



## Άσκηση 2

### Σκάκι

14 Δεκεμβρίου 2016

### Στόχοι











Στόχος αυτής της άσκησης είναι η εξοικείωσή σας με πιο προχωρημένα στοιχεία του αντικειμενοστρεφούς προγραμματισμού σε Java, όπως η κληρονομικότητα και οι εξαιρέσεις. Επίσης θα χρησιμοποιήσετε τις δυνατότητες της Java για τον χειρισμό αρχείων.

### Περιγραφή

Ζητείται ένα πρόγραμμα που να υλοποιεί το γνωστό παιχνίδι **σκάκι**. Το πρόγραμμά σας δεν θα παίζει αυτόνομα το παιχνίδι αλλά θα δίνει τη δυνατότητα σε δύο παίκτες να παίζουν μεταξύ τους, υλοποιώντας (ένα μεγάλο μέρος από) τις κινήσεις και τους κανόνες του παιχνιδιού. Τα ζητούμενα περιγράφονται λεπτομερώς παρακάτω, και δεν χρειάζεται να γνωρίζετε σκάκι για να ολοκληρώσετε την άσκηση.

### Κανόνες του σκακιού

Σε μία παρτίδα σκάκι συμμετέχουν δύο παίκτες. Κάθε παίκτης έχει αρχικά 16 **κομμάτια**<sup>1</sup>, ο ένας χρώματος **άσπρου** και ο άλλος χρώματος **μαύρου**. Τα κομμάτια κάθε παίκτη είναι αρχικά 1 βασιλιάς, 1 βασίλισσα, 2 πύργοι, 2 ίπποι, 2 αξιωματικοί και 8 πιόνια, τα οποία συμβολίζονται είτε με γραφικά σύμβολα είτε με γράμματα του αγγλικού αλφαβήτου (κεφαλαία για τα άσπρα και πεζά για τα μαύρα) όπως φαίνεται στον πίνακα 1:

Κομμάτι	Σύμβολα	Γράμματα
Βασιλιάς (King)	 	K k
Βασίλισσα (Queen)	 	Q q
Πύργος (Rook)	 	R r
Ίππος (Knight)	 	N n
Αξιωματικός (Bishop)	 	B b
Πιόνι (Pawn)	 	P p

Πίνακας 1: Τα κομμάτια του σκακιού.

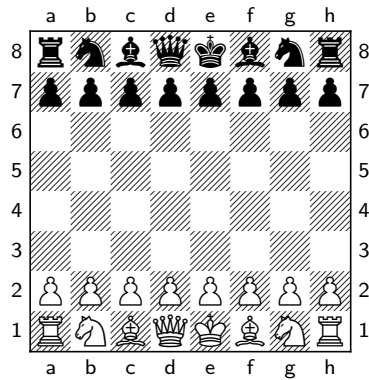
Τα κομμάτια τοποθετούνται αρχικά στην **σκακιέρα** (μία ορθογώνια διάταξη  $8 \times 8$  τετραγώνων) όπως φαίνεται στο σχήμα 1 στην επόμενη σελίδα. Οι γραμμές της σκακιέρας αριθμούνται με γράμματα του λατινικού αλφαβήτου, ενώ οι στήλες με αριθμούς.

<sup>1</sup>Ο πιο επίσημος όρος για το κομμάτι είναι *πεσός*.

```

abcdefgh
8rnbqkbnr8
7pppppppp7
6          6
5          5
4          4
3          3
2PPPPPPPP2
1RNBQKBNR1
abcdefgh

```



**Σχήμα 1:** Αρχική θέση των κομματιών στη σκακιέρα.

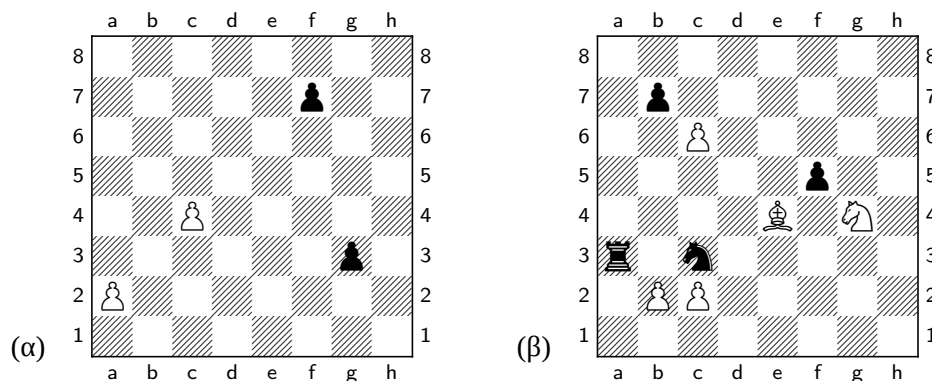
Οι παίκτες παίζουν **εναλλάξ**, μετακινώντας κάθε φορά ένα κομμάτι τους σύμφωνα με τους παρακάτω κανόνες:<sup>2</sup>

- Ο **πύργος** κινείται μόνο οριζόντια ή κατακόρυφα, σε όση απόσταση (πλήθος τετραγώνων) επιθυμεί ο παίκτης.
- Ο **ίππος** κινείται κατά μήκος τριών (3) τετραγώνων με τους εξής δύο τρόπους: (α) 1 τετράγωνο κατακόρυφα (πάνω ή κάτω) και 2 τετράγωνα οριζόντια (αριστερά ή δεξιά), ή (β) 2 τετράγωνα κατακόρυφα (πάνω ή κάτω) και 1 τετράγωνο οριζόντια (αριστερά ή δεξιά). Οι κινήσεις του μοιάζουν να σχηματίζουν ένα «Γ».
- Ο **αξιωματικός** κινείται μόνο διαγώνια, σε όση απόσταση επιθυμεί ο παίκτης.
- Η **βασίλισσα** κινείται όπως ο πύργος ή ο αξιωματικός (δηλαδή οριζόντια, κατακόρυφα ή διαγώνια).
- Ο **βασιλιάς** κινείται οριζόντια, κατακόρυφα ή διαγώνια, αλλά μόνο ένα τετράγωνο κάθε φορά.
- Όλα τα παραπάνω κομμάτια μπορούν να πραγματοποιήσουν μία κίνηση εφόσον στην τελική θέση δεν υπάρχει κομμάτι του ίδιου χρώματος. Εφόσον στην τελική θέση υπάρχει κομμάτι του αντίθετου χρώματος, η κίνηση μπορεί να πραγματοποιηθεί και τότε το «αντίπαλο» κομμάτι απομακρύνεται από τη σκακιέρα (μειώνοντας έτσι τη «δύναμη» του αντιπάλου).
- Ο πύργος, ο αξιωματικός και η βασίλισσα δεν μπορούν να υπερπηδήσουν άλλα κομμάτια, δηλαδή κατά την κίνησή τους δεν πρέπει να υπάρχουν άλλα κομμάτια (οποιοδήποτε χρώματος) μεταξύ της αρχικής και της τελικής θέσης. Αντίθετα, ο ίππος μπορεί να υπερπηδήσει οποιαδήποτε άλλα κομμάτια βρίσκονται στις ενδιάμεσες θέσεις.
- Το **πίονι** έχει τους πιο σύνθετους κανόνες κίνησης από όλα τα κομμάτια. Γενικά κινείται μόνο μπροστά (δηλαδή τα άσπρα πύονια προς τα πάνω και τα μαύρα πύονια προς τα κάτω, στο παραπάνω σχήμα), κατά 1 τετράγωνο κάθε φορά. Ειδικά κατά την πρώτη του κίνηση, ένα πύονι μπορεί να κινηθεί κατά 1 ή 2 τετράγωνα (επιλέγει ο παίκτης την κίνηση, αν δεν υπάρχει εμπόδιο). Στην τελική θέση της κίνησης του πιονιού απαγορεύεται να υπάρχει άλλο κομμάτι, ακόμα και του αντίθετου χρώματος (δηλαδή το πύονι δεν απομακρύνει αντίπαλα κομμάτια με τον συνηθισμένο τρόπο). Επίσης, αν κινηθεί κατά 2 τετράγωνα, στο ενδιάμεσο τετράγωνο απαγορεύεται να υπάρχει άλλο κομμάτι.

Επιπλέον ένα πύονι μπορεί να κινηθεί διαγώνια μπροστά (αντί για ευθεία μπροστά) αν πρόκειται με την κίνηση αυτή να απομακρύνει αντίπαλο κομμάτι από τη σκακιέρα (αυτός είναι ο μόνος τρόπος που ένα πύονι απομακρύνει αντίπαλο κομμάτι).

Όλες οι δυνατότητες κίνησης του πιονιού φαίνονται στο σχήμα 2 στην επόμενη σελίδα.

<sup>2</sup>Για τους πλήρεις κανόνες του σκακιού μπορείτε να ανατρέξετε στο [http://el.wikipedia.org/wiki/Κανόνες\\_του\\_σκακιού](http://el.wikipedia.org/wiki/Κανόνες_του_σκακιού) και στα συνοδευτικά άρθρα που περιγράφουν τα διάφορα κομμάτια. Για την άσκηση θα υλοποιήσετε μόνο όσα αναφέρονται στην εκφώνηση.



**Σχήμα 2: (α)** Το πiónι στη θέση a2 μπορεί να κινηθεί είτε στην a3 είτε στην a4 καθώς δεν έχει μετακινηθεί από την αρχική του θέση. Αντίθετα, το πiónι στη θέση c4 μπορεί να κινηθεί μόνο στη θέση c5. Παρομοίως, το πiónι στη θέση f7 μπορεί να κινηθεί είτε στην f6 είτε στην f5, ενώ το πiónι στην g3 μπορεί να κινηθεί μόνο στην g2. **(β)** Το πiónι στη θέση b2 μπορεί να κινηθεί στη θέση b3 ή στη θέση b4, ή μπορεί να απομακρύνει τον πύργο αν μετακινηθεί στη θέση a3 ή τον ίππο στη θέση c3. Το πiónι στη θέση c2 δεν μπορεί να μετακινηθεί καθόλου. Τα δύο πiónια στις θέσεις b7 και c6 απειλούν (δηλαδή μπορούν να απομακρύνουν) το ένα το άλλο, ή μπορούν να κινηθούν το καθένα προς τα εμπρός. Το πiónι στη θέση f5 μπορεί να απομακρύνει τον αξιωματικό στη θέση e4 ή τον ίππο στη θέση g4, ή να κινηθεί στη θέση f4.

Οι ακόλουθες τρεις κινήσεις είναι μέρος των κανόνων του σκακιού, αλλά η υλοποίησή τους **δεν** ζητείται:

- Μικρό και μεγάλο **ροκέ**.
- **Προαγωγή** πιονιού.
- Απομάκρυνση πιονιού **en passant** (εν διελεύσει).

Αν επιθυμείτε, μπορείτε να υλοποιήσετε και αυτές τις κινήσεις για **επιπλέον** βαθμολογία (για να βαθμολογηθεί μία κίνηση θα πρέπει να υλοποιηθεί ολοκληρωμένα). Για λεπτομέρειες, αν δεν γνωρίζετε σκάκι, ανατρέξτε στα αντίστοιχα άρθρα της (ελληνικής ή/και αγγλικής) Wikipedia.

Ένας παίκτης κερδίζει όταν περιορίσει τον αντίπαλο βασιλιά ώστε να μην μπορεί να κινηθεί χωρίς να απειλείται (**ματ**). Είναι επίσης δυνατό να τελειώσει ένα παιχνίδι ισόπαλο, αν δεν μπορεί να επιτευχθεί ματ από κανέναν παίκτη. **Δεν** ζητείται να υλοποιήσετε τίποτα σχετικό με το τέλος του παιχνιδιού, απλώς κάθε φορά οι κινήσεις θα ελέγχονται για εγκυρότητα σύμφωνα με τους παραπάνω κανόνες.

## Διεπαφή προγράμματος

Οι παίκτες θα παίζουν σκάκι μέσω της γραμμής εντολών. Κάθε παίκτης μπορεί στη σειρά του να εισαγάγει είτε μία **κίνηση** είτε μία **εντολή**.

Οι **κινήσεις** εισάγονται δίνοντας τις συντεταγμένες της θέσης του κομματιού που θα κινηθεί και τις συντεταγμένες της τελικής του θέσης (χωρίς κενά μεταξύ τους), π.χ. η κίνηση του πιονιού από την αρχική του θέση e2 κατά δύο τετράγωνα δίνεται ως e2e4. Οι κινήσεις ελέγχονται για την εγκυρότητά τους και σε περίπτωση άκυρης κίνησης τυπώνεται κατατοπιστικό μήνυμα (ο παίκτης δεν χάνει τη σειρά του). Τα μηνύματα πρέπει να διαχωρίζουν *τουλάχιστον* τις ακόλουθες περιπτώσεις άκυρων κινήσεων:

1. Συντακτικά σφάλματα, π.χ. abcd, d67e7. Συντακτικά σωστή θεωρείται μία είσοδος κίνησης όταν αποτελείται από τέσσερις χαρακτήρες, εκ των οποίων ο πρώτος και ο τρίτος είναι γράμματα και ο δεύτερος και ο τέταρτος είναι αριθμητικά ψηφία.<sup>3</sup>
2. Συντεταγμένες (αρχικής ή τελικής θέσης) εκτός ορίων, π.χ. a9b1, e2z5.
3. Μη ύπαρξη κομματιού στην αρχική θέση.
4. Κίνηση λάθος χρώματος κομματιού, π.χ. κίνηση μαύρου κομματιού όταν παίζουν τα άσπρα.
5. Μη επιτρεπόμενη κίνηση για το κομμάτι, π.χ. κίνηση για πiónι d2d5. Σε αυτή την περίπτωση πρέπει

<sup>3</sup>Για τέτοιους ελέγχους δείτε την κλάση Character της Java.

να φαίνεται και ο τύπος του κομματιού που κινείται, και είναι επιθυμητό το μήνυμα να παρέχει όσο το δυνατόν περισσότερες πληροφορίες για το σφάλμα της κίνησης, π.χ. «Ο πύργος μπορεί να κινηθεί μόνο οριζόντια ή κατακόρυφα».

Οι **εντολές** δίνονται βάζοντας άνω και κάτω τελεία πριν το αντίστοιχο γράμμα, ώστε να ξεχωρίζουν εύκολα από τις κινήσεις. Οι εντολές που ζητούνται είναι οι εξής:

**:h – Εμφάνιση κειμένου βοήθειας**

Θα εμφανίζεται ένα κείμενο που θα επεξηγεί τη χρήση του παιχνιδιού, περιγράφοντας σύντομα τον τρόπο εισαγωγής κινήσεων και εντολών και τις διαθέσιμες εντολές.

**:s – Αποθήκευση παιχνιδιού**

Αποθηκεύει τις (έγκυρες) κινήσεις του παιχνιδιού μέχρι εκείνο το σημείο. Ζητείται το όνομα του αρχείου στο οποίο θα γίνει η αποθήκευση.

**:o – Φόρτωση παιχνιδιού**

Φορτώνει ένα παιχνίδι που έχει προηγουμένως αποθηκευτεί και επιτρέπει τη **συνέχισή** του από το σημείο της αποθήκευσης. Εφόσον μία παρτίδα έχει αρχίσει, ο χρήστης ερωτάται πρώτα αν επιθυμεί να τη διακόψει. Ζητείται το όνομα του αρχείου από το οποίο θα γίνει η φόρτωση.

**:x – Έξοδος**

Τερματισμός του προγράμματος. Εφόσον μία παρτίδα έχει αρχίσει, ο χρήστης ερωτάται πρώτα αν επιθυμεί να τη διακόψει.

Μετά από κάθε κίνηση ή εντολή θα εμφανίζεται η τρέχουσα κατάσταση της σκακιέρας, στη μορφή κειμένου που φαίνεται στο σχήμα **1** (αριστερά).

Γενικά φροντίστε να κάνετε την εφαρμογή σας όσο το δυνατόν πιο **φιλική** προς τους χρήστες: χειριστείτε λάθος κινήσεις ή εντολές εκτυλώνοντας κατάλληλα μηνύματα σφάλματος, χειριστείτε περιπτώσεις που δεν δίνεται καμία είσοδος, κ.λπ.

## Υλοποίηση

Σε αυτή την ενότητα παρουσιάζονται όλες οι κλάσεις που οφείλετε να αναπτύξετε. Οι περιγραφές των κλάσεων δίνονται σε διαφορετικά επίπεδα λεπτομέρειας. Τα πεδία δίνονται περιγραφικά και θα πρέπει να αποφασίσετε τον τύπο τους. Οι μέθοδοι δεν δίνονται όλες και θα πρέπει να τις συμπληρώσετε με βάση τις προδιαγραφές που παρέχονται. Όσες μέθοδοι δίνονται θα πρέπει να υλοποιηθούν και να χρησιμοποιηθούν χωρίς μεταβολές.

Για να ολοκληρώσετε την υλοποίηση οφείλετε να κατανοήσετε τις αρμοδιότητες κάθε κλάσης. Δεν αρκεί να υλοποιήσετε τη ζητούμενη λειτουργικότητα με οποιονδήποτε τρόπο, πρέπει να ακολουθήσετε την ανάλυση του προβλήματος στις κλάσεις που περιγράφονται.

### **enum Color**

Μία απαρίθμηση (enumeration)<sup>4</sup> με τιμές WHITE και BLACK για τον καθορισμό του χρώματος κάθε κομματιού.

Περιέχει την εξής μέθοδο:

**Color nextColor()**

Επιστρέφει το «επόμενο» του τρέχοντος χρώματος (δηλαδή το χρώμα που θα παίξει στην επόμενη κίνηση), π.χ.

```
Color a = Color.BLACK;  
Color b = a.nextColor(); // Το b έχει τιμή WHITE  
Color c = b.nextColor(); // Το c έχει τιμή BLACK
```

---

<sup>4</sup>Στις απαριθμήσεις θα αναφερθούμε σε επόμενο μάθημα.

## Location

Μία κλάση που αντιπροσωπεύει μία θέση της σκακιάρας. Χρησιμοποιείται για να διευκολύνει τις μετατροπές από και προς τον συμβολισμό κειμένου της θέσης (π.χ. e2). Ένα αντικείμενο αυτού του τύπου θα αντιστοιχεί ακριβώς σε μία θέση της σκακιάρας και η αντιστοίχιση δεν θα μπορεί να αλλάξει (δεν θα υπάρχουν μέθοδοι `set . . . ( )`).

Περιέχει τις εξής μεθόδους:

### **Location(int r, int c)**

Μέθοδος δημιουργίας, δέχεται τον αριθμό γραμμής και στήλης της θέσης. Θα σας εξυπηρετήσει να αποθηκεύσετε τους αριθμούς γραμμής και στήλης ως ακεραίες τιμές 0–7. Για τη μέθοδο αυτή υποθέστε ότι οι δύο παράμετροι έχουν έγκυρες τιμές.

### **Location(String loc)**

Μέθοδος δημιουργίας, δέχεται μία συμβολοσειρά της μορφής «e2» και την αναλύει δίνοντας κατάλληλες τιμές στα πεδία του αντικειμένου για τη γραμμή και τη στήλη. Υποθέστε ότι η συμβολοσειρά `loc` έχει μήκος 2 (ακριβώς) αλλά άγνωστη μορφή. Σε περίπτωση σφάλματος, η μέθοδος προκαλεί μία εξαίρεση του τύπου `InvalidLocationException` (βλ. παρακάτω).

### **int getRow(), int getCol()**

Επιστρέφουν τη γραμμή και τη στήλη της θέσης, στο εύρος 0–7.

### **String toString()**

Επιστρέφει τον συμβολισμό κειμένου της θέσης, π.χ. «e2».

## Piece

Αφηρημένη κλάση που αντιπροσωπεύει ένα κομμάτι του παιχνιδιού. Κάθε κομμάτι χαρακτηρίζεται από το **χρώμα** και τη **θέση** του, και επίσης περιέχει μία αναφορά προς τη **σκακιάρα** στην οποία ανήκει (κλάση `Board`, βλ. παρακάτω). Κατά τη δημιουργία ενός κομματιού δίνονται τα τρία αυτά στοιχεία του. Το χρώμα και η σκακιάρα του κομματιού δεν μπορούν να αλλάξουν, ενώ η θέση του (προφανώς) μπορεί να αλλάξει.

Με βάση τα παραπάνω διαμορφώστε μεθόδους δημιουργίας και πρόσβασης στα πεδία της κλάσης (**μόνο όσες χρειάζονται**).

Επίσης η κλάση δηλώνει την ακόλουθη *αφηρημένη* μέθοδο:

### **void moveTo(Location newLoc)**

Υλοποιεί την κίνηση του κομματιού σε μία νέα θέση. Εφόσον η κίνηση είναι έγκυρη την πραγματοποιεί, αλλάζοντας τη θέση του κομματιού στη σκακιάρα, απομακρύνοντας αντίπαλο κομμάτι που τυχόν υπάρχει στην τελική θέση, κ.λπ. Σε διαφορετική περίπτωση προκαλεί μία εξαίρεση του τύπου `InvalidMoveException` (βλ. παρακάτω) με κατάλληλα χαρακτηριστικά.

## King, Queen, Rook, Knight, Bishop, Pawn

Υποκλάσεις της `Piece` που αντιπροσωπεύουν συγκεκριμένους τύπους κομματιών.

Υλοποιούν την αφηρημένη μέθοδο `moveTo ( )` της `Piece`, και επίσης τη μέθοδο

### **String toString()**

Επιστρέφει ένα χαρακτήρα που αντιπροσωπεύει το κομμάτι, σύμφωνα με τον πίνακα 1, π.χ. «κ» για τον μαύρο βασιλιά.

## Board

Η κλάση αυτή αντιπροσωπεύει τη σκακιάρα. Περιέχει τα κομμάτια του παιχνιδιού — ίσως η πιο εύκολη αναπαράσταση είναι με έναν πίνακα δύο διαστάσεων του οποίου τα στοιχεία είναι κομμάτια (`Piece`). Προσέξτε εδώ ότι η σχέση των κομματιών και της σκακιάρας είναι αμφίδρομη (η σκακιάρα

περιέχει τα κομμάτια, αλλά και τα κομμάτια γνωρίζουν τη σκακιέρα στην οποία βρίσκονται), κάτι που είναι απαραίτητο για την υλοποίηση του παιχνιδιού.

Η κλάση υλοποιεί τουλάχιστον τις ακόλουθες μεθόδους που έχουν σχέση με τον χειρισμό των κομματιών πάνω στη σκακιέρα:

#### **void init()**

Δημιουργεί τα κομμάτια που αντιστοιχούν στην αρχή του παιχνιδιού και τα τοποθετεί στις αρχικές θέσεις τους (σχήμα 1).

#### **Piece getPieceAt(Location loc)**

Επιστρέφει το κομμάτι που βρίσκεται στη θέση loc.

#### **void movePiece(Location from, Location to)**

Υλοποιεί στη σκακιέρα την κίνηση του κομματιού που βρίσκεται στη θέση from, στη θέση to. Όταν καλείται αυτή η μέθοδος, έχει ήδη επιβεβαιωθεί η εγκυρότητα της κίνησης. Στη θέση to δεν υπάρχει (αντίπαλο) κομμάτι, άρα δεν γίνεται απομάκρυνση κομματιού.

#### **void movePieceCapturing(Location from, Location to)**

Όπως και η προηγούμενη, με τη διαφορά ότι στην τελική θέση to υπάρχει (αντίπαλο) κομμάτι, το οποίο απομακρύνεται από τη σκακιέρα.

#### **boolean freeHorizontalPath(Location from, Location to)**

Επιστρέφει true αν η διαδρομή από τη θέση from στη θέση to είναι ελεύθερη. Χρησιμοποιείται κατά τον έλεγχο εγκυρότητας της κίνησης του πύργου και της βασίλισσας. Η αρχική και η τελική θέση δεν ελέγχονται (καθώς στην αρχική θέση βρίσκεται το κομμάτι που κινείται, και η κατάσταση της τελικής θέσης ελέγχεται χωριστά). Όταν καλείται η μέθοδος, έχει ήδη επιβεβαιωθεί ότι οι θέσεις from και to βρίσκονται στην ίδια γραμμή της σκακιέρας.

#### **boolean freeVerticalPath(Location from, Location to)**

Όπως η προηγούμενη μέθοδος, αλλά για τον έλεγχο κάθετης διαδρομής. Ισχύουν αντίστοιχες υποθέσεις για την αρχική και την τελική θέση.

#### **boolean freeDiagonalPath(Location from, Location to)**

Όπως η προηγούμενη μέθοδος, αλλά για τον έλεγχο διαγώνιας διαδρομής της μορφής «/», π.χ. a2c4 ή e7b4. Ισχύουν αντίστοιχες υποθέσεις για την αρχική και την τελική θέση.

#### **boolean freeAntidiagonalPath(Location from, Location to)**

Όπως η προηγούμενη μέθοδος, αλλά για τον έλεγχο διαγώνιας διαδρομής της μορφής «\», π.χ. b6g1 ή d3a6. Ισχύουν αντίστοιχες υποθέσεις για την αρχική και την τελική θέση.

Επίσης υλοποιεί τη μέθοδο:

#### **String toString()**

Επιστρέφει την αναπαράσταση της σκακιέρας σε μορφή κειμένου, όπως φαίνεται στο σχήμα 1 (αριστερά).

### **Game**

Η κλάση αντιπροσωπεύει μία παρτίδα σκάκι. Αποθηκεύει (τουλάχιστον) τη **σκακιέρα** και γνωρίζει τον **παίκτη** που παίζει κάθε στιγμή.

Εκτός πιθανής μεθόδου δημιουργίας, μόνη public μέθοδος της κλάσης είναι η ακόλουθη:

#### **void play()**

Υλοποιεί ένα παιχνίδι σκακιού, σύμφωνα με τη διεπαφή που περιγράφηκε παραπάνω. Χειρίζεται την είσοδο του χρήστη και ενημερώνει για την εξέλιξη του παιχνιδιού.

Για καλύτερη δόμηση του προγράμματος, η κλάση αυτή θα περιέχει και private μεθόδους για την υλοποίηση των λειτουργιών του προγράμματος, τουλάχιστον τις ακόλουθες:

#### **void handleMove(String moveString)**

Όταν η είσοδος του χρήστη δεν ξεκινάει από «:» υποτίθεται ότι είναι η κίνησή του. Η μέθοδος αυτή

δέχεται όλη αυτή την είσοδο του χρήστη και την επεξεργάζεται κατάλληλα ώστε, αν είναι έγκυρη να πραγματοποιήσει την αντίστοιχη κίνηση, ή διαφορετικά να χειριστεί τις διάφορες εξαιρέσεις που θα προκληθούν.

#### **void saveGame()**

Ζητάει το όνομα του αρχείου στο οποίο θα αποθηκευτεί η παρτίδα μέχρι εκείνο το σημείο, και επιχειρεί να πραγματοποιήσει την αποθήκευση. Χειρίζεται τυχόν εξαιρέσεις που θα προκύψουν (π.χ. αν δεν μπορεί να γράψει στο αρχείο, εγκαταλείπει τη διαδικασία εκτυπώνοντας πληροφοριακό μήνυμα).

#### **void openGame()**

Εφόσον έχει ήδη ξεκινήσει μία παρτίδα, ο χρήστης ερωτάται αν θέλει να τη διακόψει. Σε καταφατική απάντηση, ζητάει το όνομα του αρχείου από το οποίο θα διαβάσει τη σειρά των κινήσεων μίας αποθηκευμένης παρτίδας, και επιχειρεί να τις φορτώσει (και να τις εφαρμόσει σε μία νέα παρτίδα ώστε να μπορεί αυτή να συνεχιστεί από το σημείο που αποθηκεύθηκε).

#### **boolean exitGame()**

Εφόσον έχει ήδη ξεκινήσει μία παρτίδα, ο χρήστης ερωτάται αν θέλει να τη διακόψει. Σε καταφατική απάντηση, το πρόγραμμα τερματίζεται.

#### **void printHelp()**

Εκτυπώνει ένα σύντομο κείμενο με βασικές οδηγίες χρήσης του προγράμματος.

### ***InvalidLocationException, InvalidMoveException***

Ο χειρισμός όλων των σφαλμάτων στο πρόγραμμα θα γίνεται μέσω χειρισμού εξαιρέσεων. Από τα σφάλματα που διαχωρίζονται στην ενότητα «Διεπαφή προγράμματος», τα δύο πρώτα θα παράγουν εξαιρέσεις του τύπου `InvalidLocationException`, ενώ τα τρία επόμενα θα παράγουν εξαιρέσεις `InvalidMoveException`. Προφανώς οι εξαιρέσεις που παράγονται θα πρέπει να συλλαμβάνονται μέσα στον κώδικά σας ώστε να αντιμετωπίζονται τα σφάλματα.

### ***Αποθήκευση / φόρτωση παρτίδας***

Για την αποθήκευση μίας παρτίδας είναι απαραίτητο να έχουν φυλαχθεί κάπου όλες οι (έγκυρες) κινήσεις που έχουν πραγματοποιηθεί μέχρι στιγμής. Πρέπει να αποφασίσετε εσείς **πού** (σε ποια κλάση) και **με ποιον τρόπο** (σε ποια μορφή) θα φυλάξετε τις κινήσεις.

Επίσης μπορείτε να αποφασίσετε ελεύθερα τον τρόπο με τον οποίο θα αποθηκεύσετε την ακολουθία των κινήσεων στο αρχείο καθώς και τον τύπο του αρχείου, επιλέγοντας ό,τι θεωρείτε ότι σας διευκολύνει περισσότερο.

### ***Διαδικαστικά***

- Μπορείτε να εργαστείτε **μόνοι σας ή σε ομάδες των δύο ατόμων**. Σε περίπτωση ομάδας, την άσκηση θα παραδώσει στο e-class μόνο το ένα από τα δύο μέλη.
- Για αυτή την άσκηση ζητείται να παραδώσετε τον κώδικα που υλοποιεί τα παραπάνω. Συμπιέστε όλο το φάκελο του έργου του Netbeans σε ένα αρχείο .zip το οποίο θα παραδώσετε ηλεκτρονικά **μέσω του e-class**. Ως παράδειγμα του περιεχομένου του αρχείου μπορείτε να δείτε τα αρχεία στην ενότητα «Προγράμματα διαλέξεων» στα «Έγγραφα» του e-class.
- Στο συμπιεσμένο αρχείο θα συμπεριλάβετε ένα **αρχείο κειμένου** με όνομα «Students.txt» που θα περιέχει τα στοιχεία σας (ονοματεπώνυμο και αριθμό μητρώου).
- Ασκήσεις που δεν θα ακολουθήσουν αυτές τις οδηγίες παράδοσης δεν θα βαθμολογηθούν. Μην παραδίδετε project από άλλο περιβάλλον (π.χ. Eclipse). Μην παραδίδετε μόνο τα αρχεία του κώδικα ή μόνο τον φάκελο src. Μην παραδίδετε τον κώδικα σε οποιαδήποτε άλλη μορφή (π.χ. μέσα σε αρχείο του Word).

- Η συνεργασία ενθαρρύνεται, όμως η αντιγραφή απαγορεύεται! Σε περίπτωση αντιγραφής, όλες οι όμοιες ασκήσεις θα **μηδενιστούν**.
- Η άσκηση πρέπει να είναι αποτέλεσμα της δικής σας προσπάθειας. Σε περίπτωση ομαδικής εργασίας, πρέπει και τα δύο μέλη της ομάδας να έχουν συμμετάσχει ουσιαστικά στη δημιουργία της άσκησης. Φοιτητές που δεν έχουν συμμετάσχει στην εκπόνηση της άσκησης που θα παραδώσουν, θα μηδενιστούν.
- Ενδέχεται να πραγματοποιηθεί **εξέταση** των ασκήσεων που θα παραδοθούν, με τρόπο που θα ανακοινωθεί αργότερα. Φοιτητές που δεν θα προσέλθουν στην εξέταση δεν θα βαθμολογηθούν.
- Προθεσμία παράδοσης: **Παρασκευή 13 Ιανουαρίου 2017**. Για κάθε ημέρα καθυστέρησης θα υπάρξει μείωση της βαθμολογίας κατά 10%. Υπενθυμίζεται ότι για να περάσετε το μάθημα θα πρέπει να έχετε προβιβάσιμο μέσο όρο στις ασκήσεις.
- Για οποιαδήποτε απορία γράψτε τις απορίες σας στην σχετική «Περιοχή συζητήσεων» στο e-class ώστε να επωφεληθούν όλοι οι συμφοιτητές σας. **Δεν** θα απαντηθούν απορίες μέσω email.