



ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ  
Εθνικό και Καποδιστριακό  
Πανεπιστήμιο Αθηνών

---

## Λογικός Προγραμματισμός

Ασκήσεις

Παναγιώτης Σταματόπουλος

Τμήμα Πληροφορικής και Τηλεπικοινωνιών

---

## **Περιεχόμενα**

1.	Α' Ομάδα Ασκήσεων "Λογικού Προγραμματισμού" Ακαδημαϊκού Έτους 2013-14.....	3
1.1	Άσκηση 1.....	3
1.2	Άσκηση 2.....	3
1.3	Άσκηση 3.....	4

# 1. Α' Ομάδα Ασκήσεων "Λογικού Προγραμματισμού" Ακαδημαϊκού Έτους 2013-14

## 1.1 Άσκηση 1

Θεωρήστε ότι ένας πίνακας μπορεί να αναπαρασταθεί στην Prolog σαν μία λίστα από τις γραμμές του, όπου κάθε γραμμή είναι μία λίστα από τα στοιχεία της. Έχοντας αυτό σαν δεδομένο, ορίστε το κατηγορημα `matr_transp/2` έτσι ώστε το `matr_transp(M1,M2)` να επιστρέφει στο `M2` τον ανάστροφο πίνακα (δηλαδή αυτόν που προκύπτει με εναλλαγή γραμμών και στηλών) του `M1`. Για παράδειγμα:

```
?- matr_transp([[5,8,9,7,2],[3,6,1,1,4],[2,4,2,8,0]],M).
```

```
M = [[5,3,2],[8,6,4],[9,1,2],[7,1,8],[2,4,0]]
```

Ορίστε και το κατηγορημα `matr_mult/3` έτσι ώστε το `matr_mult(M1,M2,M3)` να επιστρέφει στο `M3` το γινόμενο των πινάκων `M1` και `M2`. Για παράδειγμα:

```
?- matr_mult([[1,2,3],[4,5,6],[6,5,4],[3,2,1]],  
             [[3,4],[5,6],[7,8]],M).
```

```
M = [[34,40],[79,94],[71,86],[26,32]]
```

Τέλος, ορίστε και το κατηγορημα `matr_det/2` έτσι ώστε το `matr_det(M,D)` να επιστρέφει στο `D` την ορίζουσα του πίνακα `M`. Για παράδειγμα:

```
?- matr_det([[4,3,2,1],[3,2,1,2],[1,4,1,3],[2,1,3,2]],D).
```

```
D = 51
```

Παραδοτέο για την άσκηση είναι ένα πηγαίο αρχείο Prolog με όνομα `matopers.pl`, μέσα στο οποίο θα πρέπει να είναι ορισμένα και τα τρία κατηγορήματα που ζητούνται.

## 1.2 Άσκηση 2

Ορίστε σε Prolog ένα κατηγορημα `exchange/2`, το οποίο να παίρνει σαν πρώτο όρισμα μία λίστα θετικών ακεραίων και να επιστρέφει στο δεύτερο όρισμα τη μέγιστη αποτίμηση αυτής της λίστας, όπως θα εξηγηθεί αυτός ο όρος στη συνέχεια. Η βασική αποτίμηση μίας λίστας ακεραίων είναι η αριθμητική τιμή της έκφρασης που προκύπτει αν εναλλάσσουμε αφαιρέσεις και προσθέσεις μεταξύ των στοιχείων της λίστας, με τη σειρά που είναι αυτά δοσμένα. Για παράδειγμα, η λίστα `[9,8,2]` έχει βασική αποτίμηση  $9-8+2=3$ , ενώ η `[1,4,89,12]` έχει βασική αποτίμηση  $1-4+89-12=74$ . Οποιαδήποτε άλλη αποτίμηση μίας λίστας είναι η βασική αποτίμηση μίας λίστας που παίρνουμε με εναλλαγή στις θέσεις των αριθμών ενός ή περισσότερων ζευγαριών διαδοχικών στοιχείων με ταυτόχρονη αντιστροφή στη σειρά των ψηφίων αυτών των αριθμών. Δηλαδή οι διαδοχικοί αριθμοί `17, 34` μπορούν να γίνουν `43, 71` και οι `123, 45` να γίνουν `54, 321`. Όμως, κανείς αριθμός δεν μπορεί να μετακινηθεί περισσότερο από μία θέση μακριά από την αρχική του. Το ζητούμενο από το κατηγορημα `exchange/2` είναι να μας επιστρέφει τη μέγιστη δυνατή αποτίμηση μίας λίστας ακεραίων, από τη βασική της ή από αυτές που προκύπτουν με εναλλαγές στοιχείων, όπως περιεγράφηκε προηγουμένως. Κάποια παραδείγματα εκτέλεσης είναι τα εξής:

```
?- exchange([1,2],V).
```

```
V = 1
```

```
?- exchange([12,56,34],V).
```

```
V = 78
```

```
?- exchange([210,123,34,455,8,100],V).  
V = 934
```

```
?- exchange([144,5,62,37,110,24,8,55,21,100],V).
```

```
V = 308
```

Παραδοτέο για την άσκηση είναι ένα πηγαίο αρχείο Prolog με όνομα **exchange.pl**.

### 1.3 Άσκηση 3

Μία εκδοχή του λεγόμενου job-shop προβλήματος είναι η εξής: Έχουμε να εκτελέσουμε μία σειρά από έργα  $j_1, j_2, j_3 \dots$ , ανεξάρτητα το ένα από το άλλο, καθένα από τα οποία αποτελείται από έναν αριθμό από εργασίες  $t_{11}, t_{12} \dots, t_{21}, t_{22}, \dots, t_{31}, t_{32}, \dots$  οι οποίες πρέπει να εκτελεσθούν με τη δεδομένη σειρά για κάθε έργο. Κάθε εργασία μπορεί να εκτελεσθεί σε συγκεκριμένο τύπο μηχανής και χρειάζεται δεδομένο χρόνο για να έλθει σε πέρας. Επίσης είναι γνωστό το πλήθος των μηχανών που έχουμε διαθέσιμες από κάθε τύπο. Αυτό που θέλουμε να βρούμε είναι πότε και σε ποιες μηχανές πρέπει να εκτελέσουμε τις εργασίες έτσι ώστε όλα τα έργα να έχουν τελειώσει πριν από δεδομένη προθεσμία. Ένα συγκεκριμένο στιγμιότυπο αυτού του προβλήματος δίνεται με τα παρακάτω γεγονότα Prolog:

```
job(j1,[t11,t12]).  
job(j2,[t21,t22,t23]).  
job(j3,[t31]).  
job(j4,[t41,t42]).
```

```
task(t11,m1,2).  
task(t12,m2,6).  
task(t21,m2,5).  
task(t22,m1,3).  
task(t23,m2,3).  
task(t31,m2,4).  
task(t41,m1,5).  
task(t42,m2,2).
```

```
machine(m1,1).  
machine(m2,2).
```

```
deadline(14).
```

Τα γεγονότα `job/2` περιγράφουν τα έργα που πρέπει να εκτελέσουμε και τις εργασίες από τις οποίες αποτελούνται, κάθε γεγονός `task/3` δίνει, για μία εργασία, σε τι τύπου μηχανή πρέπει να εκτελεσθεί και πόσες μονάδες χρόνου απαιτούνται για την εκτέλεσή της, τα γεγονότα `machine/2` μας πληροφορούν πόσες μηχανές έχουμε διαθέσιμες από κάθε τύπο και τέλος το κατηγορημα `deadline/1` δίνει την προθεσμία (σε μονάδες χρόνου) μέσα στην οποία πρέπει να έχουν εκτελεσθεί όλα τα έργα. Ορίστε σε Prolog το κατηγορημα `job_shop/1`, το οποίο να επιστρέφει το πρόγραμμα εκτέλεσης των εργασιών στις μηχανές. Παράδειγμα:

```
?- job_shop(S).  
S = [execs(m1,[t(t11,0,2),t(t41,2,7),t(t22,7,10)]),  
     execs(m2,[t(t12,2,8),t(t42,8,10),t(t23,10,13)])],
```

```
execs(m2, [t(t21, 0, 5), t(t31, 5, 9)])] -> ;
```

```
S = .....  
.....
```

```
?- findall(S, job_shop(S), L), length(L, N).
```

```
.....  
N = 9136
```

Η παραπάνω πρώτη λύση του προβλήματος σημαίνει ότι στη μηχανή τύπου m1 θα εκτελεσθούν από τη χρονική στιγμή 0 έως τη 2 η εργασία t11, από τη χρονική στιγμή 2 έως την 7 η εργασία t41 και από τη χρονική στιγμή 7 έως τη 10 η εργασία t22. Μετά ακολουθούν τα προγράμματα εργασίας για τις δύο μηχανές τύπου m2.

Μία δεύτερη εκδοχή του job-shop προβλήματος είναι αυτή στην οποία για κάθε εργασία έχουμε και ένα δεδομένο πλήθος ατόμων που πρέπει να εργασθούν στη μηχανή που θα τη φέρει σε πέρας. Δηλαδή, τώρα έχουμε το κατηγορήμα task/4 (αντί για task/3), όπου το τελευταίο επιπλέον όρισμα δίνει και το πλήθος αυτό των ατόμων. Τα γεγονότα τώρα είναι:

```
task(t11, m1, 2, 3).  
task(t12, m2, 6, 2).  
task(t21, m2, 5, 2).  
task(t22, m1, 3, 3).  
task(t23, m2, 3, 2).  
task(t31, m2, 4, 2).  
task(t41, m1, 5, 4).  
task(t42, m2, 2, 1).
```

Επίσης έχουμε δεδομένο και το προσωπικό που υπάρχει διαθέσιμο ανά πάσα χρονική στιγμή, το οποίο ορίζεται με το γεγονός staff/1, δηλαδή:

```
staff(6).
```

Υποτίθεται ότι κάθε εργαζόμενος μπορεί να δουλέψει σε οποιαδήποτε μηχανή για οποιοδήποτε εργασία.

Ορίστε σε Prolog το κατηγορήμα job\_shop\_with\_manpower/1, το οποίο να λύνει και αυτήν την εκδοχή του job-shop προβλήματος. Παράδειγμα εκτέλεσης:

```
?- job_shop_with_manpower(S).  
S = [execs(m1, [t(t11, 0, 2), t(t41, 2, 7), t(t22, 7, 10)]),  
     execs(m2, [t(t21, 0, 5), t(t12, 5, 11), t(t23, 11, 14)]),  
     execs(m2, [t(t42, 7, 9), t(t31, 10, 14)])] -> ;
```

```
S = .....  
.....
```

```
?- findall(S, job_shop_with_manpower(S), L), length(L, N).
```

```
.....  
N = 8
```

Τα δεδομένα του προβλήματος, στη μορφή των γεγονότων Prolog που δίνονται στην εκφώνηση, βρίσκονται στο [http://www.di.uoa.gr/~takis/jobshop\\_data.pl](http://www.di.uoa.gr/~takis/jobshop_data.pl).

Παραδοτέο για την άσκηση είναι **ένα πηγαίο αρχείο Prolog** με όνομα `jobshop.pl`.

## Σημειώματα

### Σημείωμα Ιστορικού Εκδόσεων Έργου

Το παρόν έργο αποτελεί την έκδοση 1.0.

### Σημείωμα Αναφοράς

Copyright Εθνικών και Καποδιστριακών Πανεπιστημίων Αθηνών, Παναγιώτης Σταματόπουλος.  
«Λογικός Προγραμματισμός, Η γλώσσα προγραμματισμού Prolog». Έκδοση: 1.0. Αθήνα 2015.  
Διαθέσιμο από τη δικτυακή διεύθυνση: <http://opencourses.uoa.gr/courses/DI117/>.

### Σημείωμα Αδειοδότησης

Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά, Μη Εμπορική Χρήση Παρόμοια Διανομή 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό και τα οποία αναφέρονται μαζί με τους όρους χρήσης τους στο «Σημείωμα Χρήσης Έργων Τρίτων».



[1] <http://creativecommons.org/licenses/by-nc-sa/4.0/>

Ως **Μη Εμπορική** ορίζεται η χρήση:

- που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
- που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
- που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

### Διατήρηση Σημειωμάτων

- Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:
- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων

- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει) μαζί με τους συνοδευόμενους υπερσυνδέσμους.

### **Σημείωμα Χρήσης Έργων Τρίτων**

Το Έργο αυτό κάνει χρήση των ακόλουθων έργων:

Εικόνες/Σχήματα/Διαγράμματα/Φωτογραφίες



## Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στο πλαίσιο του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αθηνών**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

