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

Εαρινό 2012

ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ Ι

Άσκηση 2

Καταληκτική ημερομηνία και ώρα ηλεκτρονικής υποβολής: 16/7/2012, 12:30 ή 30/7/2012, 12:30

Ψηφοφόροι στα άκρα... (0.25 + 0.25 = 0.5 βαθμοί)

Μέχρι τις επαναληπτικές εκλογές η πόλωση των ψηφοφόρων αναμένεται να φτάσει στα άκρα. Υποθέτω ότι κάθε φορά που πολιτικοί μας θα αλληλοβρίζονται σε κάποια παράθυρα, όλο και κάποιος ψηφοφόρος θα μετακινείται προς κάποιο άκρο του πολιτικού φάσματος. Για να μην το κομματικοποιήσουμε όμως, ας θεωρήσουμε ότι οι ψηφοφόροι χωρίζονται σε αυτούς που τα βλέπουν όλα άσπρα και σε αυτούς που τα βλέπουν όλα μαύρα. Ανεξάρτητα του πώς βλέπουν τα πράγματα, ας θεωρήσουμε ότι αρχικά βρίσκονται σε κάποιο σημείο του πολιτικού φάσματος το οποίο θα θεωρήσουμε ως ένα μονοδιάστατο πίνακα **N** θέσεων.

Οι άσπροι ψηφοφόροι μπορούν είτε να μετακινηθούν κατά μία θέση προς τα αριστερά αν η θέση αυτή είναι ελεύθερη, ή να μετακινηθούν κατά δύο θέσεις προς τα αριστερά αν η αμέσως επόμενη αριστερή προς αυτούς θέση είναι κατειλημμένη και η μεθεπόμενη προς τα αριστερά τους είναι ελεύθερη (δηλαδή αν μπορούν να "πηδήξουν" κάποιον άλλο ψηφοφόρο). Οι μαύροι ψηφοφόροι μπορούν να κάνουν κάτι ανάλογο αλλά προς τα δεξιά: μπορούν είτε να μετακινηθούν κατά μια θέση κατά δεξιά τους αν η θέση είναι ελεύθερη ή να "πηδήξουν" πάνω από κάποιον άλλο ψηφοφόρο και να βρεθούν δύο θέσεις προς τα δεξιά αν η θέση στην οποία θα καταλήξουν είναι αρχικά ελεύθερη. Ο μόνος περιορισμός στις μετακινήσεις είναι ότι όλοι οι ψηφοφόροι πρέπει να παραμείνουν μέσα στα όρια του πολιτικού φάσματος.[1]

Αυτό που ζητείται είναι, με βάση κάποια αρχική κατάσταση (αρχικές θέσεις των ψηφοφόρων στο πολιτικό φάσμα), να γραφούν προγράμματα σε ML και σε Java τα οποία να επιστρέφουν (σε SML/NJ) ή να τυπώνουν (σε MLton, Objective Caml, και Java) τον ελάχιστο αριθμό κινήσεων που απαιτούνται έτσι ώστε οι ψηφοφόροι που τα βλέπουν όλα άσπρα να βρεθούν στις πιο αριστερές θέσεις του πολιτικού φάσματος και οι ψηφοφόροι που τα βλέπουν όλα μαύρα να βρεθούν στις πιο δεξιές, χωρίς να υπάρχουν κενά μεταξύ ψηφοφόρων της ίδιας απόχρωσης. Αν η αρχική κατάσταση είναι τέτοια ώστε να μην υπάρχει τρόπος να πραγματοποιηθεί αυτή η μετακίνηση, π.χ. αν δεν υπάρχουν κενές θέσεις και ψηφοφόροι διαφορετικού χρώματος κάπου είναι σε λάθος σειρά, τότε το πρόγραμμά σας πρέπει να επιστρέφει/τυπώνει **`-1`** (**`~1`** στην ML).

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

```
- moves "voters1.txt";
val it = 5 : int
- moves "voters2.txt";
val it = 0 : int
- moves "voters3.txt";
val it = ~1 : int
```

```
> java Moves voters1.txt
5
> java Moves voters2.txt
0
> java Moves voters3.txt
-1
```

Τα αρχεία εισόδου έχουν την εξής μορφή: Η πρώτη γραμμή αποτελείται από έναν ακέραιο **N** που δηλώνει το πλήθος των θέσεων του πολιτικού φάσματος. Η επόμενη γραμμή αποτελείται από ένα string με **N** το πλήθος χαρακτήρες από το σύνολο **`{b,w,e}`** όπου ο κάθε χαρακτήρας δείχνει αν η θέση καταλαμβάνεται από κάποιον μαύρο ή άσπρο ψηφοφόρο ή είναι κενή.

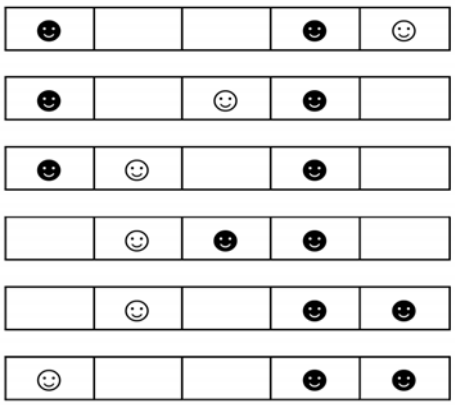
[1] Παρατηρήστε ότι οι ψηφοφόροι μετακινούνται μόνο προς την κατεύθυνση που προστάζει το χρώμα τους, αλλά διακρίσεις όσον αφορά στο χρώμα των ψηφοφόρων που πηδάνε δεν κάνουν!

Τα περιεχόμενα των αρχείων των παραδειγμάτων χρήσης των προγραμμάτων που δείξαμε παραπάνω είναι τα εξής:

```
> cat voters1.txt
5
beebw

> cat voters2.txt
3
www

> cat voters3.txt
7
wewebbw
```



Η εικόνα στα δεξιά, από πάνω προς τα κάτω, δείχνει την αρχική κατάσταση του αρχείου **voters1.txt** και τις πέντε (ελάχιστες) κινήσεις που απαιτούνται για να μετακινηθούν όλοι οι άσπροι ψηφοφόροι στις πιο αριστερές θέσεις και οι μαύροι στις δεξιότερες. Ας σημειωθεί ότι για το συγκεκριμένο test case υπάρχει και λύση με έξι κινήσεις αλλά προφανώς δεν είναι βέλτιστη.

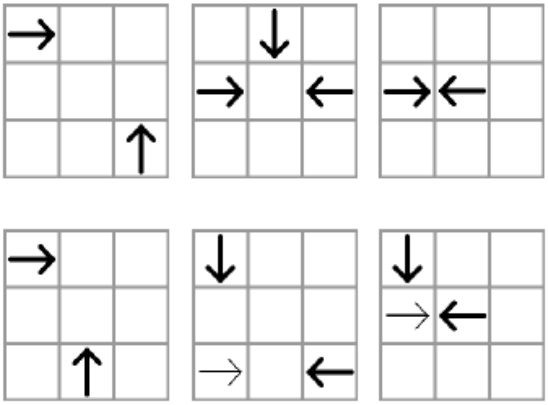
Πολιτικοί διαξιφισμοί (0.25 + 0.25 = 0.5 βαθμοί)

Το τι διαξιφισμούς αναμένεται να δούμε στα διάφορα προεκλογικά –και κατά πάσα πιθανότητα μετεκλογικά– τηλεοπτικά παράθυρα δε λέγεται. Για την ακρίβεια, το όλο σκηνικό αναμένεται να θυμίζει κάποια Ρωμαϊκή αρένα στην οποία οι συμμετέχοντες πολιτικοί μας πετάνε πέτρες ο ένας στον άλλον.

Για να το κάνουμε όμως λίγο πιο πολιτισμένο από το παραπάνω, ας υποθέσουμε ότι ο σκοπός είναι όντως οι διαξιφισμοί, δηλαδή οι πολιτικοί να διασταυρώνονται και να αναμετρούνται μεταξύ τους. Ας θεωρήσουμε λοιπόν ότι το "στούντιο" είναι ένα παραλληλόγραμμο με **C** στήλες και **R** γραμμές. Οι **N** συμμετέχοντες βρίσκονται ο καθένας σε κάποιο τετράγωνό του και είναι έτοιμοι να χρησιμοποιήσουν το επιχείρημά τους προς μία συγκεκριμένη κατεύθυνση (προς τα πάνω, προς τα κάτω, προς τα αριστερά ή τα δεξιά). Με το σύνθημα του τηλεπαρουσιαστή (τη χρονική στιγμή 0) όλοι μαζί (ως συνήθως...) εκτοξεύουν τα επιχειρήματά τους τα οποία όλα κινούνται με την ίδια ταχύτητα (ένα τετράγωνο κάθε χρονική στιγμή) και συνεχίζουν την κίνησή τους είτε μέχρι να συγκρουστούν με κάποιο άλλο επιχείρημα (ή περισσότερα του ενός) οπότε αλληλοεξοντώνονται και εξαφανίζονται, ή μέχρι να βρεθούν εκτός του παραλληλογράμμου.

Τα τρία τετράγωνα στην πρώτη σειρά δεξιά χρησιμοποιούν χοντρά βελάκια για να δείξουν τα επιχειρήματα που θα εκτοξευθούν. Όλα τα επιχειρήματα θα συγκρουστούν με άλλα (και θα εξαφανιστούν).

Αντίθετα, στο πρώτο τετράγωνο της δεύτερης σειράς, τα χοντρά βελάκια δε θα συγκρουστούν ποτέ μεταξύ τους. Στα υπόλοιπα δύο τετράγωνα, το ένα μόνο από τα χοντρά βελάκια θα συγκρουστεί με το λεπτό (εξουδετερώνοντας και τα δύο εκείνη τη χρονική στιγμή) και, λόγω της σύγκρουσής τους, το άλλο χοντρό βελάκι θα αποφύγει τη σύγκρουση.



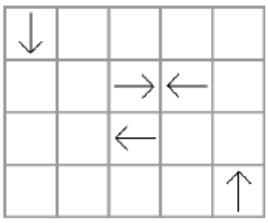
Αυτό που ζητάει η άσκηση είναι να γραφούν προγράμματα (ένα σε ML και ένα σε Java) που να δέχονται ως είσοδο ένα παραλληλόγραμμο **C×R** με τις αρχικές θέσεις των συμμετεχόντων και τις κατευθύνσεις των επιχειρημάτων τους και να υπολογίζουν το συνολικό αριθμό των τετραγώνων του παραλληλογράμμου που μπορούν τα επιχειρήματα τους να καλύψουν.

Η είσοδος πρέπει να διαβάζεται από ένα αρχείο το οποίο έχει την εξής μορφή: η πρώτη γραμμή του περιέχει τις διαστάσεις **C** και **R** του παραλληλογράμμου. Η επόμενη γραμμή περιέχει το πλήθος των συμμετεχόντων **N**. Κάθε μία από τις επόμενες **N** γραμμές του αρχείου περιέχει δύο

φυσικούς αριθμούς **X** και **Y** με τις συντεταγμένες των συμμετεχόντων ($1 \le \mathbf{X} \le \mathbf{C}$ και $1 \le \mathbf{Y} \le \mathbf{R}$) και ένα χαρακτήρα από το σύνολο **{u,d,l,r}** που αναπαριστά την κατεύθυνση στην οποία θα κινηθεί το επιχείρημά τους.

Έστω ότι το αρχείο **input.txt** έχει το παρακάτω περιεχόμενο (αντιστοιχεί στο διπλανό σχήμα):

```
5 4
5
3 3 l
3 2 r
4 2 l
5 4 u
1 1 d
```



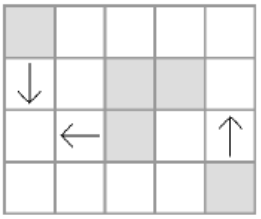
Σε ένα σύστημα Unix, η χρήση του προγράμματος σε Java δείχνει ως εξής:

```
> javac Debate.java
> java Debate input.txt
11
```

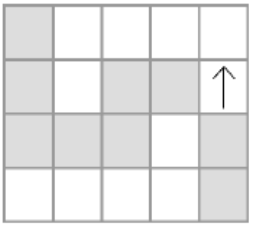
Η έξοδος του προγράμματος αντιστοιχεί στο συνολικό αριθμό από τετράγωνα που μπορούν τα επιχειρήματα να καλύψουν μετά από επαρκή χρόνο. (Δίνεται εξήγηση παρακάτω.)

Το πρόγραμμα σε SML/NJ πρέπει να έχει μια συνάρτηση **debate** η οποία δουλεύει ως εξής:

```
- debate "input.txt";
val it = 11 : int
```

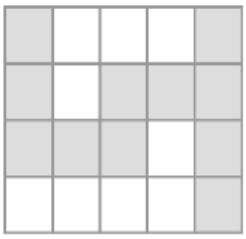


Τα δύο επιχειρήματα που είναι γειτονικά θα συγκρουστούν (τη χρονική στιγμή 0.5) και θα εξαφανιστούν. Η διπλανή εικόνα δείχνει την κατάσταση τη χρονική στιγμή 1. Τα γκρίζα τετράγωνα δείχνουν ποια τετράγωνα έχουν καλύψει τα επιχειρήματα μέχρι εκείνη τη στιγμή.



Τη χρονική στιγμή 2, το πρώτο και το πέμπτο επιχείρημα της εισόδου θα συγκρουστούν μεταξύ τους και η κατάσταση αμέσως μετά θα είναι αυτή που φαίνεται στο διπλανό σχήμα.

Φυσικά στη συνέχεια το επιχείρημα που έχει απομείνει δεν είναι δυνατό να συγκρουστεί με κάποιο άλλο οπότε θα συνεχίσει την κίνησή του μέχρι να βρεθεί εκτός του παραλληλογράμμου. Όπως φαίνεται στην τελευταία εικόνα, τα γκρίζα τετράγωνα είναι 11 συνολικά.



Ένα ακόμα αρχείο εισόδου φαίνεται παρακάτω. Η έξοδος του είναι **29**.

```
7 6
12
3 2 d
6 3 l
7 1 d
1 5 r
3 6 u
6 6 u
4 5 l
1 3 r
6 5 l
5 1 l
6 4 d
4 1 d
```

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 ασκήσεις με άλλους συμφοιτητές σας ή να βάλετε τις ασκήσεις σας σε μέρος που άλλοι μπορούν να τις βρουν εύκολα (π.χ. σε κάποια σελίδα στο διαδίκτυο, σε ιστοτόπους συζητήσεων, …).
- Τα προγράμματα σε ML πρέπει να είναι σε ένα αρχείο και να δουλεύουν σε SML/NJ v110.67 ή σε MLton 20100608 ή σε Objective Caml version 3.11.2. Το σύστημα ηλεκτρονικής υποβολής επιτρέπει να επιλέξετε μεταξύ αυτών των υλοποιήσεων της ML.
- Ο κώδικας των προγράμματα σε Java μπορεί να βρίσκεται σε περισσότερα του ενός αρχείου αλλά θα πρέπει να μπορεί να μεταγλωττιστεί χωρίς προβλήματα με τον Java compiler από το command line με εντολές της μορφής **`javac Moves.java`** ή **`javac Debate.java`**
- Η υποβολή των προγραμμάτων θα γίνει ηλεκτρονικά μέσω του moodle, όπως και στην προηγούμενη άσκηση. 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.