ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Εαρινό Εξάμηνο 2026

ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ Ι

Άσκηση 3

Καταληκτική ημερομηνία και ώρα ηλεκτρονικής υποβολής: **26/6/2026, 23:59:59**

Μετακινήσεις στο ΕΜΠ, ξανά (0.25 βαθμοί)

Το πρόβλημα με τις μετακινήσεις στο ΕΜΠ και τα εμπόδια σας είναι γνωστό από την πρώτη σειρά ασκήσεων της φετινής χρονιάς. Το ζητούμενο αυτής της άσκησης είναι να γραφεί η λύση του προβλήματος σε Prolog. Το βασικό κατηγόρημα του προγράμματός σας θα πρέπει να συμπεριφέρεται ως συνάρτηση (δηλ. να δίνει πάντα μία μόνο λύση) όπως φαίνεται παρακάτω:

```
?- walks("grid1.txt", Walks).
Walks = 1.
?- walks("grid2.txt", Walks).
Walks = 10.
```

Ηλεκτρολόγοι στο σκοτάδι, ξανά (0.25 βαθμοί)

Το πρόβλημα με τα φώτα στα κτίρια του ΕΜΠ σας είναι γνωστό από τη δεύτερη σειρά ασκήσεων της φετινής χρονιάς. Το ζητούμενο αυτής της άσκησης είναι να γραφεί η λύση του προβλήματος σε Python. Το πρόγραμμά σας θα πρέπει να δουλεύει όπως φαίνεται παρακάτω:

```
$ python3 lights.py lights1.txt
1
$ python3 lights.py lights2.txt
4
$ python3 lights.py lights3.txt
IMPOSSIBLE
```

Στρατηγική Σπουδών (0.25+0.25 = 0.5 βαθμοί)

Ο Δημήτρης, φοιτητής στη ΣΗΜΜΥ, θέλει να ολοκληρώσει τις σπουδές του **όσο πιο γρήγορα γίνεται**. Για να το πετύχει αυτό, ξεκινάει να σχεδιάζει σε ποιο εξάμηνο θα πάρει κάθε μάθημα. Έχει στη διάθεσή του δύο ειδών πληροφορίες:

- Τα **προαπαιτούμενα** κάθε μαθήματος: για κάποια ζευγάρια μαθημάτων (a, b) ισχύει ότι το μάθημα a πρέπει να έχει ολοκληρωθεί σε προηγούμενο (**αυστηρά μικρότερο**) εξάμηνο από το μάθημα b.
- Τις **χρονικές συγκρούσεις**: για κάποια ζευγάρια μαθημάτων (x, y), οι διαλέξεις και τα εργαστήριά τους πέφτουν τις ίδιες ώρες της εβδομάδας, οπότε ο Δημήτρης δεν μπορεί να τα παρακολουθήσει **στο ίδιο εξάμηνο**.

Ο Δημήτρης θέλει να αναθέσει σε κάθε μάθημα έναν θετικό ακέραιο αριθμό εξαμήνου, με τέτοιο τρόπο ώστε να ικανοποιούνται όλοι οι παραπάνω περιορισμοί, και να ελαχιστοποιείται ο **μέγιστος** αριθμός εξαμήνου που χρησιμοποιείται. Επιπλέον, ξέρει ότι αν πάρει παραπάνω από K μαθήματα σε ένα εξάμηνο δεν θα τα προλάβει· επομένως **σε κάθε εξάμηνο μπορεί να αντιστοιχιστούν το πολύ K μαθήματα**.

Η άσκηση σας ζητάει να γράψετε δύο προγράμματα (ένα σε Python και ένα σε Prolog) τα οποία να διαβάζουν την περιγραφή των μαθημάτων από ένα αρχείο κειμένου και να τυπώνουν τον **ελάχιστο αριθμό εξαμήνων** που χρειάζεται ο Δημήτρης για να ολοκληρώσει όλα τα μαθήματα. Αν οι περιορισμοί προαπαιτούμενων είναι αντιφατικοί (π.χ. υπάρχει κύκλος στις σχέσεις προαπαιτούμενων), το πρόγραμμα θα πρέπει να τυπώνει `IMPOSSIBLE` (στην Python) ή να αποτυγχάνει (στην Prolog).

Τα στοιχεία εισόδου διαβάζονται από ένα αρχείο κειμένου. Η πρώτη γραμμή περιέχει τρεις ακέραιους N, M και K ($1 \leq N \leq 20$, $0 \leq M \leq N(N-1)$, $1 \leq K \leq N$), που είναι αντίστοιχα ο αριθμός των μαθημάτων (αριθμημένα από το 1 μέχρι το N), ο αριθμός των ζευγαριών προαπαιτούμενων και ο μέγιστος αριθμός μαθημάτων ανά εξάμηνο. Καθεμιά από τις επόμενες M γραμμές περιέχει δύο ακέραιους a και b, που σημαίνει ότι το μάθημα a είναι προαπαιτούμενο του b. Η επόμενη γραμμή περιέχει έναν ακέραιο C ($0 \leq C \leq N(N-1)/2$), τον αριθμό των ζευγαριών μαθημάτων με χρονική σύγκρουση. Καθεμιά από τις επόμενες C γραμμές περιέχει δύο ακέραιους x και y, που σημαίνει ότι τα μαθήματα x και y δεν μπορούν να είναι στο ίδιο εξάμηνο.

Παρακάτω δίνονται τρία παραδείγματα αρχείων εισόδου:

```
$ cat studies1.txt
6 4 2
1 3
2 3
3 5
4 5
2
1 2
3 4
```

```
$ cat studies2.txt
6 1 2
1 2
0
```

```
$ cat studies3.txt
3 3 5
1 2
2 3
3 1
0
```

Τα προγράμματά σας θα πρέπει να τρέχουν όπως φαίνεται παρακάτω.

Σε Python

```
$ python3 studies.py studies1.txt
4
$ python3 studies.py studies2.txt
3
$ python3 studies.py studies3.txt
IMPOSSIBLE
```

Σε Prolog

```
?- studies("studies1.txt", Semesters).
Semesters = 4.
?- studies("studies2.txt", Semesters).
Semesters = 3.
?- studies("studies3.txt", Semesters).
false.
```

Εξηγούμε το πρώτο παράδειγμα. Έχουμε 6 μαθήματα και ο Δημήτρης μπορεί να πάρει το πολύ $K = 2$ μαθήματα ανά εξάμηνο. Τα μαθήματα 1 και 2 είναι προαπαιτούμενα του 3, και τα μαθήματα 3 και 4 είναι προαπαιτούμενα του 5. Επιπλέον, τα μαθήματα 1, 2 και 3, 4 έχουν ανά δύο χρονική σύγκρουση. Μια βέλτιστη ανάθεση είναι: εξάμηνο 1: μαθήματα {1, 4}, εξάμηνο 2: {2, 6}, εξάμηνο 3: {3}, εξάμηνο 4: {5}. Άρα ο ελάχιστος αριθμός εξαμήνων είναι 4.

Στο δεύτερο παράδειγμα δεν υπάρχουν χρονικές συγκρούσεις και η μόνη σχέση προαπαιτούμενου είναι $1 \to 2$. Καθώς όμως υπάρχουν 6 μαθήματα και $K = 2$ μαθήματα ανά εξάμηνο, χρειάζονται τουλάχιστον 3 εξάμηνα. Πράγματι, 3 εξάμηνα αρκούν: π.χ. εξάμηνο 1: {1, 3}, εξάμηνο 2: {2, 4}, εξάμηνο 3: {5, 6}.

Στο τρίτο παράδειγμα τα προαπαιτούμενα σχηματίζουν κύκλο $1 \to 2 \to 3 \to 1$, άρα δεν υπάρχει έγκυρη ανάθεση.

Περαιτέρω οδηγίες για τις ασκήσεις

- Μπορείτε να δουλέψετε σε ομάδες το πολύ δύο ατόμων. Μπορείτε αν θέλετε να σχηματίσετε διαφορετική ομάδα σε σχέση με τις προηγούμενες σειρές ασκήσεων – οι ομάδες στο σύστημα υποβολής είναι έτσι και αλλιώς καινούργιες για κάθε σειρά ασκήσεων.

- Δεν επιτρέπεται να μοιράζεστε τα προγράμματά σας με συμφοιτητές εκτός της ομάδας σας ή να τα βάλετε σε μέρος που άλλοι μπορούν να τα βρουν (π.χ. σε κάποια σελίδα στο διαδίκτυο, σε ιστοσελίδες συζητήσεων, ...). Σε περίπτωση που παρατηρηθούν *περίεργες* ομοιότητες σε προγράμματα, ο βαθμός των εμπλεκόμενων φοιτητών σε **όλες τις σειρές ασκήσεων** γίνεται αυτόματα μηδέν ανεξάρτητα από το ποια ομάδα... *εμπνεύστηκε* από την άλλη.

- Σε αντίθεση με το παραπάνω, επιτρέπεται να μοιράζεστε test cases.

- Μπορείτε να χρησιμοποιήσετε "βοηθητικό" κώδικα (π.χ. κάποιο κώδικα που διαχειρίζεται κάποια δομή δεδομένων) που βρήκατε στο διαδίκτυο στα προγράμματά σας, με την προϋπόθεση ότι το πρόγραμμά σας περιέχει σε σχόλια την παραδοχή για την προέλευση αυτού του κώδικα και ένα σύνδεσμο σε αυτόν.

- Τα προγράμματα σε Python πρέπει να είναι σε ένα αρχείο, να μην χρησιμοποιούν αντικειμενοστραφή χαρακτηριστικά (classes) και να δουλεύουν με CPython 3.13.5. Επιτρέπεται μόνο η χρήση της ενσωματωμένης βιβλιοθήκης (standard library).

- Τα προγράμματα σε Prolog πρέπει να είναι σε ένα αρχείο και να δουλεύουν σε ένα από τα συστήματα Prolog που υποστηρίζει ο plgrader (GNU-Prolog 1.4.5, SWI-Prolog 9.2.9 ή YAP 7.6.0).

- Η υποβολή των προγραμμάτων θα γίνει ηλεκτρονικά μέσω του helios και για να μπορέσετε να τα υποβάλλετε και να βαθμολογηθείτε για αυτά, τα μέλη της ομάδας σας (και οι δύο) θα πρέπει να έχετε εγγραφεί στο μάθημα στο helios. Τα προγράμματά σας πρέπει να διαβάζουν την είσοδο όπως αναφέρεται και δεν πρέπει να έχουν κάποιου άλλους είδους έξοδο διότι δεν θα γίνουν δεκτά από το σύστημα στο οποίο θα υποβληθούν.