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

Εαρινό 2015

ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ Ι

Άσκηση 2

Καταληκτική ημερομηνία και ώρα ηλεκτρονικής υποβολής: 21/6/2015, 23:59:59

Ορατότης... περιορισμένη (0.25+0.25 = 0.5 βαθμοί)

«Ορατότης Μηδέν» είναι μια –πλέον cult– ελληνική κοινωνική δραματική ταινία κινηματογράφου του Νίκου Φώσκολου, γυρισμένη το 1969, στην οποία ακούγεται το, επίσης cult, τραγούδι «Βρέχει φωτιά στη στράτα μου» και η περίφημη φράση «Όχι άλλο κάρβουνο!». Όλα αυτά θα μπορούσαν κάλλιστα να περιγράφουν την τρέχουσα κατάσταση στην Ελλάδα και το κατά πόσο φαίνεται να υπάρχει λύση για τα προβλήματά της στον ορίζοντα, αλλά καλύτερα ας ασχοληθούμε με προβλήματα τα οποία έχουν μη μηδενική ορατότητα επίλυσης. Ας εξετάσουμε για παράδειγμα τον υπολογισμό του μεγέθους της ορατότητας που έχει κάποιος όταν κοιτάει κτίρια διαφορετικού ύψους από κάποιο σημείο, ή καλύτερα, για να μην παίζει ρόλο η θέση του σημείου από τα οποία τα κοιτάει, από κάποια πλευρά του επιπέδου.

Σας δίνεται λοιπόν ένας επίπεδος χάρτης στον οποίο στην κάτω πλευρά (x συντεταγμένες) είναι ο νότος, στην αριστερή (y συντεταγμένες) είναι η δύση, και στον οποίο υπάρχουν σημειωμένα *N* παραλληλόγραμμα τα οποία αναπαριστούν κτίρια με πιθανά διαφορετικά ύψη. Τα κτίρια έχουν πλευρές που είναι πάντα παράλληλες στους άξονες των συντεταγμένων και τα ύψη τους είναι γνωστά και αναφέρονται στο χάρτη. Άρα ο χάρτης αποτελείται από συνολικά *N+1* γραμμές όπου η πρώτη έχει μόνο τον αριθμό *N* και οι επόμενες έχουν πέντε αριθμούς: τέσσερις ακεραίους Xsw, Ysw, Xne και Yne οι οποίοι αντιστοιχούν στις συντεταγμένες των γωνιών του κάθε κτιρίου, και έναν αριθμό κινητής υποδιαστολής ο οποίος αντιστοιχεί στο ύψος του κτιρίου. Αυτό που θέλουμε να κάνουμε είναι να γράψουμε δύο προγράμματα (ένα σε ML και ένα σε Java) τα οποία διαβάζουν το χάρτη και επιστρέφουν τον αριθμό των κτιρίων που είναι ορατά από κάποιον παρατηρητή που βρίσκεται στη νότια πλευρά του χάρτη, ανεξαρτήτως του σημείου από τα οποία τα κοιτάει. Τα παραδείγματα παρακάτω το κάνουν πιο σαφές.

Περιορισμοί: $1 \le N \le 420.000$, όριο χρόνου εκτέλεσης: 30 seconds, όριο μνήμης: 512 MB.

Παρακάτω δείχνουμε κάποιες πιθανές κλήσεις των προγραμμάτων σε ML και σε Java.

Σε SML/NJ

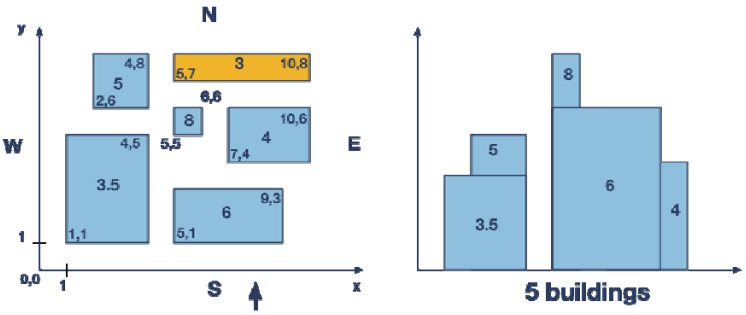
```
- oratotis "map1.txt";
val it = 5 : int

- oratotis "map2.txt";
val it = 2 : int

- oratotis "map3.txt";
val it = 4 : int
```

Σε Java

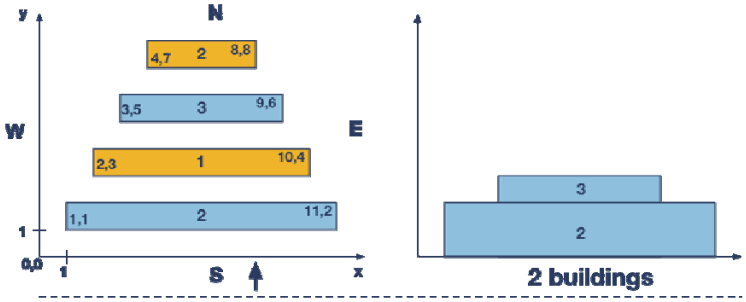
```
> java Oratotis map1.txt
5

> java Oratotis map2.txt
2

> java Oratotis map3.txt
4
```

όπου τα αρχεία με τους χάρτες είναι αυτά που φαίνονται στην επόμενη σελίδα (η εντολή **cat** είναι εντολή του Unix). Στις εικόνες φαίνονται επίσης και οι κορυφογραμμές των κτιρίων από τη νότια πλευρά.

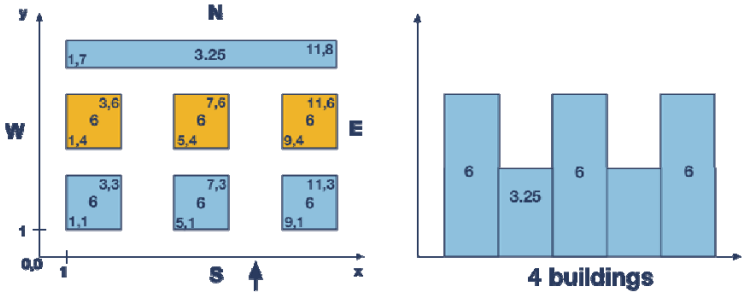
```
> cat map1.txt
6
1 1 4 5 3.5
2 6 4 8 5.0
5 1 9 3 6.0
5 5 6 6 8.0
7 4 10 6 4.0
5 7 10 8 3.0
```




```
> cat map2.txt
4
1 1 11 2 2.0
2 3 10 4 1.0
3 5 9 6 3.0
4 7 8 8 2.0
```

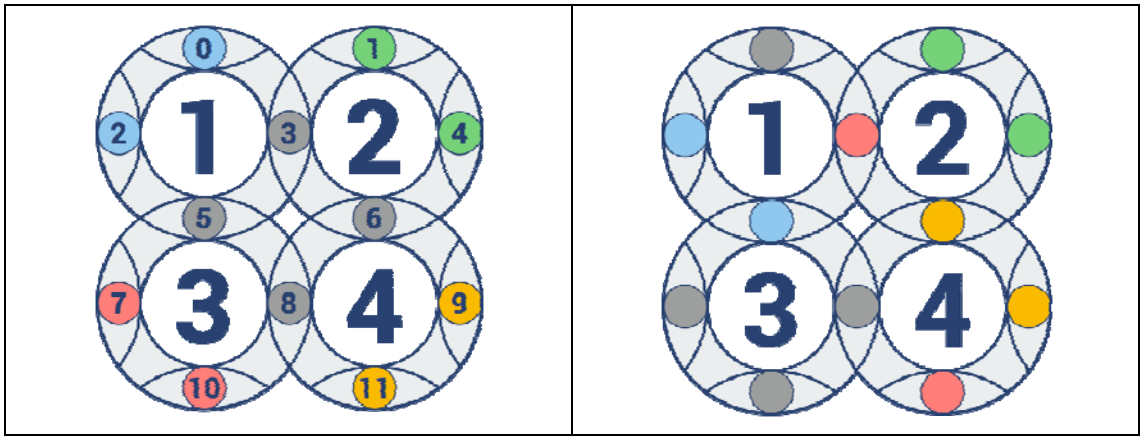



```
> cat map3.txt
7
1 1 3 3 6.0
5 1 7 3 6.0
9 1 11 3 6.0
1 4 3 6 6.0
5 4 7 6 6.0
9 4 11 6 6.0
1 7 11 8 3.25
```



Πράσινα, κόκκινα, κίτρινα, μπλε και γκρίζα σημεία διαπραγμάτευσης (0.25+0.25 = 0.5 βαθμοί)

Ζητάμε να βρούμε λύση στην παρακάτω σπαζοκεφαλιά. Τέσσερα διαπραγματευόμενα μέλη (π.χ. η Ελληνική «ομάδα διαπραγμάτευσης» και οι τρεις «θεσμοί»), τα οποία θα αναπαραστήσουμε με τους αριθμούς 1, 2, 3 και 4 και θα τους αντιστοιχίσουμε με τα τέσσερα πρώτα χρώματα του τίτλου, έχουν εμπλακεί σε μια διαπραγμάτευση δώδεκα συνολικά «σημείων» τα οποία αρχικά βρίσκονται σε κάποιον τυχαίο μπερδεμένο σχηματισμό. Θέλουν να καταλήξουν στο σχηματισμό στον οποίο τα οκτώ από τα σημεία, από δύο που είναι σημαντικά για το κάθε μέλος και έχουν το χρώμα που αντιστοιχεί σε αυτά, βρίσκονται προς την πλευρά τους, και τα μόνα σημεία που βρίσκονται ανάμεσά τους είναι τα υπόλοιπα τέσσερα σημεία που έχουν γκρίζο χρώμα. Τα παρακάτω σχήματα δείχνουν τον επιθυμητό σχηματισμό, αριστερά, και έναν τυχαίο αρχικό, δεξιά.



Αν αντιστοιχίσουμε τα χρώματα blue, green, red και yellow με τα γράμματα **b**, **g**, **r** και **y**, και το γκρίζο με το **G**, ο τελικός σχηματισμός μπορεί να αναπαρασταθεί με τη συμβολοσειρά **"bgbGgGGrGyry"** (δηλ. οι θέσεις τους είναι από πάνω προς τα κάτω, αριστερά προς τα δεξιά σε κάθε γραμμή) και αυτός της αριστερής εικόνας με τη συμβολοσειρά **"GgbrgbyGGyGr"**.

Δυστυχώς, η ευελιξία του κάθε μέλους στη διαπραγμάτευση είναι πολύ περιορισμένη. Σε κάθε βήμα μετακινείται ένα μόνο από τα τέσσερα μέλη (1, 2, 3, 4) κατά 90 μοίρες μόνο προς τα δεξιά (δηλ. κατά τη φορά των ρολογιών). Αυτό που θέλουμε να βρούμε είναι τον ελάχιστο αριθμό από μετακινήσεις που χρειάζονται να γίνουν για να καταλήξουμε στον τελικό επιθυμητό σχηματισμό.

Το παράδειγμά μας, μπορεί να λυθεί με τέσσερις κινήσεις ως εξής: το μέλος 1 μετακινείται προς τα δεξιά (90 μοίρες), μετά το μέλος 4, και τέλος το μέλος 3 δύο φορές προς τα δεξιά. Τη λύση αυτή την αναπαριστούμε με τη συμβολοσειρά **"1433"**.

Αυτό που ζητάει η άσκηση είναι να γραφούν δύο προγράμματα (ένα σε ML και ένα σε Java) τα οποία να παίρνουν ως είσοδο τη συμβολοσειρά του αρχικού σχηματισμού και επιστρέφουν ως έξοδο μια συμβολοσειρά με τις ελάχιστες κινήσεις για να επιτευχθεί ο επιθυμητός σχηματισμός.

Περιορισμοί: όριο χρόνου εκτέλεσης: 10 seconds, όριο μνήμης: 512 MB.

Στην άσκηση αυτή, η είσοδος είναι πολύ απλή και δεν έχει νόημα να βρίσκεται σε αρχείο. Παρακάτω δείχνουμε κάποιες πιθανές κλήσεις των προγραμμάτων σε ML και σε Java.

Σε SML/NJ

```
- diapragmateysi "GgbrgbyGGyGr";
val it = "1433" : string
```

Σε Java

```
$ java Diapragmateysi GgbrgbyGGyGr
1433
```

Περαιτέρω οδηγίες για την άσκηση

- 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.

- 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 v110.76 ή σε MLton 20100608 ή σε Objective Caml version 4.01.0. Το σύστημα ηλεκτρονικής υποβολής επιτρέπει να επιλέξετε μεταξύ αυτών των υλοποιήσεων της ML.

- Ο κώδικας των προγραμμάτων σε Java μπορεί να βρίσκεται σε περισσότερα του ενός αρχείου αν θέλετε αλλά θα πρέπει να μπορεί να μεταγλωττιστεί χωρίς προβλήματα με τον Java compiler με εντολές: **`javac Oratotis.java`** ή **`javac Diapragmateysi.java`**.

- Η υποβολή των προγραμμάτων θα γίνει ηλεκτρονικά μέσω του moodle, όπως και στην προηγούμενη άσκηση, και για να μπορέσετε να τις υποβάλλετε, τα μέλη της ομάδας σας (και οι δύο) θα πρέπει να έχουν ήδη λογαριασμό στο moodle. Θα υπάρξει σχετική ανακοίνωση μόλις το σύστημα υποβολής καταστεί ενεργό. Τα προγράμματά σας πρέπει να διαβάζουν την είσοδο όπως αναφέρεται και δεν πρέπει να έχουν κάποιου άλλους είδους έξοδο εκτός από τη ζητούμενη διότι δε θα γίνουν δεκτά από το σύστημα υποβολής.