Τμήμα ΗΜΜΥ **2ο Εξάμηνο**
Α.Π.Θ. **2020**

ΕΡΓΑΣΙΑ ΣΤΟΝ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΗ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ

ΤΕΤΑΡΤΟ ΠΑΡΑΔΟΤΕΟ

Υλοποίηση εικονικών συναρτήσεων και γενικών κλάσεων σε C++

Το τέταρτο παραδοτέο της εργασίας απαιτεί τη δημιουργία των κληρονομημένων κλάσεων των **καρτών** του Tichu. Συγκεκριμένα, θα υλοποιηθούν οι παρακάτω κλάσεις **που βρίσκονται όλες στα αρχεία card.h και cardimplementations.cpp**:

1. Κλάση SimpleCard
   Η κλάση αυτή δημιουργεί τα αντικείμενα των απλών καρτών. **Κληρονομεί την αφηρημένη κλάση Card**. Θα πρέπει να έχει έναν constructor, καθώς και να **υλοποιεί τις γνήσιες εικονικές συναρτήσεις canBeInCombination και canBeInBomb**. Υπενθυμίζεται ότι οι απλές κάρτες μπορούν να συμμετέχουν σε όλους τους συνδυασμούς. Τέλος, η κλάση **υλοποιεί τη γνήσια εικονική συνάρτηση getPoints**. Υπενθυμίζεται ότι κάθε 5άρι μετράει ως 5 πόντοι, ενώ κάθε 10άρι ή Ρήγας μετράει ως 10 πόντοι. Όλες οι άλλες κάρτες μετράνε ως 0 πόντοι.

2. Κλάση SpecialCard
   Η κλάση αυτή δημιουργεί τα αντικείμενα των ειδικών καρτών. **Κληρονομεί την αφηρημένη κλάση Card**. Θα πρέπει να έχει έναν constructor. Οι κάρτες που δημιουργούνται είναι οι Mah Jong, Dragon, Phoenix και Dog. Κατά την αρχικοποίηση, θα πρέπει η αξία (value) του Mah Jong να είναι 1, η αξία του Dragon να είναι 15, η αξία του Phoenix να είναι -1 και η αξία του Dog να είναι -1. Αν η κάρτα δεν ανήκει σε κάποια από τις παραπάνω κατηγορίες θα πρέπει η αξία της να είναι -1. Επίσης, η κλάση SpecialCard θα πρέπει να **υλοποιεί τις γνήσιες εικονικές συναρτήσεις canBeInCombination και canBeInBomb**. Υπενθυμίζεται ότι τα Mah Jong και Phoenix μπορούν να συμμετέχουν σε συνδυασμούς, ενώ τα Dragon και Dog δε μπορούν. Επίσης, καμία ειδική κάρτα δε μπορεί να συμμετέχει σε βόμβες. Τέλος, η κλάση **υλοποιεί τη γνήσια εικονική συνάρτηση getPoints**. Υπενθυμίζεται ότι ο Δράκος μετράει ως 25 πόντοι, ενώ ο Φοίνικας μετράει ως -25 πόντοι (μπορεί να υπάρξει και αρνητικό σκορ). Όλες οι άλλες κάρτες μετράνε ως 0 πόντοι.

Η λογική των καρτών υλοποιείται με τράπουλες (κλάση Deck). Αυτά που θα πρέπει να υλοποιήσετε σε αυτό το κομμάτι της εργασίας είναι η λογική του ανακατέματος και η λογική με την οποία «κόβουμε φύλλα». **Οι συναρτήσεις σας όμως θα πρέπει να είναι γενικές, ώστε να μπορούν να χρησιμοποιηθούν για οποιοδήποτε τύπο αντικειμένου**. Συγκεκριμένα, θα πρέπει να υλοποιήσετε τις παρακάτω **πρότυπες (template) συναρτήσεις στο αρχείο shuffler.h**:

- Συνάρτηση shuffle
  Η συνάρτηση αυτή παίρνει ως είσοδο ένα δυναμικό πίνακα δυναμικών αντικειμένων (διπλό pointer) και ανακατεύει τα αντικείμενα στις θέσεις του πίνακα. Ο αλγόριθμος ανακατέματος που θα πρέπει να χρησιμοποιήσετε είναι ο παρακάτω:

```
ΑλγόριθμοςΑνακατέματοςΣτοιχείωνΕνόςΠίνακα(Πίνακας, Μέγεθος){
    Για το i (αριθμοδείκτη πίνακα) όπου i από το τελευταίο έως και το δεύτερο στοιχείο{
        j = τυχαίος ακέραιος αριθμός στο διάστημα [0, i]
        αντιμετάθεση του στοιχείου Πίνακας[i] με το στοιχείο Πίνακας[j]
    }
}
```

  Υπόδειξη: για το loop μην ξεχάσετε ότι οι αριθμοδείκτες πινάκων ξεκινούν από το 0, οπότε το πρώτο στοιχείο είναι το 0, το δεύτερο το 1, κ.ο.κ. Επίσης, διευκρινίζεται ότι το παραπάνω loop θα πρέπει να εκτελεστεί κανονικά όταν το i δείχνει στο δεύτερο στοιχείο (δηλαδή το i πάει **μέχρι και** το δεύτερο στοιχείο).

- Συνάρτηση cut

  Η συνάρτηση αυτή παίρνει ως είσοδο ένα δυναμικό πίνακα δυναμικών αντικειμένων (διπλό pointer) και κάνει μια διαδικασία που θυμίζει τον τρόπο που «κόβουμε φύλλα» (για να είμαστε σίγουροι ότι το άτομο που ανακάτεψε δεν έχει δώσει στα φύλλα κάποια συγκεκριμένη σειρά). Ο αλγόριθμος που θα πρέπει να χρησιμοποιήσετε είναι ο παρακάτω:

```
ΑλγόριθμοςΓιαΝαΚοπούνΤαΣτοιχείαΕνόςΠίνακα(Πίνακας, Μέγεθος){
    cutPoint = τυχαίο σημείο στο οποίο θα κοπούν τα στοιχεία του πίνακα μεταξύ του
               πρώτου και του τελευταίου στοιχείου του πίνακα (να βρεθεί τυχαία)
    Αντιγραφή των πρώτων cutPoint στοιχείων του πίνακα (από το πρώτο μέχρι και το
               στοιχείο αμέσως πριν το cutPoint) σε έναν πίνακα tempArray
    Μετακίνηση όλων των στοιχείων του πίνακα αριστερά μέχρι να φτάσουν στην αρχή
    Αντιγραφή των στοιχείων του πίνακα tempArray στις τελευταίες θέσεις του πίνακα
}
```

  Ένα παράδειγμα για τον παραπάνω αλγόριθμο είναι το παρακάτω:

  Πίνακας: | 3 | 4 | 2 | 6 | 5 | 7 | 9 | 8 | 1 | Μέγεθος: | 9 | cutPoint (τυχαία): | 3 |

  Αντιγραφή σε προσωρινό πίνακα tempArray: | 3 | 4 | 2 |

  Μετακίνηση των στοιχείων του πίνακα αριστερά: | **6** | **5** | **7** | **9** | **8** | **1** | 9 | 8 | 1 |

  Αντιγραφή των στοιχείων του tempArray στο τέλος του πίνακα: | 6 | 5 | 7 | 9 | 8 | 1 | **3** | **4** | **2** |

Σημειώστε ότι η υλοποίησή σας για τις συναρτήσεις shuffle και cut θα πρέπει να γίνει μέσα στο αρχείο shuffler.h, **δεν πρέπει να φτιάξετε ξεχωριστό αρχείο shuffler.cpp**.

Εγκατάσταση

Για να βάλετε το project στο CodeBlocks, αρκεί να αποσυμπιέσετε το αρχείο Tichu.zip σε μια τοποθεσία του σκληρού σας δίσκου, και στη συνέχεια από το περιβάλλον του CodeBlocks να επιλέξετε File → Open… και να επιλέξετε το Tichu.cbp από τον φάκελο που μόλις αποσυμπιέσατε το project.

Παρατηρήσεις

Η υλοποίηση θα πρέπει να γίνει στη C++ και να μπορεί να ανοίξει με το CodeBlocks, **με τις εκδόσεις που χρησιμοποιούμε** στο πλαίσιο του μαθήματος. Ο κώδικάς σας θα πρέπει να είναι καλά τεκμηριωμένος, ώστε να είναι παντού σαφείς οι λεπτομέρειες υλοποίησης.
Για την υλοποίηση, σας δίνονται τα αρχεία κεφαλίδων .h των κλάσεων/συναρτήσεων που πρέπει να υλοποιήσετε καθώς και κάποιες βοηθητικές κλάσεις/συναρτήσεις. Επιπλέον, σας δίνεται ο κώδικας της συνάρτησης main (main.cpp). **Σε καμία περίπτωση δεν επιτρέπεται να επέμβετε στον κώδικα των κλάσεων και των συναρτήσεων αυτών. Σε περίπτωση που το κάνετε, η εργασία σας αυτομάτως θεωρείται λανθασμένη και μηδενίζεται. Θα πρέπει να γράψετε κώδικα μόνο στα αρχεία cardimplementations.cpp και shuffler.h.**

Παραδοτέο

Το παραδοτέο θα είναι **ένα αρχείο zip με όνομα Tichu.zip** που θα περιλαμβάνει **όλο το project (τα αρχεία που θα υλοποιήσετε αλλά και αυτά που σας έχουν δοθεί)**, δηλαδή **ακριβώς ίδιο με το αρχείο Tichu.zip που δίνεται**, φυσικά **με τον κώδικα υλοποιημένο**. Επιπλέον, προτείνεται πριν δημιουργήσετε το αρχείο zip, να κάνετε Clean το project (που γίνεται από το Codeblocks επιλέγοντας στο μενού Build → Clean).

Προθεσμία υποβολής

Το παραδοτέο πρέπει να παραδοθεί μέχρι τις **23:59 της Δευτέρας 8 Ιουνίου**. Καμία παρέκκλιση δε θα γίνει από την παραπάνω προθεσμία.