Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης

Εργασία Α’

Κίνηση Φαντασμάτων

Δομές Δεδομένων

Αλέξανδρος Σαχίνης
Αλέξανδρος Τζήκας

[Δημιουργία Αλγορίθμου σε Java
Έλεγχος Κινήσεων των Φαντασμάτων
Ανάλυση του Αλγορίθμου]

Περιγραφή του Προβλήματος

Σκοπός της εργασίας αυτής είναι η υλοποίηση (σε γλώσσα Java) μιας μεθόδου, η οποία ελέγχει την κίνηση των φαντασμάτων του παιχνιδιού Pacman και ονομάζεται “calculateNextGhostPosition”. Η μέθοδος αυτή δέχεται ως ορίσματα τον δισδιάστατο πίνακα τύπου Room που αναπαριστά το χώρο κίνησης των μονάδων και έναν δισδιάστατο πίνακα που περιέχει τις συντεταγμένες των φαντασμάτων πάνω στον χώρο κίνησης. Η συνάρτηση “calculateNextGhostPosition” επιστρέφει έναν μονοδιάστατο πίνακα που περιέχει την κατεύθυνση της επόμενης κίνησης του κάθε φαντάσματος, ακολουθώντας την σύμβαση:

West:0

South:1

East:2

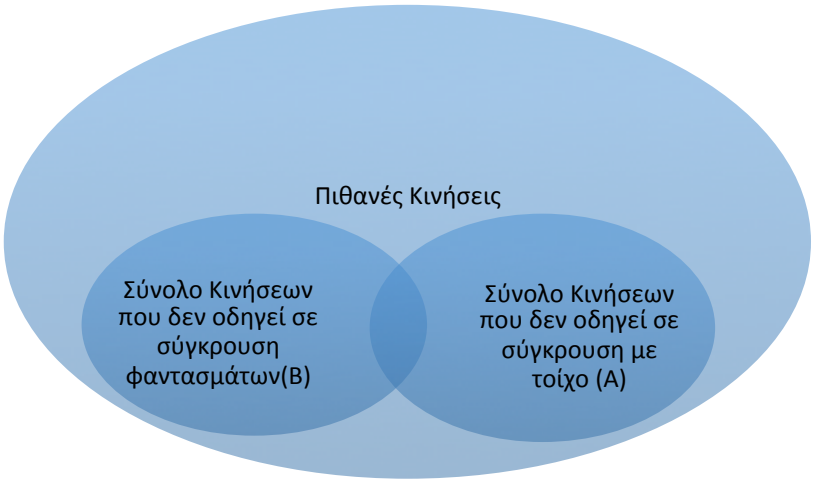
North:3

Τα φαντάσματα πρέπει να κινούνται τυχαία μέσα στον χώρο του παιχνιδιού, αλλά ταυτόχρονα το σύνολο των κινήσεων που επιστρέφει η “calculateNextGhostPosition” πρέπει να είναι έγκυρο. Αυτό σημαίνει ότι:

- η τυχαία κίνηση κάθε φαντάσματος δεν πρέπει να το οδηγεί σε σύγκρουση με τοίχο
- οι τυχαίες κινήσεις των φαντασμάτων δεν πρέπει να οδηγούν δύο ή περισσότερα φαντάσματα σε σύγκρουση μεταξύ τους

Η πρώτη προϋπόθεση ελέγχεται με χρήση της συνάρτησης int Maze[i][j].walls[k] και η δεύτερη με χρήση της μεθόδου checkCollision, ενώ η τυχαιότητα των κινήσεων εξασφαλίζεται με την χρήση της συνάρτησης Math.random().

Είναι εμφανές ότι από τις πιθανές κινήσεις των φαντασμάτων ένα μέρος των κινήσεων δεν οδηγεί τα φαντάσματα σε σύγκρουση με τοίχο και ένα άλλο μέρος των πιθανών κινήσεων δεν οδηγεί τα φαντάσματα σε σύγκρουση μεταξύ τους. Η επιθυμητή λύση είναι η τομή των δύο προαναφερθέντων υποσυνόλων κινήσεων, όπως φαίνεται στο σχήμα:



Περιγραφή και Ανάλυση Υλοποιημένου Αλγορίθμου

Αρχικά ορίζεται ένας μονοδιάστατος πίνακας nextGhostMoves ακεραίων με αριθμό κελίων ίσο με τον αριθμό των φαντασμάτων, ο οποίος και θα επιστρέφεται στο τέλος της μεθόδου. Αυτός ο πίνακας θα περιέχει τις επόμενες κινήσεις των φαντασμάτων (κάθε κελί θα περιέχει μια εκ των τιμών 0, 1, 2, 3). Ακόμα, ορίζεται μια boolean μεταβλητή mustChangeMoves και αρχικοποιείται σε false. Αυτή η μεταβλητή θα γίνει true σε περίπτωση που ένα σετ κινήσεων οδηγεί σε σύγκρουση μεταξύ φαντασμάτων.

Στη συνέχεια χρησιμοποιείται ένας βρόχος do-while:

1. *Στο τμήμα do{}:*

   Η μεταβλητή mustChangeMoves τίθεται σε τιμή false.

   Ο πίνακας nextGhostMoves λαμβάνει τιμές με μόνη προϋπόθεση οι τιμές-κινήσεις αυτές να μην οδηγούν τα φαντάσματα σε σύγκρουση με τοίχο. Αυτό γίνεται τοποθετώντας μια τιμή για την κίνηση του φαντάσματος και ελέγχοντας εάν στην κατεύθυνση της κίνησης αυτής η τωρινή θέση του φαντάσματος έχει τοίχο ή όχι. Εάν υπάρχει τοίχος δίνεται ξανά τιμή στην κίνηση του φαντάσματος αυτού (αφού η προηγούμενη τιμή θα το οδηγούσε σε τοίχο) και ξαναγίνεται ο έλεγχος. Σε αντίθετη περίπτωση ο αλγόριθμος συνεχίζει και ακολουθεί τα ίδια βήματα για το επόμενο φάντασμα. Έτσι λαμβάνει τιμές ο πίνακας nextGhostMoves, οι οποίες είναι σίγουρο ότι δεν οδηγούν τα φαντάσματα σε σύγκρουση με τοίχο. Βρισκόμαστε δηλαδή στο τμήμα A του παραπάνω σχήματος και θα ελέγξουμε εάν βρισκόμαστε και στο B.

   Τέλος λοιπόν, ελέγχουμε εάν η checkCollision επιστρέφει true (δηλαδή ύπαρξη σύγκρουσης φαντάσματος με φάντασμα) για κάποιο φάντασμα:

   i. Εάν βρεθεί έστω και μια τέτοια τιμή καταλαβαίνουμε ότι δεν βρισκόμαστε στο τμήμα $A \cap B$ και υπάρχει σύγκρουση μεταξύ φαντασμάτων. Άρα απαιτείται επανάληψη του βρόχου, δηλαδή:
      α) Αρχικοποίηση της mustChangeMoves σε false
      β) Εύρεση άλλου σετ κινήσεων που δεν οδηγεί σε σύγκρουση με τοίχο
      γ) Έλεγχος εάν αυτό το σετ οδηγεί ή οχι σε σύγκρουση μεταξύ φαντασμάτων
   ii. Αλλιώς έχουμε σετ κινήσεων που ανήκει στο $A \cap B$ και ο αλγόριθμος επιστρέφει τον πίνακα nextGhostMoves

2. *Στο τμήμα while():*

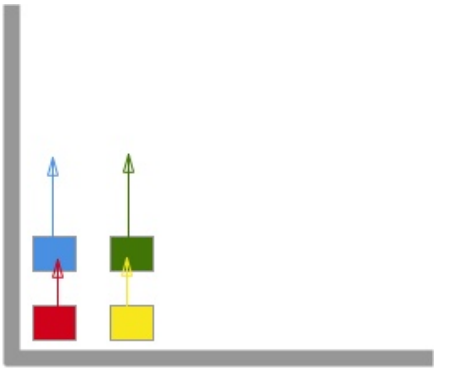
   Ελέγχεται σε ποια εκ των δύο περιπτώσεων (i) ή (ii) βρισκόμαστε με χρήση της μεταβλητής mustChangeMoves

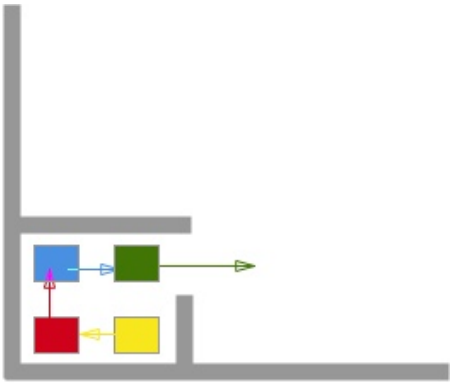
Είναι δεδομένο, λόγω της τυχαιότητας της Math.random(), ότι μετά από κάποιες επαναλήψεις του βρόχου do-while θα έχουν εξεταστεί όλοι οι συνδυασμοί κινήσεων που δεν οδηγούν σε σύγκρουση με τοίχο και θα έχει βρεθεί ένας από αυτούς που ταυτόχρονα ανήκει και στο

τμήμα Β (δηλαδή δεν οδηγεί σε σύγκρουση φαντάσματος με φάντασμα). Άρα θα λάβουμε λύση μετά από κάποιες επαναλήψεις.

Επίσης πάντα υπάρχει συνδυασμός κινήσεων που δεν οδηγέι σε σύγκρουση με τοίχο και ταυτόχρονα δεν οδηγεί σε σύγκρουση μεταξύ φαντασμάτων. Αυτός είναι ο εξής:

> Κίνηση ολών των φαντασμάτων προς μια (ίδια για όλα τα φαντάσματα, άρα 4 επιλογές) κατεύθυνση που δεν οδηγεί κάποιο φάντασμα σε σύγκρουση με τοίχο. Αυτό ισχύει έαν οι τοίχοι βρίσκονται περιμετρικά, αλλιώς μια περιστροφική κίνηση των φαντασμάτων είναι η λύση, οπως φαίνεται στα σχήματα:





Ο παραπάνω αλγόριθμος έτρεξε τουλάχιστον 50 φορές πειραματικά και δεν εμφάνισε κανένα error, έτρεχε δηλαδή ομαλά.