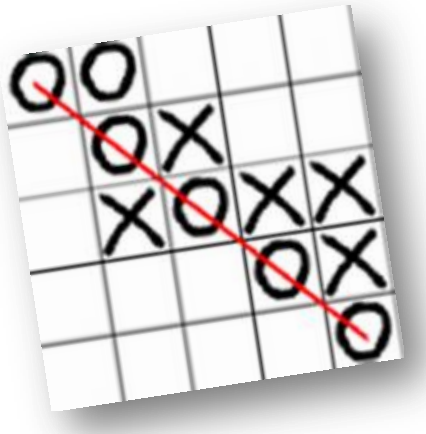
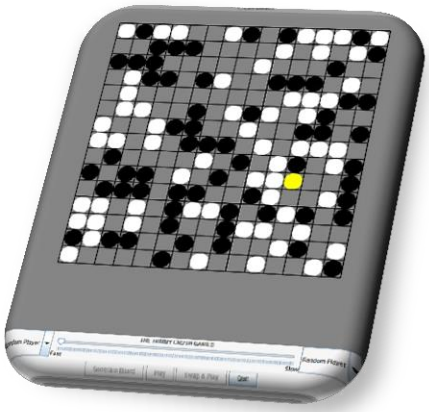
ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ
Τμήμα ηλεκτρολόγων μηχανικών & μηχανικών υπολογιστών

ΘΕΜΑ: «*Αναφορά για το πρώτο μέρος της εργασίας του μαθήματος Δομές Δεδομένων που βασίζεται στο παιχνίδι* **DS – Gomoku** »





```
public    int [] getNextMove(Board board){
    int[] position = new int[2];
    do{
        position[0] = (int)
(Math.floor(Math.random()*GomokuUtilities.NUMBER_OF_COLUMNS);
        position[1]= (int)
(Math.floor(Math.random()*GomokuUtilities.NUMBER_OF_ROWS);
    }while(board.getTile(position[0],position[1]).getColor()
);
    return position;
```

ΕΠΙΒΛΕΠΩΝ: ***Μήτκας Περικλής***

Ομάδα εργασίας:

- Triantafyllidis Theocharis / **Τριανταφυλλίδης Θεοχάρης**
  - **A.E.M.:** 7995
  - **E-mail:** theocharistr@gmail.com

Περιγραφή προβλήματος :

Το Gomoku είναι ένα 'επιτραπέζιο' παιχνίδι στρατηγικής ,στο οποίο συμμετέχουν 2 παίκτες. Σκοπός του παιχνιδιού είναι να βάλεις πρώτος 5 μάρκες συνεχόμενα κάθετα ή οριζόντια ή διαγώνια , οι παίκτες (άσπρος-μαύρος)παίζουν εναλλάξ. Πρώτος ξεκινά ο παίκτης με τα μαύρα . Κάθε παίκτης μπορεί να τοποθετήσει σε οποιαδήποτε κενή θέση του ταμπλό το πλακίδιο του. Όσον αφορά το πρώτο μέρος της εργασίας το παιχνίδι παίζεται σε ένα ταμπλό διαστάσεων 15x15. Τα δύο βασικά προβλήματα που έχουμε να αντιμετωπίσουμε στο μέρος αυτό είναι η υλοποίηση των κλάσεων Tile και RandomPlayer τροποποιώντας ή δημιουργώντας κάποιες συναρτήσεις. Εφόσον ικανοποιηθούν οι παραπάνω απαιτήσεις θα έχουμε την ομαλή λειτουργία του παιχνιδιού.

Επίλυση προβλήματος :

Κλάση Tile

Η κλάση αυτή είναι υπεύθυνη για την δημιουργία των πλακιδίων του ταμπλό του παιχνιδιού. Για την υλοποίηση της πρέπει να περιέχει τις απαραίτητες μεταβλητές και τους κατάλληλους constructors . Ωστόσο, πρέπει να τοποθετήσουμε και τους setters και getters των μεταβλητών για να τις επεξεργαστούμε και να τις προσπελάσουμε.

Προκειμένου να αντιμετωπίσουμε το πρόβλημά της κλάσης Tile θα πρέπει για αρχή να δηλώσουμε τια απαραίτητες μεταβλητές, οι οποίες είναι οι εξής :

- ✓ Int id : Περιέχει το μοναδικό κωδικό του πλακιδίου
- ✓ Int x,y: Η θέση του πλακιδίου στους άξονες x'x ,y'y
- ✓ Int color:Το χρώμα πλακιδίου (0 -άδεια θέση,1-μαύρο πλακίδιο,2-άσπρο πλακίδιο)
- ✓ boolean mark: βοηθητική μεταβλητή
- ✓ Int PlayerId: Υποδηλώνει από ποιόν έχει επιλεγεί η θέση που μας ενδιαφέρει (0 -κανέναν,1-Α παίκτη,2-Β παίκτη)

Οι μεταβλητές δηλώνονται σαν private γιατί θέλουμε να είναι προσπελάσιμες μόνο από μέλη της κλάσεως και να μην μπορούν να είναι άμεσα ορατές και προσπελάσιμες στον υπόλοιπο κώδικα.

Στη συνέχεια ,υλοποιούμε τον constructor που δέχεται σαν ορίσματα τις παραπάνω μεταβλητές(`Tile(int id, int x, int y, int color, boolean mark, int playerId)`). Ο constructor καλείται για να δημιουργήσει ένα αντικείμενο της κλάσης Tile, δηλαδή στην πράξη δεσμεύεται χώρος για τις μεταβλητές (id,x,y,color,mark,playerId) και για τις μεταβλητές των συναρτήσεων που περιέχει (setters π.χ. `setId(int id)`) αλλά γίνονται και οι εξ ορισμού

αρχικοποιήσεις. Αξιοποιούμε τη λέξη-κλειδί this και επιτυγχάνουμε προσπέλαση μελών (μεταβλητών) του ιδίου αντικειμένου της κλάσης Tile (`this.playerId = playerId;`).

Για κάθε μια μεταβλητή της κλάσης χρησιμοποιούμε τις συναρτήσεις setters και getters των μεταβλητών .

Για παράδειγμα:

```
public int getId() {return id;}
public void setId(int id) {this.id = id;}
```

Η πρόσθεση των setters και getters καθιστά την κατάσταση των μεταβλητών του πλακιδίου επεξεργάσιμη και προσπελάσιμη σε περίπτωση που κριθεί αυτό απαραίτητο. Οι setters ορίζουν την τιμή των μεταβλητών και οι getters επιστρέφουν την τιμή των μεταβλητών. Οι συναρτήσεις αυτές πρέπει να είναι public για είναι προσπελάσιμες και από τον υπόλοιπο κώδικα.

Κλάση RandomPlayer

Η κλάση αυτή εκτελεί την υλοποίηση των παικτών του παιχνιδιού. Όμοια με την προηγούμενη κλάση πρέπει να δηλωθούν οι απαραίτητες μεταβλητές , αλλά τώρα απαιτούνται δυο constructors.Ταυτόχρονα, πρέπει να προστεθούν οι setters και getters των μεταβλητών για μπορούμε να τις προσπελάσουμε. Ωστόσο, εδώ απαιτείτε και η δημιουργία της συνάρτησης getNextMove (Board board), η οποία θα προσδιορίζει την θέση που θα τοποθετεί ο παίκτης το πούλι.

Αρχικά, για την επίλυση του προβλήματος της κλάσης RandomPlayer δημιουργούμε τις απαιτούμενες μεταβλητές, οι οποίες είναι οι:

- ✓ int id: είναι μια μεταβλητή που παίρνει την τιμή 1 ή 2 ανάλογα με το αν ο παίκτης είναι ο άσπρος ή ο μαύρος
- ✓ String name: όνομα που επιθυμούμε στον παίκτη.
- ✓ int score: αποθηκεύει τον αριθμό των νικών του κάθε παίκτη.

Οι μεταβλητές και της κλάσεως αυτής δηλώνονται σαν private γιατί θέλουμε και εδώ να είναι προσπελάσιμες μόνο από μέλη της κλάσεως και να μην μπορούν να είναι άμεσα προσβάσιμες στον υπόλοιπο κώδικα.

Έπειτα δημιουργούμε τους δυο Constructors της κλάσης. Ο πρώτος constructor έχει σαν όρισμα την μεταβλητή pid(`public RandomPlayer(Integer pid)`) και καλείται για να δημιουργήσει ένα αντικείμενο της κλάσης RandomPlayer, δηλαδή για να δεσμευτεί χώρος για την μεταβλητή pid . Η συνάρτηση δόμησης αντιστοιχεί τις τιμές που έχουμε ορίσει για το id στην μεταβλητή pid. Ο δεύτερος έχει σαν ορίσματα τις παραπάνω μεταβλητές που ορίσαμε (`public RandomPlayer(Integer pid, String name, int score)`). Ο constructor καλείται για να δημιουργήσει τα αντικείμενα της κλάσης RandomPlayer, να δεσμεύεται

χώρος στην ουσία για τις μεταβλητές (id,name,score) και γίνονται οι εξ ορισμού αρχικοποιήσεις. Παράλληλα, αξιοποιούμε τη λέξη-κλειδί this και επιτυγχάνουμε προσπέλαση μελών (μεταβλητών) του αντικειμένου (`this.score = score;`). Στους δυο αυτούς constructors δηλώνουμε ότι `id = pid.intValue();`.Η `intValue()` συνάρτηση που έχει μέσα του ο τύπος δεδομένων Integer για να γίνει το casting σε int.

Κατόπιν για κάθε μια μεταβλητή της κλάσης χρησιμοποιούμε τις συναρτήσεις setters και getters των μεταβλητών .

Για παράδειγμα:

```
public int getScore()
public void setScore(int score)
```

Με την προσθήκη των setters και getters καθιστούμε προσπελάσιμη την κατάσταση των μεταβλητών του παίκτη όταν κριθεί αυτό αναγκαίο. Οι συναρτήσεις αυτές, όπως και προηγουμένως, πρέπει να είναι public για είναι προσβάσιμες και στον υπόλοιπο κώδικα.

Εν κατακλείδι πρέπει να υλοποιηθεί η συνάρτηση getNextMove (Board board). Η συγκεκριμένη συνάρτηση παίρνει σαν όρισμα την μεταβλητή board η οποία είναι αντικείμενο της κλάσης Board και επιστρέφει έναν μονοδιάστατο πίνακα ακεραίων, μεγέθους δύο (2) ο οποίος θα περιέχει την θέση (x,y) στην οποία θα τοποθετηθεί το επόμενο πούλι. Η θέση x,y επιλέγετε με τυχαίο τρόπο. Η συνάρτηση ορίζεται σαν public για να είναι προσπελάσιμη κι απο τον υπόλοιπο κώδικα που βρίσκεται σε άλλο πακέτο.

Για την υλοποίηση της συνάρτησης αυτή δημιουργούμε έναν μονοδιάστατο πίνακα δυο θέσεων (`int[] position = new int[2];`). Κατόπιν χρησιμοποιούμε έναν βρόγχο do-while για τον έλεγχο και την δέσμευση ενός πλακιδίου. Στο εσωτερικό της do βρίσκουμε την τυχαία θέση (x,y). Για την τυχαία εύρεση των (x,y) εκμεταλλευόμαστε την συνάρτηση double Math.random(), η οποία μας επιστρέφει έναν τυχαίο αριθμό τύπου double στο διάστημα [0,1). Ισχύει ότι `position[0]=x;` και `position[1]=y;`.

Οι εκφράσεις :

- `position[0] = (int) Math.floor(Math.random()*GomokuUtilities.NUMBER_OF_COLUMNS);`
- `position[1]= (int) Math.floor(Math.random()*GomokuUtilities.NUMBER_OF_ROWS);`

οδηγούν στην ανεύρεση τυχαίων θέσεων. Η `GomokuUtilities.NUMBER_OF_COLUMNS` και η `GomokuUtilities.NUMBER_OF_ROWS` δίνουν αντίστοιχα τις στήλες και τις γραμμές του ταμπλό που εδώ είναι 15 και τα δυο.

Η `(int)Math.floor(Math.random()*GomokuUtilities.NUMBER_OF_(ROWS or COLUMNS)` χρησιμοποιειται για να πάρω όλες τις ακέραιες τιμές μεταξύ [0,14] βασιζόμενη στην `double Math.random().` Κάνουμε typecasting (int) για να είναι το αποτέλεσμα μας ακέραιος (όπως η θέση (x,y)) κι όχι double που δίνει η Math.random(). Στην συνέχεια τοποθετούμε στην while την Συνθηκη για τον έλεγχο της ελευθερίας του πλακιδίου :

```
while(board.getTile(position[0],position[1]).getColor()!=0);
```

Το αντικείμενο board καλεί την συνάρτηση getTile(int x,int y) η οποία με την σειρά της καλεί την getColor() και συγκρίνουμε το χρώμα για να δούμε εάν είναι ελεύθερο (δηλαδή 0) το τυχαίο πλακίδιο του ταμπλό. Εάν είναι μηδέν μέσω της return έχουμε επιστροφή του πίνακα που περιέχει την θέση (x,y) εάν όχι εκτελείτε πάλι η do χωρίς να γίνει επιστροφή προηγουμένως.