ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ ΠΛΥ212 Τεχνικές Αντικειμενοστρεφούς Προγραμματισμού
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Δεύτερη Σειρά ασκήσεων
Ημερομηνία Παράδοσης: Παρασκευή 26 Απριλίου 11:59 μ.μ.

Οι παρακάτω ασκήσεις πρέπει να παραδοθούν μέχρι τις 26 Απριλίου τα μεσάνυχτα. Στην υλοποίηση των κλάσεων σας δεν θα πρέπει να έχετε public μεταβλητές. Βαθμοί θα αφαιρεθούν για προγράμματα που δεν είναι καλά γραμμένα, δηλαδή δεν είναι σωστά στοιχισμένα ώστε να διαβάζονται εύκολα.

Άσκηση 1

Στην τάξη υλοποιήσαμε μια στοίβα δυναμικού μεγέθους χρησιμοποιώντας συνδεδεμένα στοιχεία. Σε αυτή την άσκηση θα υλοποιήσετε μία **ουρά**. Η ουρά είναι μια δομή που υλοποιεί την FIFO (First-In-First-Out) πολιτική, όπου τα στοιχεία βγαίνουν με τη σειρά με την οποία μπαίνουν. Η ουρά θα είναι δυναμικού μεγέθους και θα αποτελείται από συνδεδεμένα στοιχεία. Στην υλοποίηση σας, παρόμοια με το παράδειγμα που κάναμε στην τάξη, η ουρά θα αποθηκεύει αντικείμενα τύπου **Person.** Η κλάση **Person** είναι αυτή που ορίσαμε στην τάξη, η οποία κρατάει για ένα άτομο το όνομα του και τον αριθμό ταυτότητας του.

Κατασκευάσετε μια κλάση **Queue** που υλοποιεί ουρά θα είναι δυναμικού μεγέθους και θα αποτελείται από συνδεδεμένα στοιχεία (**QueueElement**) και αποθηκεύει αντικείμενα Person. Η κλάση θα πρέπει να έχει υποχρεωτικά τις εξής μεθόδους:

- Μέθοδο **add** η οποία προσθέτει ένα καινούριο αντικείμενο στο τέλος της ουράς.
- Μέθοδο **remove** η οποία επιστρέφει το αντικείμενο στην κορυφή της ουράς, και το αφαιρεί. Αν η ουρά είναι άδεια επιστρέφει null.
- Μέθοδο **isEmpty** η οποία ελέγχει αν είναι άδεια η ουρά ή όχι.

Υλοποιήστε και μία κλάση **QueueTest** η οποία θα έχει τη main. Η QueueTest θα δημιουργεί ένα αντικείμενο Queue, και τέσσερα αντικείμενα Person. Στην συνέχεια θα προσθέτει ένα αντικείμενο, και μετά θα το αφαιρεί. Μετά θα προσθέτει άλλα τρία αντικείμενα και θα τα αφαιρεί. Στο τέλος θα προσπαθήσει να καλέσει την remove στην άδεια ουρά. Κάθε φορά που αφαιρεί ένα αντικείμενο από την ουρά τυπώνει και τα στοιχεία του Person.

Υπόδειξη: Για την υλοποίηση της στοίβας κρατούσαμε το πεδίο head το οποίο δείχνει στην κορυφή της στοίβας. Για την ουρά χρειαζόμαστε αναφορά και στην κορυφή (head) και στο τέλος (tail) της ουράς. Βάλετε ελέγχους για τις οριακές καταστάσεις

Άσκηση 2

Για την άσκηση αυτή θα προγραμματίσετε μια απλή εκδοχή του παιχνιδιού Αγωνία (στα αγγλικά Crazy Eights) όπου θα μπορείτε να παίζετε με τον υπολογιστή.

Κανόνες του παιχνιδιού

Το παιχνίδι παίζεται με μία τράπουλα. Η τράπουλα ανακατεύεται, μοιράζονται από 5 χαρτιά σε κάθε παίχτη και ανοίγεται ένα χαρτί στο τραπέζι. Οι παίχτες παίζουν ο ένας μετά τον άλλο, και ο κάθε παίχτης στη σειρά του, αν έχει ένα χαρτί που συμφωνεί είτε στο χρώμα (suit) είτε στο νούμερο (number) με το τελευταίο χαρτί που ρίχτηκε στο τραπέζι, το πετάει. Αν έχει ένα χαρτί με το νούμερο 8, μπορεί να το ρίξει ανά πάσα στιγμή, και να καθορίσει αυτός το χρώμα του (το οποίο μπορεί να είναι διαφορετικό από αυτό που έχει το χαρτί). Αν δεν έχει κάποιο χαρτί που να μπορεί να ρίξει τότε τραβάει χαρτιά από την τράπουλα μέχρι να έχει χαρτί το οποίο μπορεί να ρίξει., Αν ο αριθμός των εναπομεινάντων χαρτιών στην τράπουλα μειωθεί πολύ (π.χ., έχουμε λιγότερο από 10 χαρτιά), τότε τα χαρτιά στο τραπέζι ξαναμπαίνουν στην τράπουλα και ανακατεύονται εκ νέου. Όποιος παίχτης μείνει πρώτος χωρίς χαρτιά κερδίζει το παιχνίδι.

Υλοποίηση

Για την υλοποίηση σας θα ακολουθήσετε τις εξής γενικές οδηγίες.

1. Θα υλοποιήσετε μια κλάση **Card**, η οποία θα κρατάει τις πληροφορίες για το νούμερο και το χρώμα του χαρτιού. Θα δημιουργήσετε ένα αντικείμενο για το κάθε χαρτί της τράπουλας.
2. Θα υλοποιήσετε μια κλάση **Deck** η οποία θα υλοποιεί την τράπουλα. Η Deck θα κρατάει τα χαρτιά, και θα έχει μέθοδο που επιστρέφει ένα τυχαίο χαρτί και το αφαιρεί από την τράπουλα, και μέθοδο που προσθέτει τα χαρτιά από το τραπέζι στην τράπουλα. Μπορείτε να υλοποιήσετε την επιλογή του τυχαίου χαρτιού όπως σας βολεύει. Ένας τρόπος είναι να υπολογίσετε μια τυχαία αναδιάταξη των χαρτιών και κάθε φορά να επιστρέφετε το επόμενο χαρτί από αυτή την αναδιάταξη. Μπορεί να γίνει και με άλλους τρόπους.
3. Χρειάζεστε και μια κλάση για να κρατάει πληροφορία για το τι χαρτιά έχουν πέσει στο τραπέζι, και ποιο χαρτί είναι στην κορυφή. Προσοχή στην περίπτωση που το χαρτί που παίζεται είναι 8, μιας και το χρώμα του χαρτιού που δηλώνεται από τον παίκτη μπορεί να είναι διαφορετικό από αυτό έχει πραγματικά. Το πραγματικό χαρτί πρέπει να διατηρηθεί, μιας και μπορεί αργότερα να πρέπει να προστεθεί στην τράπουλα.
4. Χρειάζεστε μια κλάση **Player** που θα κρατάει τις πληροφορίες για ένα παίχτη. Θα έχει το χέρι του παίχτη και θα το τυπώνει πριν παίξει ο παίχτης. Η κλάση αυτή θα υλοποιεί και την τακτική του υπολογιστή. Ο υπολογιστής αρχικά ψάχνει αν έχει χαρτί (διαφορετικό από το 8) που να ταιριάζει με το χρώμα του χαρτιού στο τραπέζι. Αν έχει, επιστρέφει ένα από αυτά τα χαρτιά. Αν δεν έχει, ψάχνει να βρει αν έχει νούμερο που να ταιριάζει με το νούμερο στο τραπέζι. Αν έχει επιστρέφει αυτό το χαρτί. Αν δεν έχει, κοιτάει να δει αν έχει οχτάρι και αν έχει επιστρέφει αυτό το χαρτί. Αν το χαρτί που επιστρέφει είναι οχτάρι, καθορίζει το χρώμα να είναι το χρώμα στο οποίο έχει τα περισσότερα χαρτιά. Η μέθοδος θα πρέπει να αφαιρεί το χαρτί από τα χαρτιά του παίχτη και να το ρίχνει στο τραπέζι.
   Στην ίδια κλάση θα έχετε και μία μέθοδο που θα υλοποιεί το παιχνίδι του παίχτη-ανθρώπου. Θα του δείχνει το χέρι του και θα του ζητάει να επιλέξει το χαρτί που θα παίξει.
5. Στην κεντρική κλάση **CrazyEights** θα υλοποιήσετε το παιχνίδι. Η main θα δημιουργεί αντικείμενα των παραπάνω κλάσεων και θα καλεί τις μεθόδους για το παιχνίδι των παιχτών. Θα σταματάει όταν κάποιος παίχτης κερδίσει (τελειώσουν τα χαρτιά του) και θα ελέγχει αν η τράπουλα έχει λίγα χαρτιά οπότε θα βάζει τα χαρτιά στο τραπέζι πίσω στην τράπουλα.

Bonus 1: Υλοποιήσετε την εξής τακτική για τον υπολογιστή. Αρχικά για κάθε χρώμα υπολογίζει αν είναι υποψήφιο. Ένα χρώμα είναι υποψήφιο αν έχει μη μηδενικό αριθμό από χαρτιά και είτε συμφωνεί με το χρώμα του τραπεζιού, είτε έχει ένα χαρτί με νούμερο που συμφωνεί με το νούμερο του τραπεζιού. Από τα υποψήφια χρώματα ο υπολογιστής επιλέγει το χρώμα το οποίο έχει τα περισσότερα χαρτιά και παίζει το αντίστοιχο χαρτί.

Bonus 2: Στην Αγωνία υπάρχουν κι άλλα χαρτιά που έχουν ειδικό ρόλο. Ένα από αυτά είναι το 7. Αν ο ένας παίχτης ρίξει ένα 7, και ο άλλος δεν έχει 7 τότε αναγκάζεται να πάρει 2 χαρτιά. Αν έχει 7 και το ρίξει τότε ο πρώτος παίχτης παίρνει 4 χαρτιά. Γενικά αν έχουν πέσει x 7, και ο παίχτης δεν έχει 7 να ρίξει, παίρνει 2*x χαρτιά. Υλοποιήστε τον χειρισμό αυτής της περίπτωσης. Ο παίχτης-υπολογιστής θα παίζει 7 μόνο αν δεν έχει άλλο χαρτί να παίξει από το χρώμα, ή αν έχει πέσει κάτω το 7.

Υποδείξεις

- Για την υλοποίηση των παραπάνω κλάσεων θα χρειαστείτε κάποιες δυναμικές δομές, όπως η ArrayList. Μπορείτε να διαβάσετε για τις δυναμικές δομές στο documentation της Java από την Oracle.
- Να είστε προσεκτικοί αν καλέστε την remove για την ArrayList, ενώ διατρέχετε τα αντικείμενα της λίστας.
- Σε πολλές περιπτώσεις βολεύει να χρησιμοποιήσετε την τιμή null ως επιστρεφόμενη τιμή μιας μεθόδου σαν ένδειξη ότι η μέθοδος δεν είχε ικανοποιητικό αποτέλεσμα.
- Μπορεί να σας βολεύει να κρατάτε ξεχωριστά τα χαρτιά κάθε χρώματος.
- Σε κάθε κλάση μπορείτε (και θα χρειαστεί) να υλοποιήσετε και άλλες (public ή private) μεθόδους πέραν από αυτές που αναφέρονται στην εκφώνηση.

Άσκηση 3

Στην Άσκηση 2 η κλάση Player υλοποιεί το παιχνίδι και του υπολογιστή και του παίχτη-ανθρώπου. Υπάρχουν κοινές πληροφορίες που χρειάζονται και οι δύο, αλλά ο καθένας χρειάζεται διαφορετικές μεθόδους. Χρησιμοποιώντας την τεχνική της κληρονομικότητας, υλοποιήστε μια κλάση GeneralPlayer και δύο παράγωγες κλάσεις ComputerPlayer και HumanPlayer που υλοποιούν διαφορετικές μεθόδους για το παιχνίδι των αντίστοιχων παιχτών. Υλοποιήστε την κλάση GeneralCrazyEights που τις καλεί.

ΓΕΝΙΚΕΣ ΟΔΗΓΙΕΣ

Κάντε turnin τα προγράμματα σας στο assignment2@ply212.

π.χ. turnin assignment1@ply212 <τα αρχεία σας> .

Στον κώδικα να αναγράφονται σε σχόλια το όνομα το login και ο ΑΜ σας. Επίσης σε σχόλια γράψτε πριν από κάθε μέθοδο εν συντομία τι κάνει.