



## Γενικές Οδηγίες και Περιγραφή των Παραδοτέων της Εργασίας

### Φάσεις

- **Φάση Α: 10% τελικού βαθμού** (ρήτρα 5 για επιτυχία στο μάθημα)
  - 18 Νοεμβρίου - 5 Δεκεμβρίου 2023
- **Φάση Β: 15% τελικού βαθμού** (ρήτρα 5 για επιτυχία στο μάθημα)
  - 6 Δεκεμβρίου - 21 Δεκεμβρίου 2023 (bonus 5%) ή
  - 22 Δεκεμβρίου - 7 Ιανουαρίου 2024 (χωρίς bonus)

**Παρατάσεις μπορεί να δοθούν μόνο αν κριθεί σκόπιμο και το πολύ 3 μέρες για τη Φάση Α και για τη Φάση Β!**

### Φάση Α

Σε αυτή τη φάση πρέπει να γίνει ο σχεδιασμός της εφαρμογής βάσει των ιδεών και των αρχών του αντικειμενοστρεφούς προγραμματισμού που έχετε διδαχτεί. Αποτέλεσμα αυτής της φάσης είναι ο καθορισμός των αντικείμενων, των χαρακτηριστικών και της συμπεριφοράς τους που απαιτούνται για να αναπαραστήσουν τις καταστάσεις και τις λειτουργίες του θέματος της εργασίας, όπως έχουν περιγραφεί στην εκφώνηση.

Παραδοτέα αυτής της φάσης είναι :

- **Γραπτή αναφορά** (όχι σε greeklish) η οποία θα περιγράφει τα παραπάνω στοιχεία και θα παρουσιάζει το σχέδιο υλοποίησης της προγραμματιστικής εργασίας, έτσι ώστε να είναι έτοιμο το πέρασμα στην επόμενη φάση της υλοποίησης. Θα πρέπει να συμπεριλαμβάνονται και **UML class diagrams**.
- **Πηγαίος κώδικας** που περιλαμβάνει τις **διεπαφές** (interfaces) και το περίγραμμα των κλάσεων (class outline) Java του προγράμματος σας, συνοδευόμενες από τα απαραίτητα **javadoc** σχόλια, τα οποία θα καθοδηγήσουν την υλοποίηση της επόμενης φάσης.

Επιγραμματικά, οι σημαντικότερες εργασίες που πρέπει να γίνουν σε αυτή τη φάση είναι:

- Αναγνώριση των κλάσεων και διεπαφών για κάθε μικρή και μεγάλη συνιστώσα του προγράμματος. Αναγνώριση των ευθυνών κάθε κλάσης και των πιθανών σχέσεων της με άλλες.
- Εύρεση των χαρακτηριστικών και των μεθόδων κάθε κλάσης.
- Εύρεση της συμπεριφοράς (behaviour) κάθε κλάσης και διεπαφής, καθώς και της επικοινωνίας μέσω μηνυμάτων (method calls) που χρειάζεται να έχουν μεταξύ τους.
- Οργάνωση των κλάσεων σε ιεραρχίες με στόχο την μέγιστη δυνατή επαναχρησιμοποίηση του κώδικα σας.
- Για κάθε κλάση που υλοποιεί μια διεπαφή δώστε τις υπογραφές (signatures) για όλες τις μεθόδους και τις εκ των προτέρων, εκ των υστέρων και αμετάβλητες συνθήκες (preconditions, postconditions, invariants) που τις διέπουν σε μορφή javadoc σχολίων.

**ΣΗΜΑΝΤΙΚΟ:** Σημειώστε ότι όσο πληρέστερη και αναλυτικότερη δουλειά κάνετε στην σχεδίαση τόσο πιο σωστή, εύκολη και επιτυχημένη θα είναι η υλοποίηση. Επειδή τα προηγούμενα χρόνια παρατηρήσαμε ότι όσοι δεν είχαν κάνει καλή Φάση Α σπάνια έκαναν καλή Φάση Β, από πρόπερσι υπάρχει η ρήτρα 5 και στη Φάση Α, ελπίζοντας ότι αυτό θα σας βοηθήσει να κάνετε καλύτερη κατανομή του χρόνου σας και να μεγιστοποιήσετε τα οφέλη από το μάθημα και τον τελικό βαθμό που θα επιτύχετε.

#### UML Plugins

Μπορείτε να συμβουλευτείτε το link που βρίσκεται στην σελίδα του μαθήματος στο section Project : “ Πληροφορίες για UML tools/plugins”

## Αναλυτική βαθμολόγηση για τη Α φάση

Παραδοτέα αυτής της φάσης είναι :

Παραδοτέα	Μονάδες
Γραπτή αναφορά (όχι σε greeklish) η οποία θα περιγράφει τα παραπάνω στοιχεία και θα παρουσιάζει το σχέδιο υλοποίησης της προγραμματιστικής εργασίας, έτσι ώστε να είναι έτοιμο το πέρασμα στην επόμενη φάση της υλοποίησης. Στο moodle υπάρχει ενδεικτικό template για την γραπτή αναφορά της εργασίας σας.	20 %
Θα πρέπει να συμπεριλαμβάνονται και UML class diagrams και να εξηγείται τι απεικονίζει το κάθε διάγραμμα.	10 %
Χωρισμός εργασίας σύμφωνα με το μοντέλο MVC (Model View Controller)	15 %
Για κάθε κλάση που υλοποιεί μια διεπαφή δώστε τις υπογραφές (signatures) για όλες τις μεθόδους και τις εκ των προτέρων, εκ των υστέρων και τις αμετάβλητες συνθήκες (pre-conditions, post-conditions και invariants αντίστοιχα) που τις διέπουν σε μορφή javadoc σχολίων.	15 %
Οργάνωση των κλάσεων σε ιεραρχίες με στόχο την μέγιστη δυνατή επαναχρησιμοποίηση του κώδικα. Χρησιμοποίηση abstract κλάσεων ή/και interfaces για τις βασικές κλάσεις Card και Square.	15 %
Να υπάρχουν οι κλάσεις και οι απαραίτητες μέθοδοι (25%) Square, Card, Deck, Pawn, Player, Controller, View Πρέπει να υπάρχουν οι μέθοδοι για τα βασικά στοιχεία του παιχνιδιού  Ενδεικτικά κάποιες βασικές μέθοδοι ( <b>θα πρέπει να ορίσετε και αρκετές ακόμα</b> ) Μέθοδοι για την αρχικοποίηση του παιχνιδιού (ταμπλό, κάρτες, παίκτες, πιόνια) Μέθοδος που καθορίζει τη σειρά (ποιος παίκτης παίζει) Μέθοδοι για την παραλαβή κάρτας Μέθοδοι για την κίνηση των πιονιών ανάλογα με την κάθε κάρτα Μέθοδος αν ο παίκτης μπορεί να επιλέξει fold. Μέθοδος που ελέγχει αν τελείωσε το παιχνίδι/νικητής	25 %
<b>ΣΥΝΟΛΟ</b>	<b>100</b>

Αναφορές που κάνουν μόνο copy paste τον κώδικα από netbeans/eclipse με τα σχόλια θα παίρνουν χαμηλό βαθμό (μισό βαθμό και λιγότερο ανάλογα την περίπτωση) για να μην αδικούνται οι φοιτητές που έκαναν καλή αναφορά. Μια καλή αναφορά πρέπει να έχει

προφανώς τα στοιχεία σας και να είναι οργανωμένη σε ενότητες (καλό είναι να έχει αριθμημένες σελίδες και πίνακα περιεχομένων στην αρχή).

## **Φάση Β**

Σε αυτή τη φάση πρέπει να γίνει η κυρίως υλοποίηση της εφαρμογής, βάσει της σχεδίασης που έχει προηγηθεί (φάση Α). Δεν επιβάλλεται να χρησιμοποιηθεί αυτούσια η σχεδίαση της φάσης Α, καθώς κάποιες σχεδιαστικές επιλογές αποδεικνύεται στην πορεία ότι χρειάζονται αναθεώρηση. Εντούτοις, η τελική βαθμολογία θα εξαρτηθεί και από τη συνέπεια της τελικής υλοποίησης ως προς την αρχική σχεδίαση.

Σε αυτή τη φάση, παραδοτέα είναι :

- ο **πηγαίος κώδικας** που υλοποιεί την εργασία
- αναλυτικές οδηγίες για το πώς μεταγλωττίζεται και πώς τρέχει το πρόγραμμά σας (README, Ant, Maven κλπ)
- αναφορά, στην οποία θα αναλύεται :
  - ο η τελική σχεδίαση της εφαρμογής,
  - ο ποιές αλλαγές έγιναν σε σχέση με τη σχεδίαση της Α' φάσης (και γιατί),
  - ο οι αλγόριθμοι που χρησιμοποιήθηκαν
  - ο τυχόν διαφοροποιήσεις στους κανόνες σε σχέση με τους κανόνες που δίνονται παραπάνω
  - ο οι σχεδιαστικές και προγραμματιστικές αποφάσεις που ελήφθησαν και πώς αυτό αντανακλάται στον τελικό χρήστη (π.χ. ευκολία/δυσκολία χειρισμού)
  - ο τα προβλήματα που αντιμετωπίστηκαν
  - ο τα JUnit tests που φτιάχτηκαν για τον έλεγχο της ορθότητας
  - ο γενικά ό,τι άλλο κρίνετε απαραίτητο να αναφερθεί (όπως για παράδειγμα τι ενδεχομένως δεν κάνατε).

## ***Βαθμολογία Εργασίας***

Για τη βαθμολογία της εργασίας σας θα συνεκτιμηθούν:

- εάν (και πόσο) η σχεδίαση της εφαρμογής εφαρμόζει τις έννοιες και τεχνικές του αντικειμενοστρεφούς προγραμματισμού που διδαχθήκατε στο μάθημα
- εάν (και πόσο) υλοποιήθηκαν οι υποχρεωτικές λειτουργίες της εφαρμογής
- η πληρότητα της τελικής αναφοράς, η οποία θα καταγράφει και θα τεκμηριώνει την σχεδίαση και υλοποίηση της εφαρμογής

Ο τρόπος βαθμολόγησης περιγράφεται αναλυτικά στην επόμενη σελίδα.

Για διευκρινήσεις σχετικά με την παραπάνω εργασία μπορείτε να στέλνετε μηνύματα με απορίες σας στο σχετικό **forum** στην ιστοσελίδα του moodle. Ερωτήσεις που στέλνονται στην λίστα του μαθήματος [hy252-list@csd.uoc.gr](mailto:hy252-list@csd.uoc.gr) **δε θα απαντώνται**.

***Καλή Εργασία***

## Αναλυτική βαθμολόγηση για τη Β φάση

Η αναλυτική βαθμολόγηση για τη Β φάση του Project θα είναι η εξής:

Παραδοτέα	Μονάδες
Αναφορά (Ανανεωμένη από την Φάση Α)	8
Σχόλια Javadoc (Pre – Post Conditions)	4
JUnit Tests	4
UML Diagrams με βάση την τελική έκδοση και επεξήγηση τους	4
<b>Λειτουργικότητα-Γραφικά</b>	
Αρχικοποίηση Παικτών-Ταμπλό-Στοιβάς Καρτών	10
Τήρηση Σειράς	7
Χρήση καρτών με αριθμό 1 & 2	7
Χρήση καρτών με αριθμό 3,5	6
Χρήση καρτών με αριθμό 8, 12	6
Χρήση καρτών με αριθμό 4 & 10	7
Χρήση κάρτας με αριθμό 7	5
Χρήση Κάρτας με αριθμό 11 (+ έλεγχος για Safety Zone)	5
Χρήση Κάρτας Sorry! (+ έλεγχος για Safety Zone)	5
Σωστή χρήση θέσης Slide	6
Έλεγχος για Fold	6
Κανόνες για πιόνια στην ίδια θέση	5
Σωστή χρήση θέσης Home- Νικητής	5
Παιχνίδι έως 4 παίκτες (Bonus 5%)	5
Αποθήκευση Παιχνιδιού (Bonus 5%)	5
<b>Σύνολο</b>	<b>110</b>

**ΣΗΜΑΝΤΙΚΟ:** Η ελάχιστη υλοποίηση που χρειάζεται κάποιος για να πάρει βαθμό 50/100 είναι να υπάρχει αναφορά, σχόλια, JUnit tests, UML Diagrams, γραφικό περιβάλλον, σωστή αρχικοποίηση του ταμπλό, των παιχτών και των καρτών και να γίνεται έστω κάποια κίνηση από τον κάθε παίκτη μαζί με τήρηση σειράς.



## Sorry! Board Game

### Project



### Εκπαιδευτικοί Στόχοι

Προδιαγραφή και σχεδίαση συστήματος  
Προδιαγραφή Αφαιρετικών Τύπων Δεδομένων (ΑΤΔ) που απαιτούνται για την επιτυχή ολοκλήρωσή του συστήματος  
Υλοποίηση εξαρτημάτων (ΑΤΔ) του συστήματος των οποίων η προδιαγραφή δίνεται  
Χρήση κληρονομικότητας και πολυμορφισμού  
Δημιουργία Γραφικής Διεπαφής  
Απεξάρτηση του πυρήνα του συστήματος από τη Γραφική Διεπαφή  
Επαναχρησιμοποίηση διεπαφών και κλάσεων  
Χρήση JFC (Java Collection Framework)  
Τεκμηρίωση, Έλεγχος

## Σύντομη Περιγραφή Εργασίας

**Στόχος.** Στην εργασία αυτή καλείστε να σχεδιάσετε και να υλοποιήσετε το επιτραπέζιο παιχνίδι “Sorry!”.

**Η εργασία είναι ατομική και απαγορεύεται η χρήση κώδικα που δεν έχετε γράψει οι ίδιοι, είτε από το διαδίκτυο είτε από κάποιο συμφοιτητή σας. Σε περίπτωση εντοπισμού αντιγραφής η εργασία θα μηδενίζεται. Αν κάτι αντιγράψετε από κάπου (όσο μικρό ή μεγάλο είναι), πρέπει να το αναφέρετε στην αναφορά.**

### Σκοπός του Παιχνιδιού.

Ο σκοπός του παιχνιδιού είναι για κάθε παίκτη να μετακινήσει τα πιόνια του από την αρχική του θέση (θέση Start) στο “σπίτι” του (θέση Home), με κινήσεις που πραγματοποιούνται με τη χρήση καρτών. Τα πιόνια μετακινούνται σύμφωνα με τη φορά των δεικτών του ρολογιού, αλλά ανάλογα με την κάρτα, μπορούν να μετακινηθούν ακόμα προς τα πίσω. Το παιχνίδι είναι ανατρεπτικό, μιας και ανάλογα με τις κινήσεις του αντιπάλου ένα πιόνι που βρίσκεται κοντά στον τερματισμό μπορεί να βρεθεί ξανά στην αρχική του θέση.

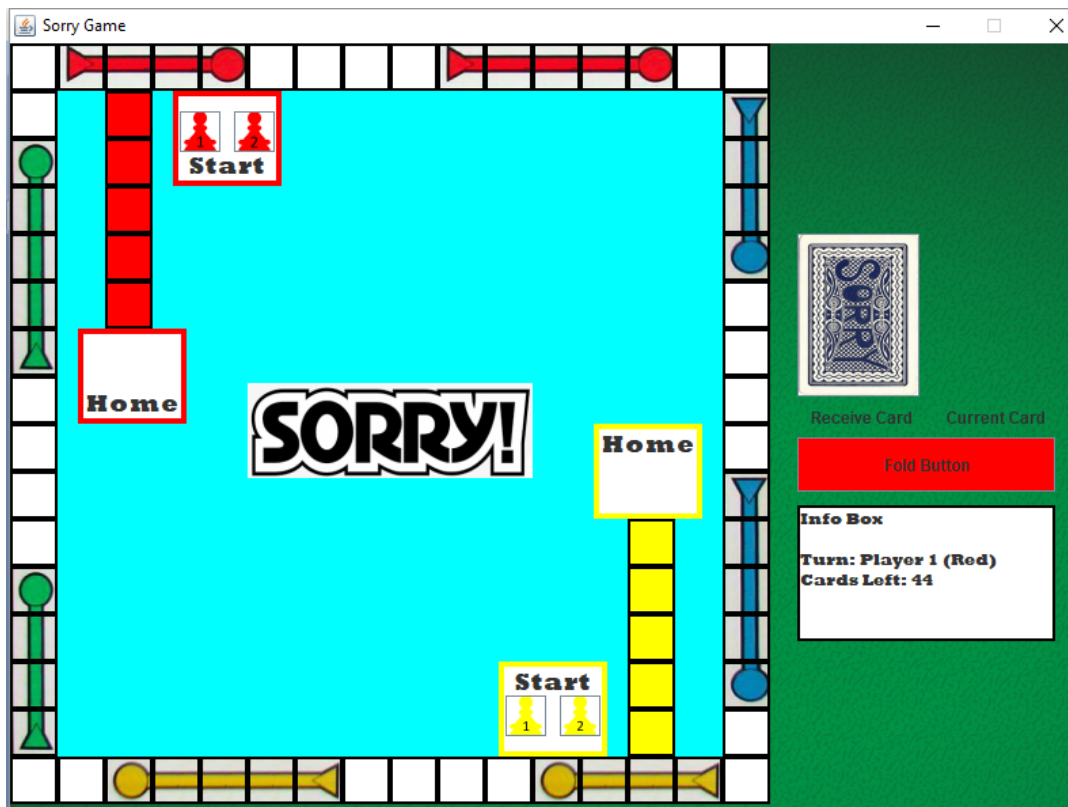
Σε σχέση με την επίσημη έκδοση του παιχνιδιού, υπάρχουν κάποιες μικρές παραλλαγές, οπότε θα πρέπει να βασιστείτε στην παρούσα εκφώνηση του project για τους κανόνες.

## Αναλυτική Περιγραφή της Εργασίας

### Περιεχόμενα παιχνιδιού

Τα παιχνίδια που σας ζητείται αποτελείται από :

- 1 ταμπλό με διαστάσεις 16x16
- 2 παίκτες (Κόκκινος Παίκτης και Κίτρινος Παίκτης)
- 2 πιόνια ίδιου χρώματος για κάθε παίκτη
- 44 Κάρτες
  - ο 4 κάρτες για κάθε έναν από τους αριθμούς 1,2,3,4,5,7,8,10,11,12
  - ο 4 κάρτες Sorry!



Εικόνα 1. Η απεικόνιση του ταμπλό στην αρχή του παιχνιδιού

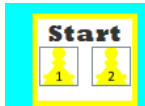

Στην Εικόνα 1 βλέπουμε το ταμπλό στην αρχή του παιχνιδιού. Τα κίτρινα πιόνια αντιστοιχούν στον έναν παίκτη και τα κόκκινα στον αντίπαλό του. Τα πιόνια μετακινούνται με τη βοήθεια των καρτών και ο σκοπός είναι ο παίκτης να μετακινήσει τα πιόνια του από τη θέση Start στη θέση Home. Δεξιά βλέπουμε τη στοίβα με τις κάρτες, ένα κουμπί fold (που χρησιμοποιείται αν ο παίκτης δε μπορεί να κάνει κίνηση) και ένα πλαίσιο που περιέχει πληροφορίες.


## Προετοιμασία του παιχνιδιού

Η αρχικοποίηση του παιχνιδιού περιλαμβάνει


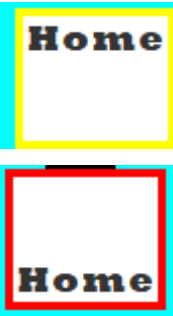

- την αρχικοποίηση του ταμπλό
- την αρχικοποίηση όλων των καρτών
- την αρχικοποίηση των παικτών και των πιονιών
- την επιλογή του ποιος παίκτης θα παίξει πρώτος (π.χ. random).

## Θέσεις στο Ταμπλό & Κανόνες

Θέση	Επεξήγηση	Εικόνα
Start (Εκκίνηση)	Υπάρχει 1 τέτοια θέση για κάθε παίκτη. Στη θέση αυτή τοποθετεί ο παίκτης τα 2 του αρχικά πιόνια.	 

<p>Slide (Τσουλήθρα )</p>	<p>Υπάρχουν 8 διαφορετικές τσουλήθρες, 2 για κάθε διαφορετικό χρώμα. Για το κάθε χρώμα υπάρχει μία τσουλήθρα με μήκος 5 τετράγωνα και μία με μήκος 4 τετράγωνα.</p> <p>Όταν ο παίκτης φτάσει στο πρώτο τετράγωνο μιας θέσης slide (έχει σήμα ένα τρίγωνο), η οποία έχει διαφορετικό χρώμα από του παίκτη, τότε ο παίκτης γλιστράει μέχρι το τέλος της “τσουλήθρας” (έχει σήμα ένα κύκλο). Αν μέσα την “τσουλήθρα” υπάρχουν άλλα πιόνια (είτε του αντιπάλου είτε του παίκτη που παίζει!), τότε αυτά γυρνάνε στην αρχική τους θέση δηλαδή στην θέση εκκίνησης τους (Start).</p> <p>Αν ο παίκτης φτάσει στο πρώτο τετράγωνο μιας θέσης slide η οποία έχει το ίδιο χρώμα με τον παίκτη, απλώς παραμένει στο τετράγωνο αυτό και δε μετακινείται στο τέλος της τσουλήθρας.</p> <p>Αν ένα πιόνι φτάσει σε ένα από τα υπόλοιπα τετράγωνα μίας θέσης slide εκτός από το πρώτο (ανεξάρτητα από το χρώμα της τσουλήθρας), τότε δε γίνεται κάποια έξτρα κίνηση. Απλώς κινδυνεύει να γυρίσει στην αρχή, αν ένα άλλο πιόνι φτάσει στο πρώτο τετράγωνο της συγκεκριμένης τσουλήθρας.</p>	
-----------------------------------	---	---


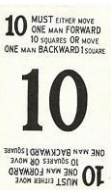





<p>Safety Zone (Περιοχή Ασφαλείας)</p>	<p>Υπάρχει 1 περιοχή ασφαλείας για κάθε παίκτη, όπου αποτελείται από 5 τετράγωνα. Ένα πιόνι μπορεί να μπει στην περιοχή ασφαλείας μόνο με κίνηση προς τα εμπρός (με τη φορά των δεικτών του ρολογιού)</p> <p>Η περιοχή ασφαλείας οδηγεί στο “Σπίτι” ενός παίκτη και μπορεί να χρησιμοποιηθεί μόνο από τον παίκτη που του ανήκει το σπίτι. Επίσης η περιοχή ασφαλείας, προστατεύει το πιόνι του παίκτη από επιθέσεις του αντιπάλου μέσω κάποιων καρτών, όπως θα δούμε και παρακάτω.</p>	
<p>Home (Σπίτι)</p>	<p>Η θέση Home (Σπίτι) είναι η τελική θέση όπου ο παίκτης καλείται να φέρει τα 2 του πιόνια και βρίσκεται στο τέλος της περιοχής ασφαλείας. <u>Μόλις φτάσει σε αυτή τη θέση ένα πιόνι, δε μπορεί να μετακινηθεί ξανά.</u> Για να φτάσει σε αυτή τη θέση ένα πιόνι, είναι απαραίτητο να προχωρήσει με τον ακριβή αριθμό των τετραγώνων που απαιτούνται, και όχι με μεγαλύτερο (παραδείγματα υπάρχουν παρακάτω στην εκφώνηση).</p>	
<p>Απλή θέση</p>	<p>Όλες οι υπόλοιπες θέσεις του ταμπλό είναι απλές, δεν έχουν κάποια επιπλέον ιδιότητα.</p>	

## Κάρτες & Κανόνες

Παρακάτω, απεικονίζονται οι κάρτες του παιχνιδιού και περιγράφονται οι κανόνες για κάθε διαφορετική κάρτα.

Κάρτα	Επεξήγηση
	Με αυτήν την κάρτα μπορεί ένα πιόνι να ξεκινήσει από τη θέση Start. Εναλλακτικά, μπορεί ο παίκτης να μετακινήσει ένα πιόνι μία θέση προς τα εμπρός (σύμφωνα με τη φορά των δεικτών του ρολογιού).
	Με αυτήν την κάρτα μπορεί ένα πιόνι να ξεκινήσει από τη θέση Start ή να μετακινήσει ένα πιόνι δύο θέσεις προς τα εμπρός (σύμφωνα με τη φορά των δεικτών του ρολογιού). Με αυτήν την κάρτα ο παίκτης πρέπει να ξαναπαίξει.
	Μετακινήστε όλα σας τα πιόνια τρεις θέσεις προς τα εμπρός (σύμφωνα με τη φορά των δεικτών του ρολογιού). Αν δεν είναι εφικτό να μετακινήσετε όλα σας τα πιόνια, τότε απλά μετακινείτε το πιόνι που μπορείτε.
	Μετακινήστε ένα πιόνι τέσσερις θέσεις <b>προς τα πίσω</b> (δηλαδή αντίθετα με τη φορά των δεικτών του ρολογιού).
	Μετακινήστε όλα σας τα πιόνια πέντε θέσεις προς τα εμπρός (σύμφωνα με τη φορά των δεικτών του ρολογιού). Αν δεν είναι εφικτό να μετακινήσετε όλα σας τα πιόνια, τότε απλά μετακινείτε το πιόνι που μπορείτε.
	Μετακινήστε ένα πιόνι επτά θέσεις προς τα εμπρός ή χωρίστε τις επτά θέσεις προς τα εμπρός ανάμεσα σε δύο πιόνια (όπως τέσσερις θέσεις για ένα πιόνι και τρεις για άλλο, πέντε θέσεις για ένα πιόνι και δύο για το άλλο ή έξι θέσεις για ένα πιόνι και μία για το άλλο). Δεν μπορείτε να μετακινηθείτε προς τα πίσω με αυτήν την κάρτα, αλλά ούτε και να ξεκινήσετε ένα πιόνι από τη θέση Start.

	<p>Μετακινήστε ένα πιόνι οκτώ θέσεις προς τα εμπρός ή τραβήξτε 1 νέα κάρτα</p>
	<p>Μετακινήστε ένα πιόνι δέκα θέσεις προς τα εμπρός ή <b>μία θέση προς τα πίσω</b>. Εάν κανένα από τα πιόνια ενός παίκτη δεν μπορεί να προχωρήσει 10 θέσεις εμπρός, τότε ένα πιόνι πρέπει αναγκαστικά να μετακινηθεί πίσω μία θέση αν φυσικά είναι εφικτή μία τέτοια κίνηση.</p>
	<p>Μετακινήστε ένα πιόνι έντεκα θέσεις προς τα εμπρός ή αν το επιθυμείτε ανταλλάξτε θέση σε ένα πιόνι σας με ένα πιόνι του αντιπάλου. Η κάρτα 11 δεν μπορεί να χρησιμοποιηθεί για:</p> <ul style="list-style-type: none"> <li>• να ανταλλάξετε ένα πιόνι του αντιπάλου που βρίσκεται στην περιοχή ασφαλείας του ή στις θέσεις Εκκίνησης και Σπίτι.</li> <li>• να ανταλλάξετε ένα πιόνι σας που βρίσκεται στις θέσεις εκκίνησης ή Σπίτι</li> </ul> <p><b>Αν δε γίνεται να μετακινηθεί ένα πιόνι 11 θέσεις μπροστά, δεν είστε υποχρεωμένοι να ανταλλάξετε ένα πιόνι σας με του αντιπάλου.</b> Σε αυτή την περίπτωση απλώς αλλάζει η σειρά (επιλέγει ο παίκτης να πατήσει fold) και παίζει ο άλλος παίκτης.</p>
	<p>Μετακινήστε ένα πιόνι δώδεκα θέσεις προς τα εμπρός ή τραβήξτε 1 νέα κάρτα.</p>
	<p>Πάρτε ένα πιόνι σας που βρίσκεται <b>στη θέση Start (και μόνο)</b> και μετακινήστε το κατευθείαν σε μία θέση που καταλαμβάνεται από το πιόνι ενός αντιπάλου, στέλνοντάς το πιόνι του αντιπάλου πίσω στη δική του θέση Start.</p> <ul style="list-style-type: none"> <li>• Η κάρτα δεν μπορεί να χρησιμοποιηθεί για πιόνι ενός αντιπάλου σε μια περιοχή ασφαλείας.</li> <li>• Αν δεν υπάρχουν πιόνια στη θέση Start του παίκτη που παίζει ή τα πιόνια του αντιπάλου δε μπορούν να μετακινηθούν (πχ αν είναι στην περιοχή ασφαλείας τους), τότε απλώς η κάρτα απορρίπτεται και αλλάζει η σειρά.</li> </ul>

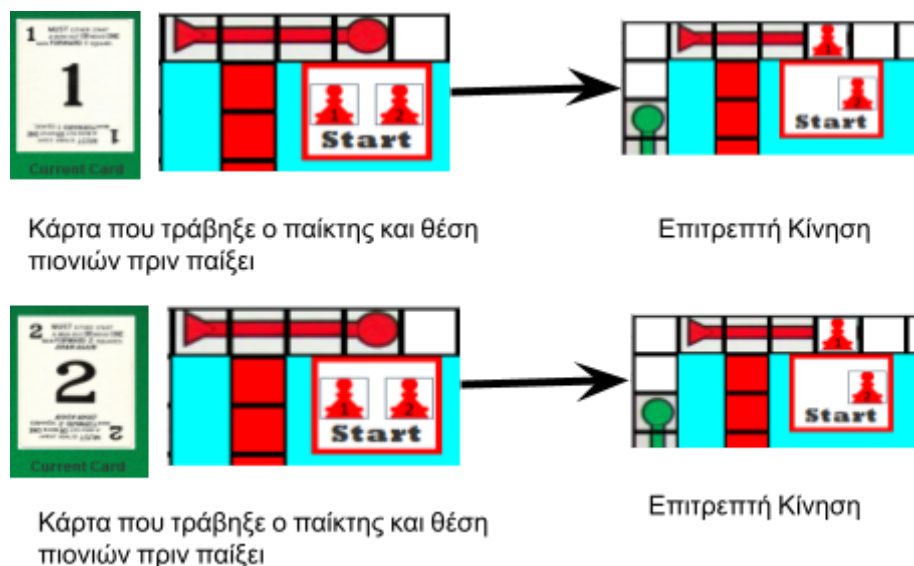
**Ποιες κινήσεις γίνονται στη σειρά κάθε παίκτη**

Στη σειρά του ο κάθε παίκτης:

1. Παίρνει την πρώτη κάρτα από τη στοίβα καρτών και κινεί ένα από τα πιόνια του (αν είναι εφικτό), ανάλογα με την κάρτα που ήρθε. Αν δε μπορεί να κάνει καμία κίνηση, τότε μπορεί να πατήσει το κουμπί fold.
2. Η κάρτα αυτή έπειτα από την κίνηση που γίνεται (ή από το fold) βγαίνει εκτός παιχνιδιού. Όμως αν τελειώσουν οι κάρτες, τότε θα πρέπει να τις ανακατέψετε (random) και να τις χρησιμοποιήσετε ξανά από την αρχή.
3. Η σειρά του παίκτη τελειώνει μετά την κίνηση του σύμφωνα με την κάρτα που τράβηξε, εκτός αν η κάρτα που τράβηξε ήταν η κάρτα 2. Σε αυτήν την περίπτωση ξαναπαίζει. Αν έρθει ξανά κάρτα με αριθμό 2, ο παίκτης πάλι ξαναπαίζει, κ.ο.κ.

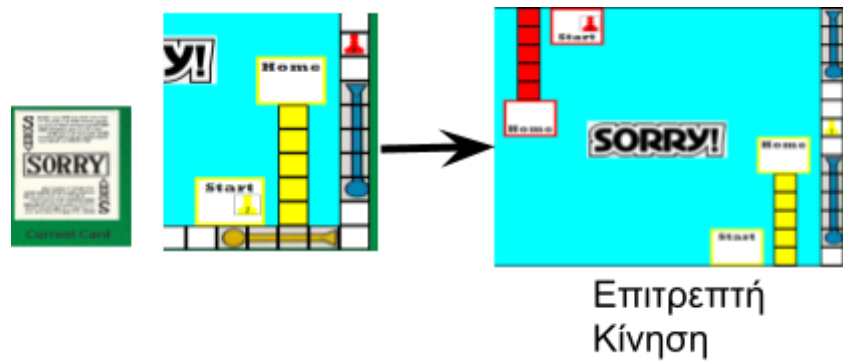
### ***Πως ξεκινάει ένα πιόνι από τη θέση Εκκίνησης Start***

Ένα πιόνι ενός παίκτη μπορεί να ξεκινήσει είτε με κάρτα με αριθμό 1, είτε με κάρτα με αριθμό 2, είτε με κάρτα Sorry. Αν ξεκινήσει με αριθμό 1 ή 2, τότε τοποθετεί το πιόνι του στο τετράγωνο που έχει μέσα τον κύκλο, ακριβώς κάτω από την θέση Start, όπως φαίνεται και στα παραδείγματα στην Εικόνα 2.



**Εικόνα 2. Ξεκίνημα από τη Θέση Start με κάρτα 1 ή 2**

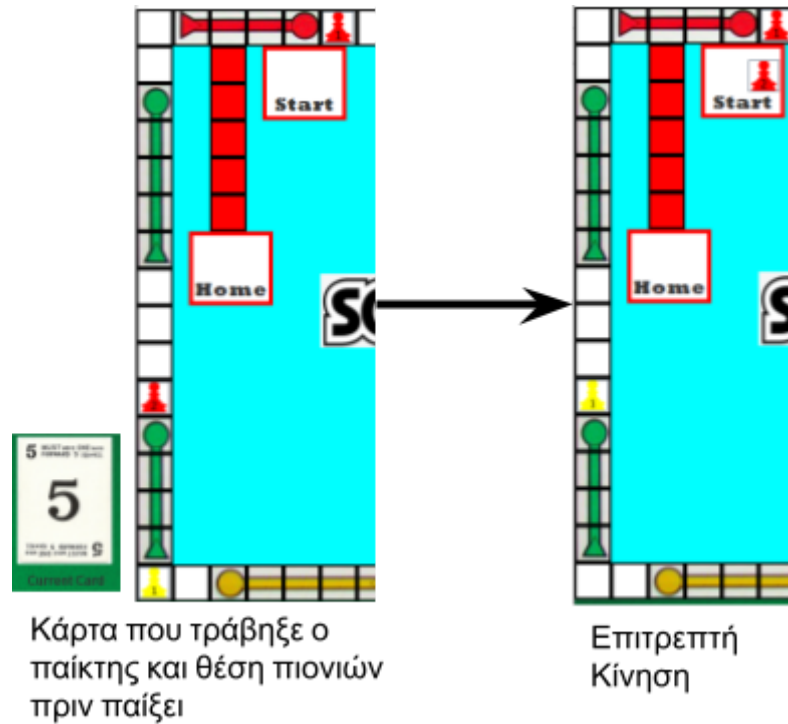
Αν έρθει η κάρτα Sorry (για παράδειγμα για τον παίκτη με τα κίτρινα πιόνια στην Εικόνα 3), τότε μπορεί να γίνει η ανταλλαγή του πιονιού του παίκτη με ένα του αντιπάλου, όπως φαίνεται στην Εικόνα 3. Αν όμως τα πιόνια του κόκκινου παίκτη βρίσκονταν στις προστατευόμενες θέσεις τους ( SafetyZone, Home) τότε δεν θα ήταν εφικτή αυτή η κίνηση.



Εικόνα 3. Ξεκίνημα από τη θέση Start με κάρτα Sorry

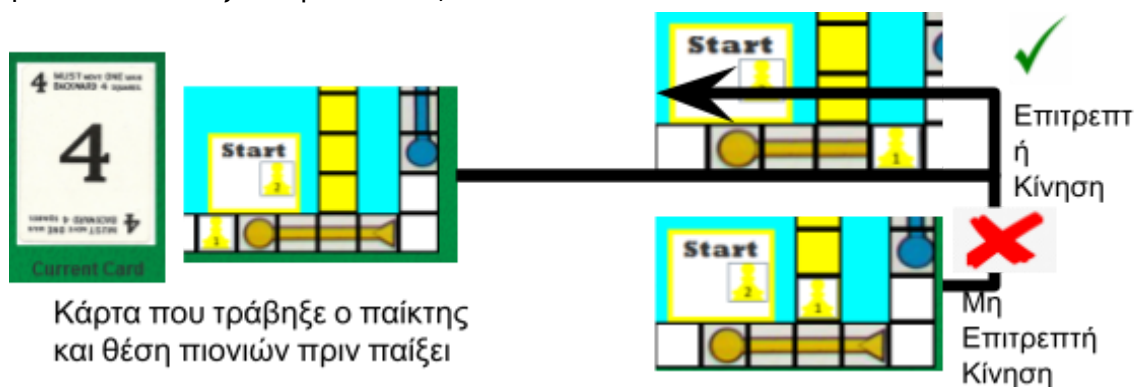
## Κανόνες κίνησης μέσα στο ταμπλό

1. Τα πιόνια ενός παίκτη δε γίνεται σε καμία περίπτωση να εισέλθουν στη θέση εκκίνησης, στην περιοχή ασφαλείας και στη θέση Σπίτι **του αντιπάλου**.
2. Ένα πιόνι μπορεί να περάσει πάνω από ένα άλλο, αλλά δε γίνεται να βρίσκονται ποτέ στο ίδιο τετράγωνο 2 πιόνια (εξαιρούνται βέβαια οι θέσεις εκκίνησης και τερματισμού για τα πιόνια του κάθε παίκτη). Συγκεκριμένα ισχύουν οι κανόνες:
  - a. Δύο πιόνια ίδιου χρώματος δεν μπορούν να βρίσκονται ποτέ στο ίδιο τετράγωνο. Αν η μοναδική εφικτή κίνηση έχει αυτό το αποτέλεσμα, τότε αλλάζει η σειρά χωρίς να γίνει κάποια κίνηση (ο παίκτης πατάει το κουμπί fold).
  - b. Δύο πιόνια διαφορετικού χρώματος δεν μπορούν να βρίσκονται ποτέ στο ίδιο τετράγωνο. Όμως αν ένας παίκτης φτάσει σε ένα τετράγωνο που ο αντίπαλος του έχει πιόνι, τότε παίρνει τη θέση του, και το πιόνι του αντιπάλου γυρνάει στην αρχική του θέση (θέση Start), όπως φαίνεται στο παράδειγμα στην Εικόνα 4.



Εικόνα 4. Παράδειγμα με πιόνια στην ίδια θέση

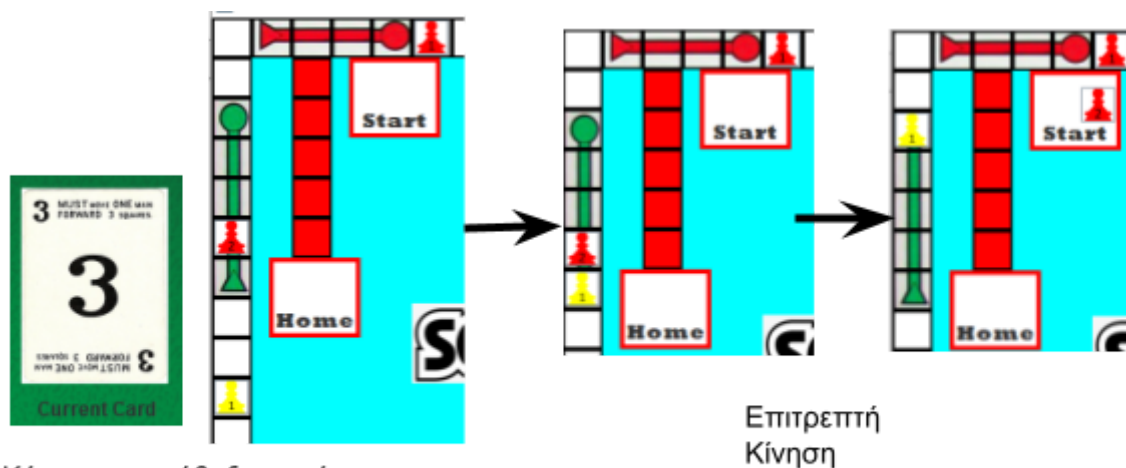
4. Όμως αν τουλάχιστον ένα πιόνι μπορεί να μετακινηθεί, τότε πρέπει να μετακινηθεί ακόμα και αν αυτό αποτελεί μειονέκτημα για τον παίκτη που παίζει (**εκτός από τον κανόνα για την κάρτα 11, που ο παίκτης έχει την επιλογή να μην ανταλλάξει το πιόνι του αν δε θέλει**).
5. Δε γίνεται να εισέλθει ένα πιόνι στην περιοχή ασφαλείας του πηγαίνοντας προς τα πίσω (πχ. χρησιμοποιώντας την κάρτα 4). Όμως μπορεί να φτάσει κοντά στην περιοχή ασφαλείας πηγαίνοντας προς τα πίσω, έτσι ώστε σε επόμενο γύρο να βρίσκεται πολύ κοντά στην περιοχή αυτή. Στο παράδειγμα στην Εικόνα 5, όπου ήρθε η κάρτα με αριθμό 4, ο παίκτης μπορεί να πάει 4 τετράγωνα πίσω και να βρεθεί κοντά στην περιοχή ασφαλείας του (δείτε πάνω δεξιά στην Εικόνα 5), αλλά δε μπορεί να βρεθεί μέσα στην περιοχή ασφαλείας του (οπότε δεν είναι επιτρεπτή μία τέτοια κίνηση, όπως φαίνεται κάτω δεξιά στην Εικόνα 5).



Εικόνα 5. Παράδειγμα με κίνηση προς τα πίσω

## Επιπλέον Παραδείγματα

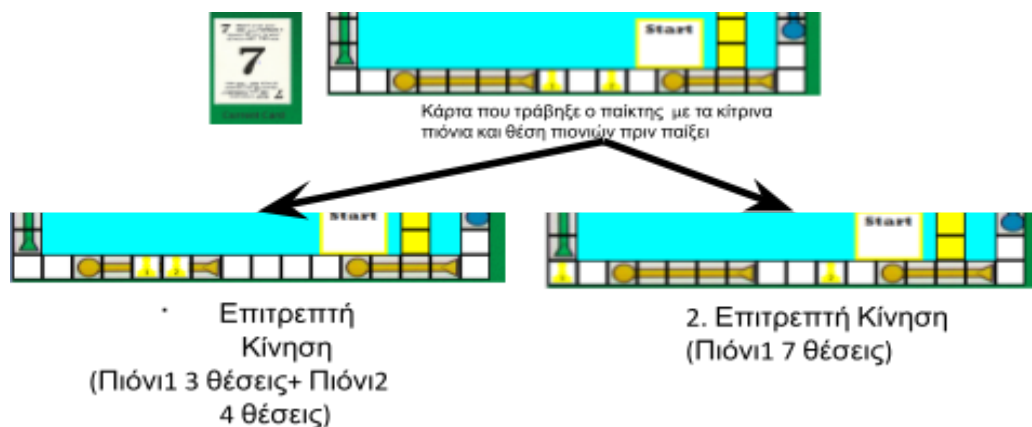
1. **Παράδειγμα για θέσεις Slide.** Στην Εικόνα 6 βλέπουμε ένα παράδειγμα για τη θέση τσουλήθρας. Συγκεκριμένα στην σειρά του παίκτη με τα κίτρινα πιόνια, ήρθε η κάρτα με αριθμό 3. Με αυτήν την κάρτα ο παίκτης πήγε στην αρχή μίας πράσινης τσουλήθρας. Από τη στιγμή που η τσουλήθρα έχει διαφορετικό χρώμα από του παίκτη, προχωράει μέχρι το τέλος της τσουλήθρας και διώχνει όποιο πιόνι βρίσκεται μέσα σε αυτή, συγκεκριμένα το στέλνει στην αρχική του θέση, όπως φαίνεται και στην Εικόνα 6. Στο συγκεκριμένο παράδειγμα αν η τσουλήθρα ήταν κίτρινου χρώματος, τότε το κίτρινο πιόνι θα σταματούσε απλώς στην αρχή της τσουλήθρας.



Κάρτα που τράβηξε ο παίκτης με τα κίτρινα πιόνια και θέση πιονιών πριν παίξει

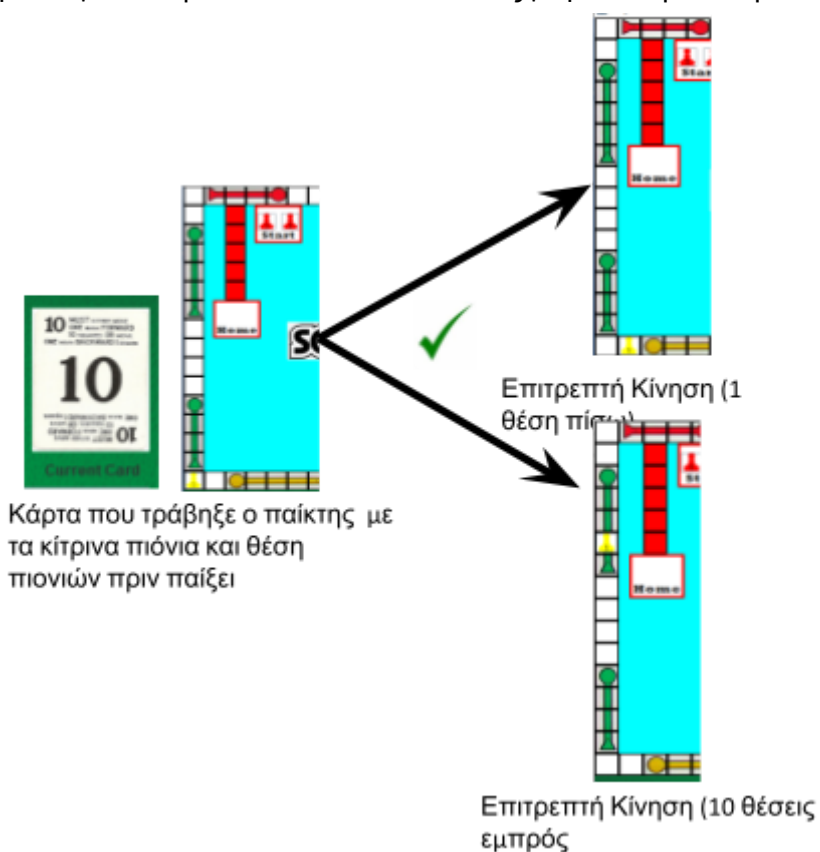
Εικόνα 6. Παράδειγμα Θέσης Slide

2. **Παράδειγμα για κάρτα 7.** Στην Εικόνα 7 βλέπουμε ένα παράδειγμα για την κάρτα 7. Συγκεκριμένα στην σειρά του παίκτη με τα κίτρινα πιόνια, ήρθε η κάρτα με αριθμό 7. Ο παίκτης έχει πολλές εναλλακτικές επιλογές στο παράδειγμα αυτό, μπορεί να κινήσει και τα 2 του πιόνια, πχ 3 θέσεις το πιόνι 1 και 4 θέσεις το πιόνι 2 (όπως φαίνεται κάτω αριστερά), 7 θέσεις ένα συγκεκριμένο πιόνι (όπως φαίνεται κάτω δεξιά), κλπ.



Εικόνα 7. Παραδείγματα κάρτας με αριθμό 7

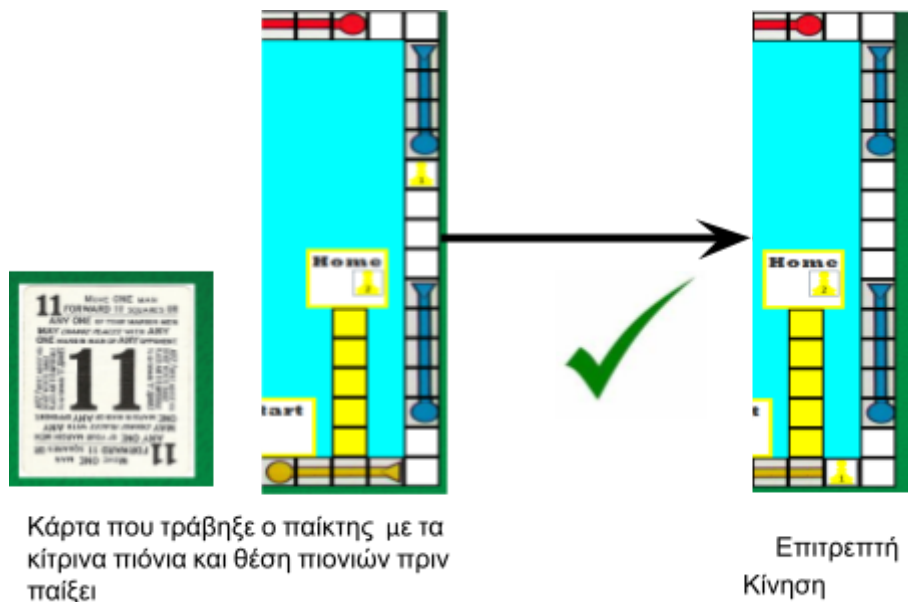
3. **Παράδειγμα για κάρτα 10.** Στην Εικόνα 8 βλέπουμε ένα παράδειγμα για την κάρτα 10. Συγκεκριμένα στην σειρά του παίκτη με τα κίτρινα πιόνια, ήρθε η κάρτα με αριθμό 10. Ο παίκτης μπορεί να μετακινήσει ένα πιόνι του 10 θέσεις μπροστά ή 1 θέση πίσω.



Εικόνα 8. Παράδειγμα Κάρτας με αριθμό 10

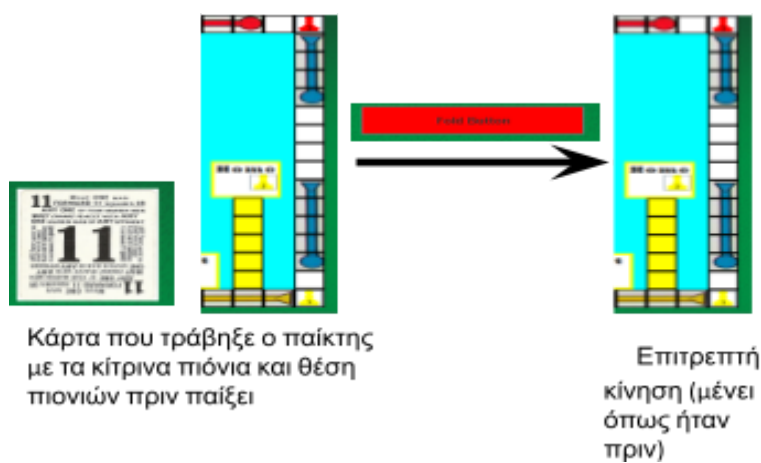
4. **Παράδειγμα για κάρτα 11.** Στην Εικόνα 9 βλέπουμε ένα παράδειγμα για την κάρτα 11. Συγκεκριμένα στην σειρά του παίκτη με τα κίτρινα πιόνια, ήρθε η κάρτα με αριθμό 11. Στο πρώτο παράδειγμα, ο παίκτης επέλεξε να μετακινήσει ένα του πιόνι 11 θέσεις.





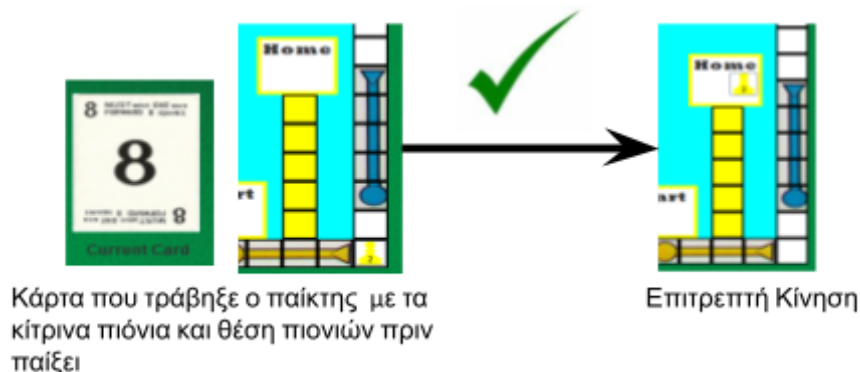
Εικόνα 9. Παράδειγμα Κάρτας 11 (Κίνηση Εμπρός)

Μπορεί επίσης να ανταλλάξει το πιόνι του με ένα του αντιπάλου, αλλά δεν είναι υποχρεωμένος να το κάνει. Στο παράδειγμα στην Εικόνα 10, ο παίκτης με τα κίτρινα πιόνια δε μπορεί να προχωρήσει 11 βήματα. Όμως δεν συμφέρει τον παίκτη να ανταλλάξει το πιόνι του (πιόνι 1) με το κόκκινο πιόνι 1, γιατί η θέση που βρίσκεται το πιόνι του είναι καλύτερη από του αντιπάλου. Οπότε μπορεί απλώς να επιλέξει fold και να αλλάξει ο γύρος.



Εικόνα 10. Παράδειγμα Κάρτας 11 (Ανταλλαγή)

5. **Παραδείγματα για θέση Home.** Στην Εικόνα 11 βλέπουμε ένα παράδειγμα για την θέση Home. Συγκεκριμένα, στην σειρά του παίκτη με τα κίτρινα πιόνια, ήρθε η κάρτα με αριθμό 8. Μιας και το πιόνι του απέχει 8 θέσεις από την θέση Home, τότε μπορεί να μετακινηθεί εκεί το πιόνι του.



Εικόνα 11. Επιτρεπτή κίνηση προς τη θέση Home

Στο δεύτερο παράδειγμα στην Εικόνα 12, το πιόνι του παίκτη είναι πάλι 8 θέσεις μακριά από τη θέση Home, αλλά ήρθε κάρτα με αριθμό 12. Σε αυτήν την περίπτωση δε μπορεί να μετακινηθεί στη θέση Home.



Εικόνα 12. Μη επιτρεπτή κίνηση προς τη θέση Home

## Εξτρά Προσοχή

- Στις κάρτες 3 & 5, ο παίκτης πρέπει να μετακινήσει όλα τα πιόνια που μπορεί
- Στις κάρτες 8 & 12 ο παίκτης μπορεί να επιλέξει να τραβήξει άλλη κάρτα

## Τέλος Παιχνιδιού - Νικητής

Το παιχνίδι τελειώνει όταν ο πρώτος παίκτης φέρει τα 2 του πιόνια στην περιοχή Home (Σπίτι). Ο παίκτης αυτός ανακηρύσσεται αυτόματα νικητής.

### Bonus 1 (5%)

Δυνατότητα αποθήκευσης του παιχνιδιού και μετέπειτα συνέχισης του σε επόμενο στιγμιότυπο εκτέλεσης της εφαρμογής. Συγκεκριμένα, θα πρέπει να δίδεται στο χρήστη μέσω του μενού μια επιλογή αποθήκευσης και εξόδου από την εφαρμογή. Αυτή η επιλογή θα έχει ως αποτέλεσμα τη μόνιμη αποθήκευση (persistent storage) της κατάστασης (state) του παιχνιδιού εκείνη τη δεδομένη χρονική στιγμή. Αφήνεται στη δική σας ευχέρεια, ως σχεδιαστές της εφαρμογής, να ορίσετε τι θα πρέπει να αποθηκεύσετε και πως. Έπειτα, δεδομένου του ότι έχει προηγηθεί μια τέτοια ενέργεια αποθήκευσης, θα πρέπει η εφαρμογή σας να είναι σε θέση πάλι μέσω του μενού εκτέλεσης να προσφέρει τη δυνατότητα συνέχισης του συγκεκριμένου παιχνιδιού.

### Bonus 2 (5%) Παιχνίδι έως 4 παίκτες

Προσθέστε λειτουργία έτσι ώστε να μπορεί ο χρήστης να επιλέξει στην αρχή τον αριθμό των παικτών που επιθυμεί να παίξουν (2 , 3 ή 4 ). Ανάλογα την επιλογή του, στο ταμπλό θα προστίθενται τα πιόνια και οι περιοχές start και home για κάθε παίκτη. Τα νέα πιόνια θα έχουν χρώμα πράσινο και μπλέ έτσι ώστε να υπάρχει slide ίδιο χρώμα με αυτά.

### Υποδείξεις για Σχεδίαση/Υλοποίηση

Κατά τη διάρκεια της σχεδίασης του παιχνιδιού (και εν γένει οποιουδήποτε λογισμικού ή τεχνικού έργου), επιδιώκεται η αποσύνθεση του συστήματος σε μικρότερα τμήματα, με στόχο την ανάθεση σαφώς ορισμένων και καθορισμένων αρμοδιοτήτων σε κάθε τμήμα και την επικύρωση ότι όλα τα τμήματα μαζί επιτυγχάνουν τους σκοπούς του συστήματος. Επομένως, η σχεδίαση είναι μια διαδικασία επίλυσης και κατακερματισμού του αρχικού προβλήματος σε επιμέρους μικρότερα και ευκολότερα επιλύσιμα υπο-προβλήματα που θα ικανοποιούν τις λειτουργικές απαιτήσεις και θα υπόκεινται σε συγκεκριμένες αρχές καλής σχεδίασης.

Μια τέτοια αρχή αποτελεί η αποσύνδεση του μοντέλου (model), που περιγράφει τα δεδομένα, τη συμπεριφορά τους και το σύνολο των κανόνων που τα διέπει, από την απεικόνισή τους (view). Βασικός στόχος μιας τέτοιας αποσυνδεδεμένης προσέγγισης είναι η ελαχιστοποίηση των απαιτούμενων επεμβάσεων σε κώδικα που μπορούν να έχουν μελλοντικές αλλαγές είτε στο μοντέλο είτε στο τρόπο/μέσο απεικόνισης. Αυτό οδηγεί σε καλύτερη ποιότητα κώδικα, με μικρότερο κόστος συντήρησης, επέκτασης και επαναχρησιμοποίησης. Για να γίνει πιο κατανοητό αυτό θεωρήστε το ακόλουθο σενάριο. Υποθέστε ότι αρχικά έχετε σχεδιάσει μια εφαρμογή που για διάφορους λόγους εκτυπώνει τα αποτελέσματα της στην κονσόλα. Έχοντας ακολουθήσει μια τέτοια αρχιτεκτονική ‘χαλαρή’ σύνδεσης η μετάβαση σε ένα γραφικό παραθυρικό περιβάλλον γίνεται ομαλά, απλώς τροποποιώντας κατάλληλα (επεκτείνοντας) τις διαδικασίες εκείνες που ήταν υπεύθυνες για την αποτύπωση του μοντέλου στην κονσόλα.

Στη συνέχεια ακολουθεί μια προτεινόμενη σχεδίαση της εφαρμογής, την οποία μπορείτε να ακολουθήσετε και να επεκτείνετε κατάλληλα. Η συγκεκριμένη στρατηγική σχεδίασης δεν είναι μοναδική, καθώς ένα αντικειμενοστραφές σύστημα λογισμικού μπορεί προφανώς να δομηθεί με πάρα πολλούς τρόπους. Αρκεί να αναλογιστεί κανείς τον αριθμό των δυνατών κλάσεων, των λειτουργιών που μπορούν να περιλαμβάνουν και των δυνατών συσχετίσεων μεταξύ τους. Συνεπώς, είστε ελεύθεροι να προτείνετε τη δική σας σχεδίαση εφόσον ακολουθεί τις αρχές που περιγράψαμε και είναι κατάλληλα τεκμηριωμένη.

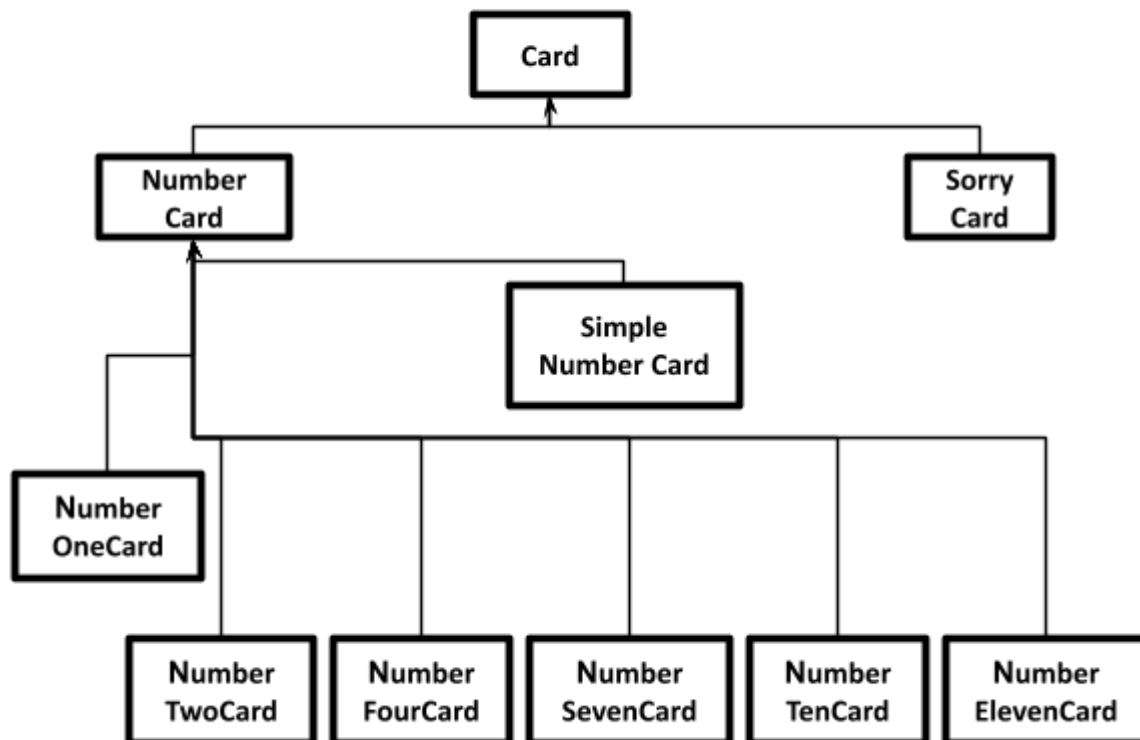
### **MVC (Model-View-Controller) pattern**

Βασική αρχή στην ανάπτυξη της παρούσας εργασίας θα πρέπει να είναι το MVC (Model-View-Controller) pattern. Σύμφωνα με αυτό το πρότυπο θα πρέπει να διαχωρίζεται η ανάπτυξη της γραφικής διεπαφής του παιχνιδιού (View) από τον πυρήνα του παιχνιδιού που περιέχει όλη την πληροφορία κατάστασης (Model) και από τον μηχανισμό διαχείρισης και ενημέρωσης των ενεργειών του παιχνιδιού με τη γραφική του απεικόνιση (Controller). Δείτε ένα video για να το καταλάβετε πλήρως: <http://www.newthinktank.com/2013/02/mvc-java-tutorial/>.

Πιο συγκεκριμένα το Model μπορεί να θεωρηθεί ότι αποτελείται από οτιδήποτε σχετίζεται με τα δεδομένα του παιχνιδιού. Υπό αυτήν την έννοια οι «Κάρτες», το «ταμπλό», οι «Παίχτες» αποτελούν μέρος του μοντέλου του παιχνιδιού καθώς περιγράφουν τα εκάστοτε δεδομένα που καλείται να διαχειριστεί ο Controller του παιχνιδιού. Ο Controller επιφορτίζεται με τη διαχείριση της αλληλεπίδρασης της γραφικής διεπαφής με το μοντέλο. Το MVC pattern σας δίνει την δυνατότητα να διαχωρίσετε την υλοποίηση των παραπάνω και κατά συνέπεια καταστούν το πρόγραμμά σας ευκολότερο και στην ανάπτυξη και στην αποσφαλμάτωση (το κάθε συστατικό μπορεί να αναπτυχθεί και να δοκιμαστεί ξεχωριστά).

### **Προτεινόμενες Κλάσεις – Model**

Το model θα πρέπει να περιλαμβάνει όλα τα περιεχόμενα του παιχνιδιού, δηλαδή θα πρέπει να υπάρχουν κλάσεις για τις **κάρτες**, τις θέσεις του **ταμπλό** του παιχνιδιού, τον **παίκτη**, τα **πιόνια** του παίκτη, το **ταμπλό** κλπ.

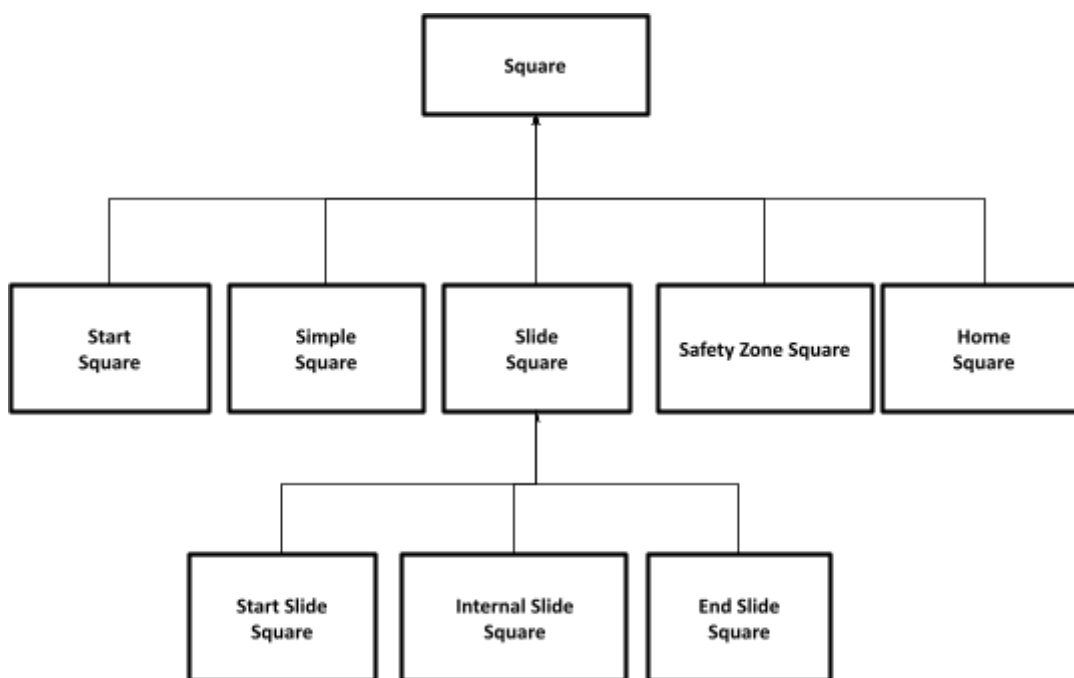


Εικόνα 13. Οι πιθανές υποκλάσεις της κλάσης Card

Πιο συγκεκριμένα, μία βασική κλάση είναι η abstract κλάση **Card**. Η κλάση αυτή μοντελοποιεί μία κάρτα του παιχνιδιού και ορίζει τα γνωρίσματα της και τις χρήσιμες μεθόδους της, για παράδειγμα για όλες τις κάρτες είναι σημαντικό να ξέρουμε αν έχουν παιχτεί ή όχι, ενώ κάθε κάρτα μπορεί να έχει και μία περιγραφή. Η κλάση αυτή μπορεί να έχει μία μέθοδο **movePawn** που να λαμβάνει ως είσοδο ένα πιόνι και το ταμπλό του παιχνιδιού, και θα ελέγχει αν το πιόνι μπορεί να παιχτεί με αυτήν την κάρτα και θα κάνει την κίνηση. Η μέθοδος αυτή θα έχει διαφορετική υλοποίηση ανάλογα με την κάθε κάρτα, γι αυτό είναι καλό να χωριστεί η κλάση **Card** σε υποκλάσεις. Συγκεκριμένα, ανάλογα με τις διαφορές που έχουν οι κάρτες σύμφωνα με τον τύπο τους, μπορούν να χωρίζονται σε υποκλάσεις, όπως φαίνεται στην Εικόνα 13, δηλαδή την **NumberCard** και **SorryCard**. Η **NumberCard** μπορεί να έχει ένα αριθμό και μια μεταβλητή για τη φορά που παίζονται οι κάρτες. Η **SorryCard** δεν έχει κάποιο αριθμό, απλώς στην υλοποίηση της πρέπει να δέχεται ως είσοδο και το πιόνι του αντιπάλου, για να τσεκάρει αν μπορεί να γίνει η ανταλλαγή. Επίσης μπορεί να έχει ως υποκλάσεις της, τις κλάσεις **SimpleNumberCard** (αφορά τις κάρτες 3,5,8,12), την **NumberFourCard**, την **NumberOneCard** και **NumberTwoCard**, οι οποίες έχουν διαφορετικές ιδιότητες (π.χ., με την κάρτα 2 μπορεί επίσης ο παίκτης να παίξει ξανά). Ακόμα μπορεί να έχει ως υποκλάση την **NumberSevenCard** η οποία μπορεί να έχει μία μέθοδο που λαμβάνει ως είσοδο 2 πιόνια και 2 αριθμούς με άθροισμα το 7, και τσεκάρει αν μπορούν να παιχτούν, την **NumberTenCard** με την οποία μπορεί κάποιος να κάνει κίνηση είτε εμπρός είτε προς τα πίσω, και την κλάση **NumberElevenCard**, με την οποία κάποιος μπορεί να ανταλλάξει ένα πιόνι του με του αντιπάλου (άρα πρέπει να δέχεται και ως είσοδο ένα πιόνι αντιπάλου)..

Μία βασική κλάση είναι η **Square**, που μοντελοποιεί το κάθε τετράγωνο του παιχνιδιού, το οποίο έχει μία συγκεκριμένη θέση στο ταμπλό και ένα συγκεκριμένο χρώμα (θεωρείστε ότι το απλό τετράγωνο έχει χρώμα άσπρο). Επίσης για κάθε τετράγωνο θα πρέπει να καταγράφουμε αν υπάρχει κάποιο πιόνι πάνω του, ή είναι προς το παρόν άδειο.

Επίσης η **Square** μπορεί να χωρίζεται σε υποκλάσεις, όπως οι **StartSquare**, **HomeSquare**, **SafetyZoneSquare**, **SimpleSquare** και η **SlideSquare**. Οι τρεις πρώτες αναπαριστούν τετράγωνα που ανήκουν σε ένα συγκεκριμένο παίκτη και ο αντίπαλος δεν μπορεί να έχει πρόσβαση σε αυτά, ενώ η **SimpleSquare** αναπαριστά ένα από τα απλά τετράγωνα του ταμπλό. Η **SlideSquare** αναπαριστά ένα τετράγωνο μίας τσουλήθρας και μπορεί να χωριστεί σε υποκλάσεις, όπως η **StartSlideSquare**, η **InternalSlideSquare** και η **EndSlideSquare**. Η **StartSlideSquare** αντιστοιχεί στο πρώτο τετράγωνο μίας τσουλήθρας, η **InternalSlideSquare** σε ένα εσωτερικό και η **EndSlideSquare** στο τελευταίο τετράγωνο μίας τσουλήθρας.



Εικόνα 14. Οι πιθανές υποκλάσεις της κλάσης **Square**

Απαραίτητες κλάσεις του παιχνιδιού είναι αυτές του παίκτη (**Player**) και των πιονιών (**Pawn**). Η κλάση **Player** προσομοιώνει τον παίκτη του παιχνιδιού. Κάθε παίκτης έχει ένα συγκεκριμένο χρώμα, ένα όνομα, διατηρεί 2 πιόνια, ενώ μπορεί να έχει και μεταβλητές και μεθόδους που να δείχνουν αν είναι η σειρά του να παίξει. Η κλάση **Pawn** προσομοιώνει ένα πιόνι, που έχει ένα συγκεκριμένο χρώμα, βρίσκεται σε συγκεκριμένο τετράγωνο και είναι ενεργό ή έχει φτάσει στον τελικό του προορισμό.

Επίσης είναι σημαντική μία κλάση **Deck**, όπου θα είναι υπεύθυνη για την αρχικοποίηση των καρτών, ενώ θα αρχικοποιεί και θα περιέχει το κεντρικό ταμπλό με τα τετράγωνα και τα πιόνια κάθε παίκτη. Μπορεί να περιέχει μεθόδους για να παίρνει ο παίκτης καινούριες κάρτες να ελέγχει αν οι κάρτες έχουν τελειώσει (για να τις ανακατέψει ξανά), να μετακινεί ένα πιόνι από μία θέση σε μία άλλη, να τσεκάρει αν ο παίκτης έχει τη δυνατότητα να κάνει fold (δεν υπάρχει κάποια εφικτή κίνηση), κλπ.

## Controller

Σκεφτείτε αναλυτικά τη ροή του παιχνιδιού και συμπεριλάβετε στη σχεδίαση σας όλες τις απαραίτητες λειτουργίες της κλάσης. Συγκεκριμένα, είναι σημαντικό να σκεφτείτε και να σχεδιάσετε το πώς θα επικοινωνούν τα διαφορετικά κομμάτια του model και του controller μεταξύ τους, πχ ποια θα είναι η ροή όταν ο παίκτης ρίξει μία κάρτα, ποιες μέθοδοι (που θα βρίσκονται σε διαφορετικές κλάσεις) και με ποια σειρά θα πρέπει να χρησιμοποιηθούν.



Η γραφική διεπαφή του παιχνιδιού πρέπει να παρουσιάζει το ταμπλό και να δίνει τη δυνατότητα στους παίκτες να παίξουν ταυτόχρονα το παιχνίδι σε έναν υπολογιστή. Ένα σχεδιαστικό παράδειγμα υλοποίησης της εν λόγω διεπαφής παρουσιάζεται στην Εικόνα 15. Σε αυτήν την εικόνα έχουμε αριστερά το ταμπλό, δεξιά το χώρο που περιέχει την στοίβα των καρτών και την κάρτα που τράβηξε ο παίκτης στο γύρο, ένα πλαίσιο πληροφοριών, και ένα fold

button. Η γραφική απεικόνιση αυτή είναι ενδεικτική. **Οποιαδήποτε διαφοροποίηση που δεν μειώνει την λειτουργικότητα του παιχνιδιού είναι αποδεκτή.** Για παράδειγμα κάποιος θα μπορούσε να προσθέσει οτιδήποτε του αρέσει που να κάνει το παιχνίδι πιο ευχάριστο όπως τα ονόματα των παιχτών, ήχους (πχ όταν ένα πιόνι κινείται), κλπ.

### Προτεινόμενες Κλάσεις – View

Το View είναι η γραφική διεπαφή με την οποία αλληλεπιδρούν οι τελικοί χρήστες του παιχνιδιού. Όσον αφορά την διεπαφή η χρήση κληρονομικότητας μπορεί να διευκολύνει την ανάπτυξη μέσω της δημιουργίας νέων τύπων UI συστατικών τα οποία επεκτείνουν τη λειτουργικότητα των υπαρχόντων. Μπορείτε να χρησιμοποιήσετε την κλάση JLayeredPane, όπου κάποιος μπορεί να τοποθετήσει ένα component (πχ, κουμπί, ετικέτα, κείμενο) πάνω σε ένα άλλο και κάθε component έχει συγκεκριμένες συντεταγμένες. Για παράδειγμα, στην Εικόνα 15, βλέπουμε ότι έχουμε ένα βασικό background (JLayeredPane), και πάνω σε αυτό έχουμε φτιάξει το ταμπλό, που αποτελείται από κάποια τετραγωνάκια που φτιάχτηκαν μέσω στιγμιότυπων JLabel. Επίσης έχουμε φτιάξει κάποια κουμπιά για τις κάρτες και τα πιόνια του κάθε παίκτη. Έτσι, όταν ο παίκτης πατάει πάνω στη στοίβα με τις κάρτες, μπορεί να παίρνει μία νέα κάρτα, και έπειτα μπορεί να πατάει πάνω σε ένα πιόνι, έτσι ώστε να κινείται (αν είναι εφικτό) αυτό το πιόνι με βάση τους κανόνες της κάρτας.

Εκτός από αυτά, έχουμε και μία περιοχή για τις πληροφορίες του παιχνιδιού που έχει φτιαχτεί μέσω της κλάσης JTextArea.

Επίσης, για κάποιες κάρτες θα χρειαστεί ο παίκτης να επιλέξει ανάμεσα σε διαφορετικά σενάρια, πχ για την κάρτα 10 αν θα πάει μπροστά η πίσω, οπότε την επιλογή αυτή μπορεί να την κάνει ο παίκτης μέσω Java Dialog, όπου μπορείτε να χρησιμοποιήσετε οποιοδήποτε τύπο Java Dialogs σας βολεύει (δείτε <http://docs.oracle.com/javase/tutorial/uiswing/components/dialog.html>).

Εναλλακτικά, μπορεί κάποιος να χρησιμοποιήσει και ένα BorderLayout για την σχεδίαση, για να διαχωρίσει την περιοχή των καρτών και του ταμπλό, αλλά και GridLayout για να φτιάξει τα τετραγωνάκια του ταμπλό (διαστάσεις 16\*16).

**Δεν υπάρχει περιορισμός στην υλοποίηση των γραφικών, ουσιαστικά προσπαθήστε να χρησιμοποιήσετε την υλοποίηση που σας φαίνεται προτιμότερη.**

### Βοηθητικά για να Ξεκινήσετε (SOS)

Επίσημες Οδηγίες και Βίντεο στα Αγγλικά. **Προσοχή, οι οδηγίες μπορεί να είναι λίγο διαφορετικές από την εκφώνηση. Εσείς πάντα στηρίζετε στις οδηγίες της εκφώνησης!!!**

**Οδηγίες:** <https://www.hasbro.com/common/instruct/Sorry.PDF>

**Video:** <https://www.youtube.com/watch?v=uMdylyrST6w> (έχει αγγλικούς υπότιτλους)

Παρακάτω αναγράφονται κάποια πράγματα που μπορούν να σας βοηθήσουν στην υλοποίηση:



- Οι φωτογραφίες για όλα τα περιεχόμενα του παιχνιδιού και template για την αναφορά της εργασίας σας έχουν ανέβει στο elearn!
- Ενδεικτικό βοήθημα για το MVC (Model View Controller) μοντέλο σε Java <http://www.newthinktank.com/2013/02/mvc-java-tutorial> .
- Επίσης **θα σας δοθεί ένα project προηγούμενης χρονιάς** ώστε να αποκτήσετε από νωρίς μια αίσθηση του πως περίπου θα είναι η τελική σας υλοποίηση.
- **(SOS) Τέλος μετά το φροντιστήριο των γραφικών, ανεβαίνουν παραδείγματα με γραφικά που σας βοηθούν να ξεκινήσετε! Μπορείτε να ξεκινήσετε με αυτά τα παραδείγματα και να τα αλλάξετε έτσι ώστε να φτιάξετε το γραφικό περιβάλλον της εργασίας σας.**

***Καλή Εργασία!!!***