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

ΕΡΓΑΣΙΑ ΣΤΟΝ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΗ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ

ΠΕΜΠΤΟ ΠΑΡΑΔΟΤΕΟ (ΠΡΟΑΙΡΕΤΙΚΟ)

Υλοποίηση έξυπνου παίκτη για το παιχνίδι σε C++

Το πέμπτο παραδοτέο της εργασίας αφορά τη δημιουργία ενός έξυπνου παίκτη για το παιχνίδι Tichu. Σημειώστε ότι **το παραδοτέο είναι προαιρετικό και δε προσμετράται στις μονάδες της εργασίας.** Αντί αυτού, οι υλοποιήσεις σας θα συμμετάσχουν σε ένα τουρνουά! Οι παίκτες της νικήτριας ομάδας θα πάρουν **bonus 1.5 μονάδας**, οι παίκτες της δεύτερης καλύτερης ομάδας θα πάρουν **bonus 1.0 μονάδας** και οι παίκτες της τρίτης καλύτερης ομάδας (που θα προκύψει από μικρό τελικό) θα πάρουν **bonus 0.5 μονάδας**.

Σημειώστε ότι **για να φτιάξετε έναν έξυπνο παίκτη θα πρέπει να έχετε υλοποιήσει πρώτα το τέταρτο παραδοτέο της εργασίας**. Πριν λοιπόν ξεκινήσετε, θα πρέπει να **αντικαταστήσετε τα αρχεία cardimplementations.cpp και shuffle.h** με τα υλοποιημένα αρχεία σας από το τέταρτο παραδοτέο.

Στη συνέχεια, θα πρέπει να υλοποιήσετε τον παίκτη σας στην κλάση SmartPlayer στο αρχείο **smartplayer.cpp**. Στο συγκεκριμένο αρχείο θα πρέπει αρχικά να θέσετε τη μεταβλητή team του παίκτη στον constructor ανάλογα με τον αριθμό της ομάδας σας χρησιμοποιώντας τρία ψηφία (π.χ. για την ομάδα 4 θα πρέπει να βάλετε όνομα "Team 004"). Στη συνέχεια, θα πρέπει να υλοποιήσετε τις εξής συναρτήσεις αποφάσεων για τον έξυπνο παίκτη:

1. Συνάρτηση setPhoenixValue
   Η συνάρτηση αυτή καλείται μόνο αν ο παίκτης έχει στην κατοχή του το Φοίνικα. Λαμβάνει ως είσοδο το τραπέζι του παιχνιδιού (από το οποίο μπορείτε να προσπελάσετε τους συνδυασμούς που έχει), τα status των παικτών (αν έχουν πάει πάσο ή αν έχουν παίξει), τον αριθμό των καρτών που έχει κάθε παίκτης στην κατοχή του και τον τελευταίο συνδυασμό του τραπεζιού. Η συνάρτηση επιστρέφει **έναν ακέραιο αριθμό από 1 έως και 14** η οποία θα είναι η τιμή του Φοίνικα. **Η συνάρτηση πρέπει οπωσδήποτε να επιστρέψει έναν ακέραιο σε αυτό το διάστημα**.
2. Συνάρτηση decideAndPlay
   Η συνάρτηση αυτή καλείται κάθε φορά που είναι η σειρά του παίκτη να παίξει. Λαμβάνει ως είσοδο το τραπέζι του παιχνιδιού (από το οποίο μπορείτε να προσπελάσετε τους συνδυασμούς που έχει), τα status των παικτών (αν έχουν πάει πάσο ή αν έχουν παίξει), τον αριθμό των καρτών που έχει κάθε παίκτης στην κατοχή του και τον τελευταίο συνδυασμό του τραπεζιού. Επίσης, λαμβάνει ως είσοδο **έναν δυναμικό πίνακα με του συνδυασμούς που μπορεί να παίξει ο παίκτης καθώς και το μέγεθος αυτού του πίνακα**. Η συνάρτηση επιστρέφει **έναν ακέραιο αριθμό ως αριθμοδείκτη του πίνακα που δηλώνει ποιος συνδυασμός θα παιχτεί** (π.χ. αν ο πίνακας είναι ο [PAIR[2:SWORD, 2:JADE], PAIR[3:SWORD, 3:STAR] , PAIR[4:SWORD, 4:STAR]], με μέγεθος 3, τότε ο παίκτης μπορεί να επιστρέψει την τιμή 0 αν θέλει να παίξει το συνδυασμό PAIR[2:SWORD, 2:JADE], την τιμή 1 αν θέλει να παίξει το συνδυασμό PAIR[3:SWORD, 3:STAR] ή την τιμή 2 αν θέλει να παίξει το συνδυασμό PAIR[4:SWORD, 4:STAR]]). Επίσης, ο παίκτης μπορεί **να επιστρέψει την τιμή -1 αν επιθυμεί να πάει πάσο**. Προσοχή: **σε περίπτωση που ο τελευταίος συνδυασμός του τραπεζιού είναι NULL, τότε ο παίκτης δεν πρέπει να επιστρέψει -1** (επειδή σε αυτήν την περίπτωση το τραπέζι δεν έχει κάποιο συνδυασμό).

Υπόδειξη: για να κατασκευάσετε τη στρατηγική σας, προτείνεται να χρησιμοποιήσετε τις παραμέτρους που δίνονται στις δύο μεθόδους. Όπως ήδη αναφέρθηκε, δίνεται πρόσβαση στο τραπέζι του παιχνιδιού (table), στα statuses όλων των παικτών (playerStatuses), στο πλήθος των καρτών κάθε παίκτη (numberOfCardsPerPlayer) και στον τελευταίο συνδυασμό του τραπεζιού (lastCombinationOnTable). Σημειώστε ότι ο παίκτης δεν μπορεί να δει όλα τα αντικείμενα του

παιχνιδιού (όπως π.χ. το χέρι των άλλων παικτών. Επίσης, ο παίκτης **απαγορεύεται να πειράξει τα αντικείμενα που δίνονται (π.χ. απαγορεύεται η χρήση των μεθόδων addCombinationOnTable και clear στο αντικείμενο table που δίνεται)**, μπορεί ωστόσο να δει την κατάστασή τους και ανάλογα να πάρει τις κατάλληλες αποφάσεις (π.χ. βλέπω ότι ο αντίπαλός μου που παίζει μετά από μένα έχει 3 κάρτες οπότε παίζω καρέ και θα είμαι βέβαιος ότι θα πάει πάσο).

Σημειώστε ότι οι παραπάνω συναρτήσεις **δεν κατασκευάζουν νέους συνδυασμούς ή κάρτες ή προσθέτουν/αφαιρούν συνδυασμούς ή κάρτες στο τραπέζι (αυτό απαγορεύεται καθώς αποτελεί επέμβαση στα αντικείμενα)**. Επίσης, **δεν ελέγχουν αν ένας συνδυασμός είναι έγκυρος** (μπορείτε φυσικά να το ελέγξετε αλλά δεν είναι αυτός ο σκοπός τους). Επιστρέφουν μόνο αυτά που αναφέρονται παραπάνω (η setPhoenixValue έναν ακέραιο στο διάστημα [1,14] και η decideAndPlay έναν ακέραιο στο διάστημα [-1, n-1] όπου n το πλήθος των συνδυασμών που μπορούν να παιχτούν). Οι υλοποιήσεις και οι έλεγχοι υλοποιούνται σε άλλες συναρτήσεις. Ο μόνος έλεγχος που θα πρέπει να κάνετε είναι στην decideAndPlay όπου, σε περίπτωση που επιθυμείτε ο παίκτης να πάει πάσο (άρα να επιστρέψει -1), **πρέπει οπωσδήποτε να ελέγξετε αν ο παίκτης έχει δικαίωμα να πάει πάσο (δηλαδή ότι ο τελευταίος συνδυασμός είναι NULL)**.

Για τη στρατηγική σας μην ξεχνάτε ότι πρέπει να μαζέψετε περισσότερους πόντους από την αντίπαλη ομάδα. Μπορείτε να μαζέψετε πόντους κερδίζοντας τα κατάλληλα φύλλα (υπάρχουν σύνολο 100 πόντοι που μπορούν να κατανεμηθούν και αρνητικά, π.χ. μέχρι σκορ 125 : -25). Επίσης, σε περίπτωση που βγουν πρώτοι οι δύο παίκτες μιας ομάδας (δηλαδή να τελειώσουν τα φύλλα τους πριν βγει κάποιος αντίπαλος), τότε το παιχνίδι λήγει και η ομάδα κερδίζει με σκορ 200 : 0.

Σημείωση για το τουρνουά: ουσιαστικά στο τουρνουά **κάθε ομάδα εργασίας θα εκπροσωπείται από δύο παίκτες** (θυμηθείτε ότι το Tichu παίζεται με δύο ομάδες των δύο παικτών η κάθε μία)! Οπότε θα κατασκευάζονται δύο αντικείμενα που θα λειτουργούν με βάση τον κώδικα που θα γράψετε στην SmartPlayer. Σε περίπτωση που θέλετε οι δύο παίκτες σας να ακολουθούν διαφορετική στρατηγική (π.χ. μπορεί να θέλετε να έχετε έναν επιθετικό κι έναν αμυντικό παίκτη) μπορείτε να το ελέγξετε χρησιμοποιώντας το index, π.χ. μέσα στη συνάρτηση decideAndPlay μπορείτε να γράψετε το παρακάτω:

```
if (index <= 1){
    // Implementation of the first player of the team
    ...
}
else{
    // Implementation of the second player of the team
    ...
}
```

Για να δοκιμάσετε τον παίκτη σας, **θα πρέπει να αλλάξετε κατάλληλα τις εντολές στις γραμμές 17, 18, 19 και 20 του αρχείου main.cpp**. Στις συγκεκριμένες γραμμές μπορείτε π.χ. να δοκιμάσετε τον παίκτη σας έναντι του απλού ComputerPlayer ως εξής:

```
players[0] = new ComputerPlayer(0);
players[1] = new SmartPlayer(1);
players[2] = new ComputerPlayer(2);
players[3] = new SmartPlayer(3);
```

Επίσης, μπορείτε π.χ. να δοκιμάσετε να παίξετε εσείς οι ίδιοι αντίπαλοι με τον παίκτη σας ως εξής:

```
players[0] = new HumanPlayer(0);
players[1] = new SmartPlayer(1);
players[2] = new HumanPlayer(2);
players[3] = new SmartPlayer(3);
```

Μπορείτε φυσικά να κάνετε αλλαγές και στον ComputerPlayer, ώστε να δείτε πώς ο παίκτης αντιμετωπίζει άλλες στρατηγικές (ή ακόμα και να αρχικοποιήσετε και όλους τους παίκτες με new SmartPlayer για να δείτε πώς ο παίκτης τα καταφέρνει ενάντια στον εαυτό του).

Σημείωση για τους κανόνες του Tichu: οι κανόνες του Tichu ισχύουν όπως περιγράφηκαν έως τώρα στα παραδοτέα της εργασίας. Επισημαίνεται ότι πλέον έχουν καλυφθεί **σχεδόν όλοι οι κανόνες, εκτός από τους παρακάτω**:

- Στην αρχή του παιχνιδιού οι παίκτες δεν ανταλλάσουν κάρτες μεταξύ τους (στους επίσημους κανόνες ο κάθε παίκτης δίνει τρεις κάρτες, από μία κάρτα σε κάθε έναν από τους άλλους τρεις παίκτες).
- Όταν κάποιος παίκτης παίζει το Mah Jong δε ζητείται κάποια κάρτα από τους άλλους παίκτες (στους επίσημους κανόνες ο παίκτης έχει δικαίωμα να ζητήσει να παιχτεί κάποια κάρτα).
- Στην αρχή του παιχνιδιού οι παίκτες δεν αναφωνούν Tichu ή Grand Tichu (στους επίσημους κανόνες ο παίκτης έχει δικαίωμα να αναφωνήσει Tichu/Grand Tichu ώστε να ρισκάρει να κερδίσει 100/200 πόντους ή να χάσει 100/200 πόντους ανάλογα με το αν θα βγει πρώτος από το παιχνίδι ή όχι).
- Αν κάποιος παίξει το Δράκο και κερδίσει (δηλαδή δεν παιχτεί από πάνω κάποια βόμβα) τότε τα φύλλα τα παίρνει αυτόματα ο αντίπαλος παίκτης που έχει τις περισσότερες κάρτες στο χέρι του (στους επίσημους κανόνες ο παίκτης έχει δικαίωμα να επιλέξει ποιος από τους δύο αντιπάλους θα πάρει τα φύλλα).

Πέραν των παραπάνω, **όλοι οι υπόλοιποι κανόνες ισχύουν κανονικά**. Σημειώστε (επιπρόσθετα με τους κανόνες που έχουν καλυφθεί έως τώρα) ότι αν κάποιος παίξει τα Σκυλιά (Dogs) τότε τη σειρά την παίρνει ο συμπαίκτης του. Και, τέλος, αν κάποιος παίξει το Φοίνικα (Phoenix) ως μονοφυλλία, τότε έχει αξία 0.5 παραπάνω από το τελευταίο φύλλο, π.χ. αν το τραπέζι έχει 7:PAGODA, τότε ο Φοίνικας (εφόσον παιχτεί ως μονοφυλλία), θα έχει αξία 7.5. **Αυτό δεν επηρεάζει την υλοποίησή σας στις συναρτήσεις αυτού του παραδοτέου, αφού γίνεται αυτόματα** (αν π.χ. ορίσετε το Φοίνικα ως 10 και τον ρίξετε πάνω σε 7, τότε αυτόματα θα μετατραπεί σε 7.5).

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 μόνο να αντιγράψετε τα αρχεία cardimplementations.cpp και shuffler.h από το τέταρτο παραδοτέο σας και να γράψετε κώδικα μόνο στο αρχείο smartplayer.cpp.**

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.