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

Εαρινό Εξάμηνο 2026

**ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ Ι**

# Άσκηση 2

Καταληκτική ημερομηνία και ώρα ηλεκτρονικής υποβολής: **15/5/2026, 23:59:59**

## Ηλεκτρολόγοι στο Σκοτάδι (0.25+0.25 = 0.5 βαθμοί)

Στο ολοκαίνουργιο κτίριο της Σχολής ΗΜΜΥ του ΕΜΠ, όπου τα δωμάτια είναι διατεταγμένα σε ένα πλέγμα μεγέθους $N \times N$, ένας πρωτοετής ηλεκτρολόγος καλωδίωσε λανθασμένα τους διακόπτες φωτισμού. Το αποτέλεσμα είναι ότι κάθε διακόπτης δεν ελέγχει μόνο το δικό του φως αλλά αλλάζει ταυτόχρονα και την κατάσταση των φώτων στα γειτονικά δωμάτια (πάνω, κάτω, αριστερά, δεξιά).

Πιο συγκεκριμένα, κάθε δωμάτιο στη θέση $(i, j)$ του πλέγματος έχει ένα φως που βρίσκεται είτε σε κατάσταση **αναμμένο** (1) είτε **σβηστό** (0). Όταν πατηθεί ο διακόπτης ενός δωματίου $(i, j)$, αλλάζει η κατάσταση (από αναμμένο σε σβηστό και αντίστροφα) τόσο του φωτός στο $(i, j)$ όσο και των φώτων στα δωμάτια $(i-1, j)$, $(i+1, j)$, $(i, j-1)$ και $(i, j+1)$, εφόσον αυτά υπάρχουν εντός του πλέγματος.

Ο επιστάτης του κτιρίου θέλει να σβήσει όλα τα φώτα για εξοικονόμηση ενέργειας. Η άσκηση σας ζητάει να γράψετε δύο προγράμματα (ένα σε Java και ένα σε ML) τα οποία να διαβάζουν την αρχική κατάσταση του πλέγματος από ένα αρχείο και να τυπώνουν τον **ελάχιστο αριθμό πατημάτων διακοπτών** που απαιτείται ώστε όλα τα φώτα να σβήσουν. Αν δεν υπάρχει τρόπος να σβήσουν όλα τα φώτα, το πρόγραμμα θα πρέπει να τυπώνει `IMPOSSIBLE`.

Τα στοιχεία εισόδου διαβάζονται από ένα αρχείο κειμένου. Η πρώτη γραμμή περιέχει το μέγεθος του πλέγματος $N$ ($1 \leq N \leq 5$). Κάθε μία από τις επόμενες $N$ γραμμές περιέχει $N$ ακεραίους (0 ή 1) χωρισμένους με κενά, που αντιστοιχούν στην αρχική κατάσταση κάθε φωτός.

Παρακάτω δίνονται τρία παραδείγματα εισόδου και οι αντίστοιχες έξοδοι:

```
$ cat lights1.txt
3
0 1 0
1 1 1
0 1 0
```

```
$ cat lights2.txt
4
1 1 1 1
1 1 1 1
1 1 1 1
1 1 1 1
```

```
$ cat lights3.txt
5
0 0 1 0 0
1 0 1 1 1
0 1 0 1 0
1 1 0 0 0
0 1 0 0 0
```

**Σε Java**

```
$ java Lights lights1.txt
1
$ java Lights lights2.txt
4
$ java Lights lights3.txt
IMPOSSIBLE
```

**Σε SML/NJ**

```
- lights "lights1.txt";
1
val it = () : unit
- lights "lights2.txt";
4
val it = () : unit
- lights "lights3.txt";
IMPOSSIBLE
val it = () : unit
```

**Σε MLton ή σε OCaml**

```
$ ./lights lights1.txt
1
$ ./lights lights2.txt
4
$ ./lights lights3.txt
IMPOSSIBLE
```

Στο πρώτο παράδειγμα, αρκεί να πατηθεί ο διακόπτης στο κεντρικό δωμάτιο $(2, 2)$ (με αρίθμηση από το 1) για να σβήσουν όλα τα φώτα. Στο δεύτερο παράδειγμα, χρειάζονται τέσσερα πατήματα. Στο τρίτο παράδειγμα, δεν υπάρχει τρόπος να σβήσουν όλα τα φώτα.

## Βουνό ή Θάλασσα (0.25+0.25 = 0.5 βαθμοί)

Ο Βαγγέλης, αφού ολοκλήρωσε τις εξετάσεις του στη ΣΗΜΜΥ, σχεδιάζει να κάνει καλοκαιρινές διακοπές μαζί με τη φίλη του τη Μαρία. Στον Βαγγέλη αρέσουν περισσότερο οι διακοπές στο βουνό, ενώ η Μαρία προτιμά τη θάλασσα. Για να είναι δίκαιοι, αποφασίζουν ότι θα πρέπει να περάσουν **ακριβώς τις μισές μέρες στο βουνό και τις μισές στη θάλασσα**.

Επικοινωνούν με τον ταξιδιωτικό τους πράκτορα και, καθώς οι οικονομίες τους είναι μετρημένες, του ζητάνε να βρει για κάθε μέρα του καλοκαιριού το οικονομικότερο δωμάτιο στο βουνό και το οικονομικότερο στη θάλασσα. Ο πράκτορας τους δίνει ένα αρχείο κειμένου που περιέχει $N$ γραμμές ($1 \leq N \leq 1.000.000$), μία για κάθε ημέρα (οι ημέρες αριθμούνται από το 1 μέχρι και το $N$). Κάθε γραμμή περιέχει δύο θετικούς ακεραίους (το πολύ 1000): την τιμή δωματίου στο βουνό και την τιμή δωματίου στη θάλασσα, αντίστοιχα.

Οι δύο φίλοι αποφασίζουν ότι κάθε μέρα θα επιλέγουν το φθηνότερο από τα δύο δωμάτια. Σε περίπτωση ισοτιμίας, τις μέρες με **περιττό** αύξοντα αριθμό θα επιλέγουν το βουνό (πρώτη τιμή) και τις μέρες με **άρτιο** αύξοντα αριθμό θα επιλέγουν τη θάλασσα (δεύτερη τιμή).

Μια έγκυρη περίοδος διακοπών είναι ένα **συνεχόμενο διάστημα ημερών** στο οποίο ο αριθμός ημερών στο βουνό ισούται με τον αριθμό ημερών στη θάλασσα (δηλαδή η περίοδος έχει άρτιο μήκος). Η άσκηση σας ζητάει να γράψετε δύο προγράμματα (ένα σε Java και ένα σε ML) που να βρίσκουν:

1. Πόσες διαφορετικές έγκυρες περιόδους διακοπών μπορούν να κάνουν οι δύο φίλοι.
2. Πόσο είναι το μεγαλύτερο σε διάρκεια συνεχόμενο χρονικό διάστημα που μπορούν να κάνουν διακοπές.
3. Πόσο είναι το ελάχιστο δυνατό κόστος για να κάνουν διακοπές με τη μέγιστη δυνατή διάρκεια.

Το πρόγραμμα θα πρέπει να τυπώνει τα τρία ζητούμενα χωρισμένα με κενά σε μία γραμμή. Αν δεν υπάρχει καμία έγκυρη περίοδος διακοπών, θα πρέπει να τυπώνει `0 0 0`.

Τα στοιχεία εισόδου διαβάζονται από ένα αρχείο κειμένου. Η πρώτη γραμμή περιέχει τον ακέραιο $N$. Κάθε μία από τις επόμενες $N$ γραμμές περιέχει δύο θετικούς ακεραίους χωρισμένους με κενό: την τιμή στο βουνό και την τιμή στη θάλασσα, αντίστοιχα.

Για παράδειγμα, έστω ότι τρία αρχεία εισόδου περιέχουν τα εξής:

```
$ cat vacation1.txt
5
10 20
17 15
21 18
12 19
13 13
```

```
$ cat vacation2.txt
3
10 20
20 10
10 20
```

```
$ cat vacation3.txt
3
10 20
5 15
8 18
```

Τα προγράμματά σας θα πρέπει να τρέχουν όπως φαίνεται παρακάτω.

**Σε Java**

```
$ java Vacation vacation1.txt
4 4 55
$ java Vacation vacation2.txt
2 2 20
$ java Vacation vacation3.txt
0 0 0
```

**Σε SML/NJ**

```
- vacation "vacation1.txt";
4 4 55
val it = () : unit
- vacation "vacation2.txt";
2 2 20
val it = () : unit
- vacation "vacation3.txt";
0 0 0
val it = () : unit
```

**Σε MLton ή σε OCaml**

```
$ ./vacation vacation1.txt
4 4 55
$ ./vacation vacation2.txt
2 2 20
$ ./vacation vacation3.txt
0 0 0
```

Στο πρώτο παράδειγμα, υπάρχουν τέσσερις διαφορετικές έγκυρες περίοδοι διακοπών: δύο τρόποι για δύο ημέρες (ημέρες 1–2 και ημέρες 3–4) και δύο τρόποι για τέσσερις ημέρες (ημέρες 1–4 και ημέρες 2–5). Η μέγιστη δυνατή διάρκεια είναι τέσσερις μέρες. Το ελάχιστο κόστος για τέσσερις μέρες διακοπών είναι 55, το οποίο προκύπτει μένοντας τη μέρα 1 στο βουνό (10), τη μέρα 2 στη θάλασσα (15), τη μέρα 3 στη θάλασσα (18) και τη μέρα 4 στο βουνό (12), δηλαδή $10 + 15 + 18 + 12 = 55$. Ο εναλλακτικός τρόπος (ημέρες 2–5) θα κόστιζε $15 + 18 + 12 + 13 = 58$.

Στο δεύτερο παράδειγμα, υπάρχουν δύο έγκυρες περίοδοι (ημέρες 1–2 και ημέρες 2–3), και οι δύο διάρκειας δύο ημερών με ελάχιστο κόστος 20.

Στο τρίτο παράδειγμα, κάθε μέρα επιλέγεται το βουνό (αφού η τιμή του βουνού είναι πάντα μικρότερη) και δεν υπάρχει κανένα συνεχόμενο διάστημα με ίσο αριθμό ημερών βουνού και θάλασσας.

Μπορείτε να θεωρήσετε ότι οι τιμές των δωματίων είναι θετικές και δεν υπερβαίνουν το 1000.

## Περαιτέρω οδηγίες για τις ασκήσεις

- Μπορείτε να δουλέψετε σε ομάδες το πολύ δύο ατόμων. Μπορείτε αν θέλετε να σχηματίσετε διαφορετική ομάδα σε σχέση με την προηγούμενη σειρά 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 ML πρέπει να είναι σε ένα αρχείο και να δουλεύουν σε SML/NJ v110.79 ή σε MLton 20241230 ή σε Objective Caml version 5.3.0. Το σύστημα ηλεκτρονικής υποβολής σας επιτρέπει να επιλέξετε μεταξύ αυτών των διαλέκτων της ML.

- Τα προγράμματα σε Java μπορεί να είναι και σε πολλά αρχεία αν θέλετε, αλλά θα πρέπει να περιέχουν μια κύρια κλάση με το όνομα που ζητείται και να δουλεύουν σε Java 21 (21.0.10).

- 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.