιστεί με διάγραμμα ροής ως εξής:



Μια εύλογη απορία είναι γιατί χρησιμοποιούμε τον συνδυασμό επανάληψης και αποτυχίας, αφού η παραπάνω διαδικασία θα μπορούσε να υλοποιηθεί αναδρομικά. Ο λόγος είναι η οικονομία μνήμης και η απόδοση του προγράμματος.

Εάν το αρχείο του οποίου τα περιεχόμενα προσπαθούμε να τυπώσουμε είναι μεγάλο τότε τα σημεία οπισθοδρόμησης τα οποία θα δημιουργηθούν από την αναδρομική διαδικασία είναι πιθανό να καταλάβουν πολύ μεγάλο μέρος της μνήμης και ίσως να οδηγήσουν σε ανώμαλο τερματισμό του προγράμματος.

Ο δεύτερος λόγος αφορά την απόδοση του συστήματος και οφείλεται στο γεγονός ότι η διαδικασία οπισθοδρόμησης μέσα στο σώμα ενός κανόνα είναι ταχύτερη από την αναδρομική κλήση.

ΜΕΡΟΣ ΙΙ

Το κατηγόρημα exodos/1 που ακολουθεί δέχεται σαν είσοδο μια λίστα με στοιχεία και τα γράφει σε ένα προκαθορισμένο αρχείο, το file_output.txt. Αρχικά ανακατευθύνει την έξοδο στο αρχείο (κλήση στο κατηγόρημα tell), στη συνέχεια καλεί το κατηγόρημα lista, το οποίο τυπώνει τα στοιχεία της λίστας (χωρίς να προσδιορίζει που τα τυπώνει – μπορεί να χρησιμοποιηθεί και για έξοδο στην οθόνη) και τέλος καλεί το κατηγόρημα told, καθιστώντας έτσι και πάλι κανάλι εξόδου την οθόνη.

```
exodos(List):- tell('file_output.txt'), lista(List), told.
```

ΑΣΚΗΣΗ

Φτιάξτε τον κανόνα lista(List) ο οποίος θα δέχεται μια λίστα και θα τυπώνει (χρησιμοποιώντας την write(X)) τα στοιχεία της λίστας με δύο κενά μεταξύ τους.

_____

_____

_____

Αν θέλουμε τη λίστα να μην την λαμβάνει από το πληκτρολόγιο αλλά από ένα αρχείο filename, ο παραπάνω κανόνας του κατηγορήματος **exodus** μετατρέπεται ως εξής:

```
exodos2(List,Filename):- see(Filename), read(List), seen,
                         tell('file_output.txt'),lista(List), told.
```

Όταν διαβάζει (read) την είσοδο από αρχείο περιμένει να διαβάσει στο τέλος κάθε πρότασης την τελεία '**.**' και για να σταματήσει να διαβάζει από το αρχείο περιμένει να δει το τέλος του αρχείου που απεικονίζεται ως - end_of_file.

ΑΣΚΗΣΗ

i. Δημιουργήστε με το Notepad στον ίδιο κατάλογο με το πρόγραμμά σας ένα αρχείο lab8.txt με περιεχόμενα:

man(giorgos).
woman(maria).

Χρησιμοποιείστε το πρόγραμμα **print_file(File)** του Μέρους Ι με είσοδο το αρχείο που φτιάξατε (**?- print_file('lab8.txt').**). Ποια είναι η έξοδος που παίρνετε; Το περιμένατε;

_____

_____

_____

_____

ii. Μετατρέψτε το παραπάνω πρόγραμμα έτσι ώστε να χρησιμοποιεί αναδρομική κλήση του κατηγορήματος print_file/1 και να έχει το ίδιο αποτέλεσμα.

_____

_____

_____

_____

_____