ΕΡΓΑΣΙΑ PACMAN
ΜΕΡΟΣ Β'

Ονοματεπώνυμο: Άρης Ελευθέριος Παπαγγέλης
ΑΕΜ:8883

Ονοματεπώνυμο: Μιχαήλ Μηναδάκης
ΑΕΜ:8858

Περιγραφή του προβλήματος

Καλούμαστε να υλοποιήσουμε μία κλάση Node, η οποία θα δέχεται σαν ορίσματα μέσω του constructor τις συντεταγμένες του Pacman(x και y), την πιθανή κίνηση του(0,1,2 ή 3), καθώς και τον δισδιάστατο πίνακα που συμβολίζει τον χώρο παιχνιδιού.

Η κλάση αυτή θα περιέχει τρεις μεθόδους, οι οποίες βρίσκουν τις θέσεις των φαντασμάτων, τις θέσεις των σημαιών, καθώς και το αν κάποια από τις σημαίες έχει ήδη κατακτηθεί από τον Pacman.

Περιέχει επίσης μία μέθοδο evaluate, η οποία αξιολογεί την κάθε πιθανή κίνηση με βάση διάφορα κριτήρια, και επιστρέφει ένα "σκορ" που δηλώνει το πόσο καλή θεωρείται, δηλαδή το πόσο πιθανό είναι να οδηγήσει τον Pacman πιό κοντά στη νίκη.

Τέλος, η επιλογή της καλύτερης κίνησης θα γίνεται μέσα στην κλάση Creature μέσω ενός ArrayList που θα περιέχει αντικείμενα τύπου Node.

Ανάλυση του αλγορίθμου

Έχουμε τις εξής μεταβλητές στην κλάση Node:

nodeX: Περιέχει την συντεταγμένη X του Pacman
nodeY: Περιέχει την συντεταγμένη Y του Pacman
nodeMove: Περιέχει την πιθανή επόμενη κίνηση του Pacman
nodeEvaluation: Περιέχει μία αξιολόγηση του πόσο καλή είναι η πιθανή κίνηση
currentGhostPos: Πίνακας δύο διαστάσεων που περιέχει τις συντεταγμένες των φαντασμάτων
flagPos: Πίνακας δύο διαστάσεων που περιέχει τις συντεταγμένες των σημαιών
currentFlagStatus: Μονοδιάστατος πίνακας boolean που περιέχει true αν μία σημαία έχει ήδη κατακτηθεί, αλλιώς false
Maze: Πίνακας δύο διαστάσεων τύπου Room, ο οποίος συμβολίζει την περιοχή παιχνιδιού

Επίσης, έχουμε δύο Constructors:

public Node88838858(): Κενός constructor, μιας και είναι καλή πρακτική να δημιουργούμε έναν και να μην χρησιμοποιούμε τον default που δημιουργεί από μόνη της η Java.Έτσι, αρχικοποιούμε με μηδέν τις μεταβλητές μας και δεσμεύουμε μνήμη για τους πίνακες.

public Node88838858(int nX, int nY, int nM, Room[][] M):Αρχικοποιούμε με τιμές τις μεταβλητές μας, με ορίσματα τα οποία περνάμε απο την κλάση Creature(θα αναλυθεί παρακάτω). Οι μεταβλητές nodeX, nodeY, nodeMove και Maze αρχικοποιούνται μέσω των ορισμάτων, ενώ οι currentGhostPos, flagPos, currentFlagStatus και nodeEvaluation μέσω των μεθόδων που ακολουθούν, με την σειρά που αναγράφονται αντίστοιχα.

Οι μέθοδοι μας είναι:

private int[][] findGhosts (Room[][] PlayArea): Μέσα στη μέθοδο, προσπελαύνουμε όλο τον πίνακα PlayArea, και αν υπάρχει φάντασμα σε κάποια θέση, αποθηκεύονται οι συντεταγμένες του σε μια τοπική μεταβλητή pos[][].Ύστερα, επιστρέφεται αυτός ο πίνακας με μια εντολή return.

private int[][] findFlags (Room[][] PlayArea): Μέσα στη μέθοδο, προσπελαύνουμε όλο τον πίνακα PlayArea, και αν υπάρχει σημαία σε κάποια θέση, αποθηκεύονται οι συντεταγμένες της σε μια τοπική μεταβλητή
pos[][].Ύστερα, επιστρέφεται αυτός ο πίνακας με μια εντολή return.

private boolean[] checkFlags (Room[][] PlayArea): Μέσα στη μέθοδο αυτή, προσπελαύνουμε τον πίνακα με τις σημαίες που βρήκαμε με την προηγούμενη μέθοδο, και αν κάποια απο αυτές έχει ήδη κατακτηθεί, τότε το αντίστοιχο κελί του τοπικού πίνακα status[] γίνεται True, ενώ σε αντίθετη περίπτωση γίνεται False.Ύστερα επιστρέφεται ο πίνακας status με μία εντολή return.

public double evaluate (): Ουσιαστικά αυτή είναι η μέθοδος που υλοποιεί όλη τη λογική του προγράμματός μας. Θα γίνει ανάλυση για μία κατεύθυνση, έστω την δυτική, αλλά ακριβώς ίδια θα είναι και για τις άλλες τρεις κατευθύνσεις.

Η μεταβλητή που θα επιστρέφουμε είναι η evaluation. Αρχικά, προσπελαύνουμε όλες τις σημαίες, και αν υπάρχει σημαία που βρίσκεται δυτικά του Pacman, αυξάνουμε το evaluation ανάλογα με την απόσταση του απο αυτή. Δηλαδή, όσο πιο κοντά είναι σε μια σημαία, τόσο πιο πολύ θα θέλει να κινηθεί προς αυτήν. Βέβαια, ο έλεγχος γίνεται μόνο για

σημαίες που δεν έχουν κατακτηθεί ήδη από τον Pacman.

Ύστερα, προσπελαύνουμε όλα τα φαντάσματα, και για κάθε φάντασμα που είναι δυτικά του Pacman και στην ίδια γραμμή με αυτόν, μειώνουμε το evaluation ανάλογα με την απόσταση του από αυτόν. Δηλαδή, όσο πιο κοντά είναι ένα φάντασμα, τόσο πιο πολύ θα θέλει ο Pacman να κινηθεί προς άλλη κατεύθυνση.

Ελέγχουμε επίσης αν ένα φάντασμα είναι ακριβώς διαγώνια του Pacman προς την δυτική κατεύθυνση, γιατί σε αυτή την περίπτωση είναι πολύ πιθανό να τον φάει. Τότε, μειώνουμε ανάλογα το evaluation. Ο έλεγχος γίνεται μόνο όταν ΔΕΝ βρισκόμαστε στα άκρα του πίνακα, ώστε να μην βγούμε εκτός του πίνακα και έχουμε indexOutOfBounds error.

Έπειτα, ελέγχουμε αν η θέση δυτικά του Pacman είναι δίπλα σε τοίχο, και αν είναι, μειώνουμε το evaluation. Αυτό το κάνουμε διότι ο Pacman δεν θέλουμε να κινείται δίπλα στους τοίχους, αφού είναι μεγάλος ο κίνδυνος να εγκλωβιστεί από τα φαντάσματα.

Τέλος, να πούμε πως ο αριθμός που προσθέτουμε ή αφαιρούμε από το evaluation ειναι διαφορετικός σε κάθε συνθήκη, διότι κάποιες συνθήκες είναι πιό σημαντικές από άλλες. Ας πούμε, το να αποφύγει τα φαντάσματα είναι πιό σημαντικό από το να πιάσει την πιό κοντινή σημαία ή το να αποφύγει να κινηθεί κοντά σε τοίχους.

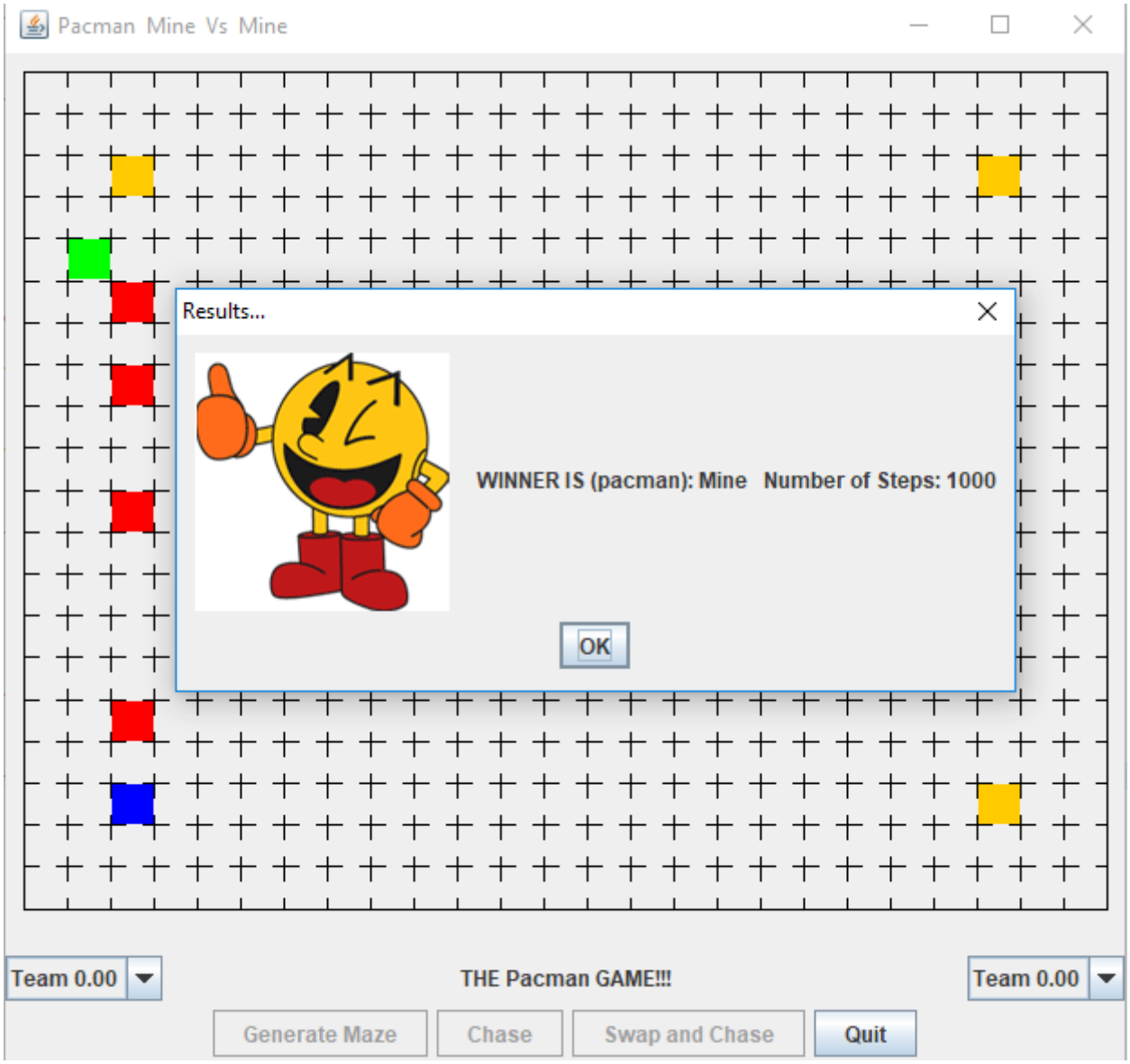
Μένει επίσης να αναλύσουμε πως χρησιμοποιούμε την Node88838858 στην κλάση Creature.

Φτιάχνουμε ένα ArrayList, στο οποίο βάζουμε αντικείμενα τύπου Node88838858. Μετά, μέσω ενός for loop απο 0 έως 3, δηλαδή για τις 4 κατευθύνσεις, ελέγχουμε αν υπάρχει τοίχος προς εκείνη την κατεύθυνση, και εφόσον δεν υπάρχει, καλούμε τον constructor του αντικειμένου moveI και ύστερα το αποθηκεύουμε στο ArrayList.

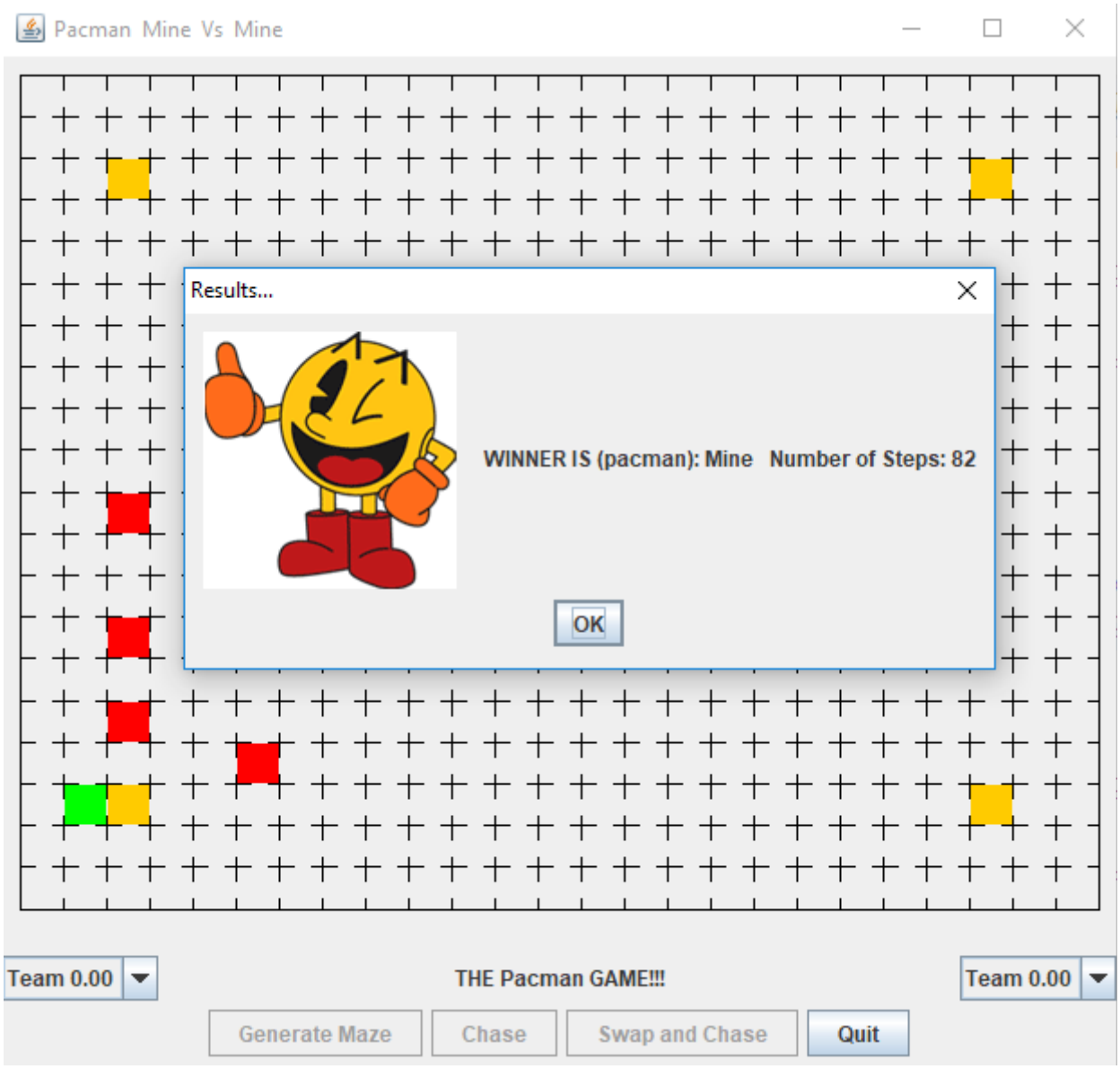
Εν τέλει, βρίσκουμε ποιό απο τα αντικείμενα του ArrayList έχει το μέγιστο nodeEvaluation, και επιστρέφουμε την κίνηση στην οποία αντιστοιχεί αυτό το μέγιστο. Αυτή θα είναι τελικά η κίνηση που θα πραγματοποιηθεί!

Επίδειξη ορθής λειτουργίας του προγράμματος

Νίκη μέσω βημάτων



Νίκη μέσω σημαιών



Το ποσοστό επιτυχίας της υλοποίησης μας γενικά είναι περίπου 2 στα 10 mazes, εφόσον παίζει μεγάλο ρόλο το πώς θα στηθούν αρχικά τα φαντάσματα.

Υ.Γ. Χρησιμοποίησα Verdana font, ελπίζω να είναι ΟΚ. Αλλιώς την επόμενη φορά να χρησιμοποιήσω Wingdings!