Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

HUNGER GAMES

MAY THE ODDS BE EVER IN YOUR FAVOR

ΜΕΡΟΣ Β

Φωτεινή Σαββίδου
(9657, sfoteini@ece.auth.gr)

Δομές Δεδομένων

Ακαδημαϊκό έτος
2019 - 2020

Πίνακας Περιεχομένων

Περιγραφή του παιχνιδιού Hunger Games

Το Hunger Games είναι ένα τηλεοπτικό παιχνίδι επιβίωσης. Οι παίκτες διαγωνίζονται σε ένα νησί του οποίου η διάμετρος μειώνεται με την πάροδο του χρόνου. Νικητής του παιχνιδιού αναδεικνύεται ο παίκτης που θα καταφέρει να παραμείνει ζωντανός και να επιβιώσει από τους κινδύνους που εγκυμονεί το νησί. Προκειμένου να το πετύχει αυτό πρέπει να αναζητήσει τροφή και όπλα, τα οποία βρίσκονται στο κέντρο του νησιού. Τα όπλα είναι απαραίτητα για την αντιμετώπιση των παγίδων που είναι κρυμμένες στο νησί, αλλά και για την εξόντωση των αντιπάλων.

Στην παρούσα εργασία υλοποιούμε μια απλουστευμένη προσομοίωση του παιχνιδιού Hunger Games. Συγκεκριμένα, δύο παίκτες διαγωνίζονται σε ένα ταμπλό αρχικής διάστασης 20×20. Σε κάθε γύρο του παιχνιδιού οι παίκτες μετακινούνται τυχαία κατά μία θέση (δεξιά, αριστερά, πάνω, κάτω, διαγώνια). Σκοπός τους είναι να φτάσουν στο κέντρο του ταμπλό, όπου είναι τοποθετημένα τα τρόφιμα και τα όπλα. Για να το πετύχουν, όμως, αυτό θα πρέπει να αποφύγουν τις διάφορες παγίδες που είναι τοποθετημένες περιμετρικά των εφοδίων. Παράλληλα, η διάσταση του ταμπλό μικραίνει και το παιχνίδι λήγει όταν το ταμπλό αποκτήσει διάσταση 4×4. Νικητής του παιχνιδιού αναδεικνύεται ο παίκτης που θα παραμείνει ζωντανός ή αυτός που θα συγκεντρώσει τους περισσότερους πόντους (σε περίπτωση που και οι δύο παίκτες επιβιώσουν).

Ένας παίκτης συγκεντρώνει πόντους όταν συλλέγει ένα εφόδιο, όταν δηλαδή μετακινηθεί σε ένα πλακίδιο του ταμπλό που περιλαμβάνει ένα εφόδιο. Αντίθετα, οι παίκτες χάνουν πόντους όταν συναντούν μια παγίδα και δεν διαθέτουν το κατάλληλο όπλο για να την αντιμετωπίσουν. Υπάρχουν τρία είδη όπλων: τόξο, σπαθί και πιστόλι. Το τόξο χρησιμοποιείται για την αντιμετώπιση των ζώων, το σπαθί για την αντιμετώπιση των παγίδων που περιλαμβάνουν σχοινιά, ενώ το πιστόλι για την εξόντωση του αντιπάλου.

Υλοποίηση του παιχνιδιού σε Java

Η προσομοίωση του παιχνιδιού Hunger Games βασίζεται στη δημιουργία ενός ταμπλό μεγέθους 20×20, που θα περιλαμβάνει συγκεκριμένο αριθμό όπλων, εφοδίων και παγίδων και δύο παικτών, οι οποίοι θα έχουν τη δυνατότητα να μετακινούνται, να συλλέγουν όπλα και τρόφιμα και να αντιμετωπίζουν παγίδες. Το δεύτερο μέρος της εργασίας αφορά τη δημιουργία ενός δυναμικού παίκτη, ο οποίος κινείται στο ταμπλό βάσει στρατηγικής. Ο παίκτης αυτός, δηλαδή, έχει τη δυνατότητα να αξιολογεί σε κάθε γύρο του παιχνιδιού τις διαθέσιμες κινήσεις του, με στόχο την επιλογή της καλύτερης κίνησης.

Η κλάση HeuristicPlayer

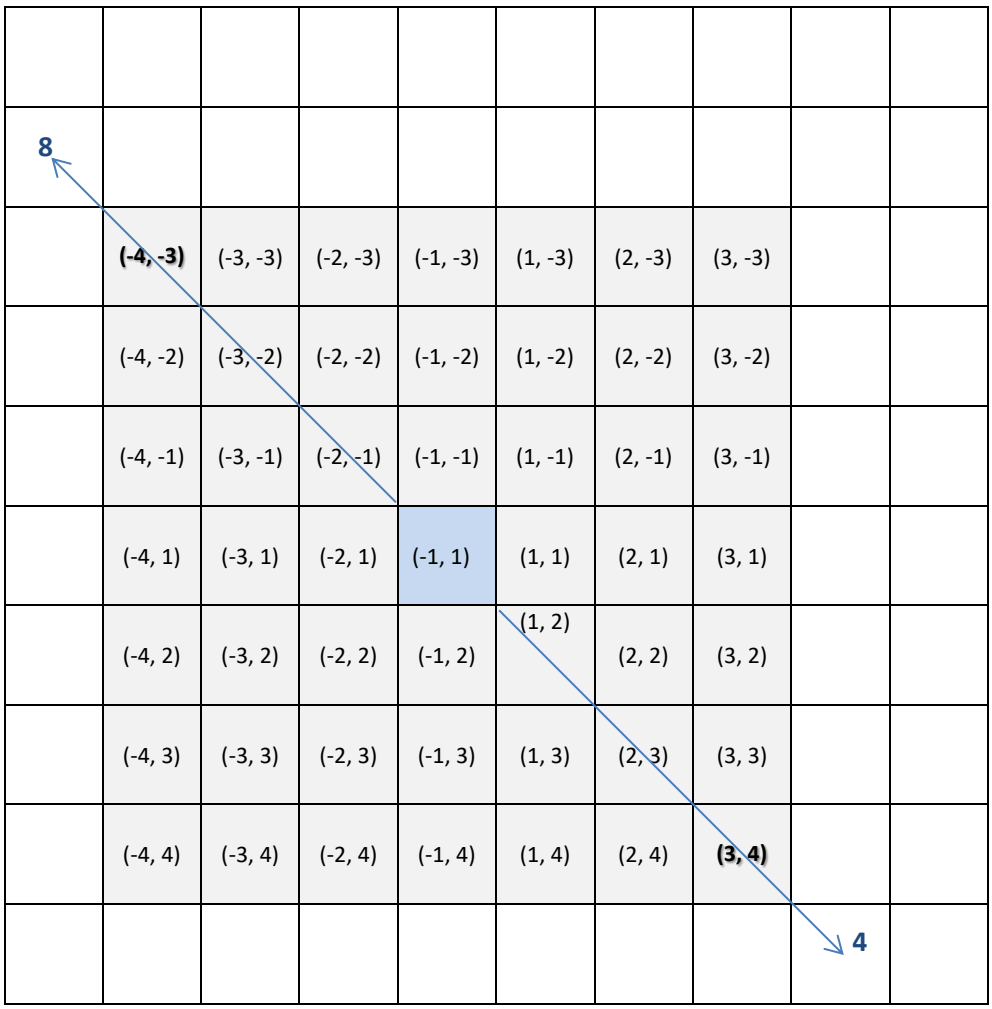
Η κλάση HeuristicPlayer αντιπροσωπεύει έναν δυναμικό παίκτη του παιχνιδιού. Η κλάση HeuristicPlayer κληρονομεί την κλάση Player. Κάθε δυναμικός παίκτης χαρακτηρίζεται από τις μεταβλητές της κλάσης Player και επιπλέον, από μία δομή του τύπου ArrayList (path), που περιλαμβάνει πληροφορίες για την κίνηση του παίκτη σε κάθε γύρο του παιχνιδιού και από την ακτίνα (r), που αναπαριστά το εύρος της περιοχής του ταμπλό που μπορεί να δει ο παίκτης. Συγκεκριμένα, στη δομή ArrayList αποθηκεύουμε τον αριθμό της κίνησης που επέλεξε ο παίκτης, τους πόντους που κέρδισε και πληροφορίες σχετικά με τον αριθμό των παγίδων που συνάντησε και τον αριθμό των τροφίμων και των όπλων (και το είδος τους) που συνέλεξε στην κίνησή του. Η ακτίνα r αναφέρεται στην περιοχή του ταμπλό που μπορεί να δει ο παίκτης ανάλογα με τη θέση του, η οποία (περιοχή) προκύπτει αν ο παίκτης μετακινηθεί κατά r πλακίδια προς οποιαδήποτε κατεύθυνση.

Υλοποιούμε δύο συναρτήσεις αρχικοποίησης (constructors). Η πρώτη συνάρτηση δεν έχει ορίσματα, καλεί την αντίστοιχη συνάρτηση αρχικοποίησης της κλάσης Player και δημιουργεί μία δομή του τύπου ArrayList. Η δεύτερη δέχεται έξι ορίσματα (id, name, board, score, x, y), καλεί την αντίστοιχη συνάρτηση αρχικοποίησης της κλάσης Player και δημιουργεί μία δομή του τύπου ArrayList.

Στις μεθόδους της κλάσης περιλαμβάνονται ακόμα οι συναρτήσεις εκχώρησης τιμής και επιστροφής της τιμής της μεταβλητής r (συναρτήσεις get και set).

Η συνάρτηση *fieldOfView()* υπολογίζει την περιοχή του ταμπλό (τετράγωνο) που μπορεί να δει ο παίκτης ανάλογα με τη θέση του και επιστρέφει τις συντεταγμένες δύο γωνιακών πλακιδίων: του πλακιδίου στην πάνω αριστερή γωνία και του πλακιδίου στην κάτω δεξιά γωνία. Αρχικά, μετακινούμε σταδιακά τον παίκτη κατά r θέσεις κατά μήκος της διεύθυνσης που ορίζεται από την κίνηση 8. Μετά από κάθε μετακίνηση ελέγχουμε (με τη συνάρτηση *isPositionValid()* της κλάσης Board) αν η νέα θέση είναι έγκυρη. Οι ζητούμενες

συντεταγμένες είναι οι συντεταγμένες της τελευταίας έγκυρης θέσης κατά τη μετακίνηση του παίκτη. Στη συνέχεια, ο παίκτης μετακινείται σταδιακά κατά r θέσεις κατά μήκος της διεύθυνσης που ορίζεται από την κίνηση 4 και επαναλαμβάνεται η παραπάνω διαδικασία. Οι συντεταγμένες των δύο γωνιακών πλακιδίων χρησιμεύουν στην εύρεση των οριακών τιμών των συντεταγμένων x, y των πλακιδίων που περικλείει η περιοχή. Στο παράδειγμα της *Εικόνας 1*, η συνάρτηση επιστρέφει τις τιμές (-4, -3) και (3, 4), οπότε η συντεταγμένη x κυμαίνεται από -4 έως 3, ενώ η συντεταγμένη y κυμαίνεται από -3 έως 4.



Εικόνα 1: Αναπαράσταση της περιοχής που μπορεί να δει ο παίκτης όταν βρίσκεται στη θέση (-1, 1) και είναι r = 3 (γκρι περιοχή). Η συνάρτηση *fieldOfView()* επιστρέφει τις συντεταγμένες (-4, -3) και (3, 4) των δυο γωνιακών πλακιδίων. Οι συντεταγμένες αυτές ορίζουν τα όρια των συντεταγμένων x, y των πλακιδίων που ανήκουν στην περιοχή αυτή.

Η συνάρτηση *playersDistance()* υπολογίζει και επιστρέφει την απόσταση ενός παίκτη από τον αντίπαλό του. Αν ο αντίπαλος δεν βρίσκεται στο οπτικό πεδίο του παίκτη (δηλαδή στην περιοχή που ορίζεται από την ακτίνα r), τότε η συνάρτηση επιστρέφει -1. Για τον υπολογισμό της απόστασης των δύο παικτών στο καρτεσιανό σύστημα που ορίσαμε,

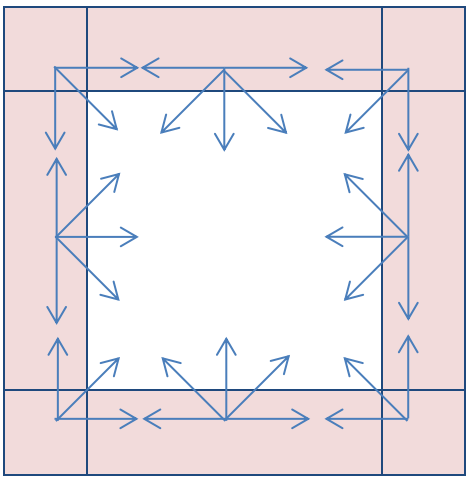
μετατρέπουμε τις συντεταγμένες x, y των παικτών σε συντεταγμένες j, i (αριθμοδείκτες πίνακα) με τις συναρτήσεις *x2j()*, *y2i()*[1] της κλάσης Board και εφαρμόζουμε τον γνωστό τύπο υπολογισμού της απόστασης δύο σημείων:

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Στη συνέχεια χρησιμοποιούμε τη συνάρτηση *fieldOfView()* για τον υπολογισμό του οπτικού πεδίου του παίκτη. Αν ο αντίπαλος βρίσκεται εντός του οπτικού πεδίου, επιστρέφουμε την απόσταση, αλλιώς επιστρέφουμε -1.

Η συνάρτηση *newPosition()* δέχεται ως όρισμα τον αριθμό της επιλεγμένης κίνησης και επιστρέφει τη νέα θέση του παίκτη. Για τον υπολογισμό της νέας θέσης, αποθηκεύουμε σε πίνακα σε μορφή συντεταγμένων τη μεταβολή των συντεταγμένων x, y για κάθε πιθανή κίνηση. Για παράδειγμα, όταν ο αριθμός της επιλεγμένης κίνησης είναι 1, τότε το x δεν μεταβάλλεται και το y μειώνεται κατά 1, οπότε αποθηκεύουμε το ζεύγος αριθμών: {0, -1}. Οι νέες συντεταγμένες προκύπτουν αν στις τρέχουσες συντεταγμένες του παίκτη προσθέσουμε το κατάλληλο ζεύγος συντεταγμένων του παραπάνω πίνακα. Στη συνέχεια ελέγχουμε αν οι νέες συντεταγμένες είναι έγκυρες (με τη συνάρτηση *isPositionValid()* της κλάσης Board) και τις επιστρέφουμε, αλλιώς επιστρέφουμε τις τρέχουσες συντεταγμένες του παίκτη.

Η συνάρτηση *possibleMoves()* επιστρέφει τους αριθμούς των διαθέσιμων κινήσεων του παίκτη ανάλογα με τη θέση του. Όταν ο παίκτης βρίσκεται στις γωνίες του ταμπλό οι διαθέσιμες κινήσεις είναι 3, ενώ όταν βρίσκεται στα πλακίδια των εξωτερικών γραμμών και στηλών οι διαθέσιμες κινήσεις είναι 5, όπως φαίνεται στην *Εικόνα 2*. Για αριθμούς από το 1 έως και το 8, καλούμε τη συνάρτηση *newPosition()* για τον υπολογισμό των πιθανών νέων συντεταγμένων. Αν οι συντεταγμένες είναι έγκυρες, τότε ο αντίστοιχος αριθμός της κίνησης είναι διαθέσιμος για τον παίκτη. Τελικά, επιστρέφουμε έναν πίνακα με τους διαθέσιμους αριθμούς κινήσεων.



Εικόνα 2: Αναπαράσταση των διαθέσιμων κινήσεων του παίκτη στα εξωτερικά πλακίδια του ταμπλό.

[1] Οι συναρτήσεις *x2j()* και *y2i()* της κλάσης Board μετατρέπουν τις συντεταγμένες x, y σε συντεταγμένες j, i (αριθμοδείκτες πίνακα). Στο σύστημα συντεταγμένων που ορίσαμε, στοιχεία που βρίσκονται στην ίδια στήλη έχουν ίδια συντεταγμένη x, οπότε την αντιστοιχίζουμε στον αριθμοδείκτη στήλης j. Αντίστοιχα, το y αντιστοιχίζεται στον αριθμοδείκτη γραμμής i.

Η συνάρτηση *evaluate()* αξιολογεί την κίνηση του παίκτη. Δέχεται ως όρισμα τον αριθμό της κίνησης και επιστρέφει έναν πραγματικό αριθμό. Αν ο αριθμός είναι θετικός, τότε η κίνηση αξιολογείται ως καλή για τον παίκτη, ενώ αν ο αριθμός είναι αρνητικός, τότε ο παίκτης πρέπει να αποφύγει αυτήν την κίνηση. Γενικότερα, στόχος των παικτών είναι να κερδίσουν πόντους από τα εφόδια, να συλλέξουν τα όπλα τους, να αποφύγουν τις παγίδες που δεν μπορούν να αντιμετωπίσουν και να σκοτώσουν τον αντίπαλό τους. Έτσι, η συνάρτηση αξιολόγησης που υλοποιούμε βασίζεται στην επιστροφή θετικών αριθμών στην περίπτωση που ο παίκτης μπορεί να σκοτώσει τον αντίπαλό του, να κερδίσει πόντους από ένα εφόδιο ή να συλλέξει ένα όπλο και στην επιστροφή αρνητικών αριθμών όταν ο παίκτης συναντά παγίδα που δεν μπορεί να αντιμετωπίσει ή κινδυνεύει να σκοτωθεί από τον αντίπαλό του. Η τιμή 0 επιστρέφεται όταν η κίνηση είναι αδιάφορη για τον παίκτη.

Αρχικά, υπολογίζουμε τη νέα θέση του παίκτη με τη συνάρτηση *newPosition()*. Καλείται η συνάρτηση *playersDistance()* για τον υπολογισμό της νέας απόστασης των παικτών και διακρίνουμε τις εξής δύο περιπτώσεις:

- Αν η απόσταση του παίκτη από τον αντίπαλό του είναι στο διάστημα [0, 2) και ο παίκτης κατέχει το πιστόλι, τότε επιστρέφεται μία μεγάλη θετική τιμή (π.χ. 1000), έτσι ώστε ο παίκτης να επιλέξει αυτήν την κίνηση (καθώς μπορεί να σκοτώσει τον αντίπαλό του).
- Αν η απόσταση των δύο παικτών βρίσκεται στο διάστημα [0, 3) και ο αντίπαλος έχει το πιστόλι, τότε επιστρέφεται μία αρνητική τιμή (π.χ. -1000), έτσι ώστε ο παίκτης να μην επιλέξει αυτήν την κίνηση (διότι κινδυνεύει να σκοτωθεί).

Στη συνέχεια, ορίζουμε μία συνάρτηση στόχου της μορφής:

$$wPoints \times gainPoints + wWeapons \times gainWeapons$$

όπου η μεταβλητή gainPoints παίρνει θετικές και αρνητικές τιμές ανάλογα με το αν η κίνηση προσφέρει ή αντίστοιχα αφαιρεί πόντους από τον παίκτη (ύπαρξη εφοδίου ή παγίδας αντίστοιχα) και η μεταβλητή gainWeapons παίρνει μόνο θετικές τιμές και αφορά τη συλλογή ενός όπλου. Οι μεταβλητές wPoints και wWeapons είναι αντίστοιχα οι συντελεστές (βάρη) των μεταβλητών gainPoints και gainWeapons και αρχικοποιούνται με την τιμή μηδέν. Διακρίνουμε τις εξής περιπτώσεις:

- Αν οι πόντοι του παίκτη είναι μικρότεροι του μηδενός ή μικρότεροι των πόντων του αντιπάλου του, τότε αυξάνουμε τη μεταβλητή wPoints και μειώνουμε τη μεταβλητή wWeapons, ώστε ο παίκτης να επιλέγει πλακίδια που περιλαμβάνουν εφόδια.
- Αν ο παίκτης κατέχει όλα τα όπλα του, τότε θέτουμε τη μεταβλητή wWeapons ίση με το μηδέν.
- Αν ο παίκτης κατέχει το τόξο ή/και το σπαθί, τότε μειώνουμε τη μεταβλητή wWeapons και αυξάνουμε τη μεταβλητή wPoints.

Αν το νέο πλακίδιο περιέχει παγίδα και ο παίκτης δεν έχει το κατάλληλο όπλο για να την αντιμετωπίσει, τότε εκχωρείται στη μεταβλητή gainPoints τιμή ίση με τους πόντους της παγίδας (αρνητική τιμή). Σε αντίθετη περίπτωση η μεταβλητή gainPoints έχει την τιμή 0 (η κίνηση είναι αδιάφορη για τον παίκτη).

Αν το νέο πλακίδιο περιέχει εφόδιο, τότε εκχωρείται στη μεταβλητή gainPoints τιμή ίση με τους πόντους του εφοδίου (θετική τιμή).

Αν το νέο πλακίδιο περιέχει όπλο, τότε αν είναι πιστόλι εκχωρείται στη μεταβλητή gainWeapons η τιμή 1000 (έτσι ώστε ο παίκτης να επιλέξει αυτήν την κίνηση), αλλιώς εκχωρείται η τιμή 10 (οπότε η πιθανότητα να επιλέξει ο παίκτης αυτήν την κίνηση είναι μεγάλη).

Η συνάρτηση *kill()* ελέγχει αν η απόσταση μεταξύ των παικτών είναι μικρότερη ενός δεδομένου αριθμού d, οπότε ο παίκτης που παίζει τη συγκεκριμένη στιγμή μπορεί να σκοτώσει τον αντίπαλό του. Σε αυτήν την περίπτωση επιστρέφεται true, αλλιώς επιστρέφεται false. Η συνάρτηση *kill()* καλείται με πρώτο όρισμα τον τρέχων παίκτη, δεύτερο όρισμα τον αντίπαλό του και τρίτο έναν αριθμό d. Για τον υπολογισμό της απόστασης των δύο παικτών χρησιμοποιούμε τη συνάρτηση *playersDistance()* της κλάσης HeuristicPlayer. Έτσι, πρέπει πρώτα να ελέγξουμε (με τον τελεστή instanceof) ποιος από τους δύο παίκτες είναι του τύπου HeuristicPlayer (ώστε να μπορούμε να καλέσουμε τη συνάρτηση). Τελικά, αν η απόσταση μεταξύ των παικτών βρίσκεται στο διάστημα [0, d) και ο παίκτης που παίζει τη συγκεκριμένη στιγμή έχει το πιστόλι, τότε επιστρέφεται η τιμή true.

Η συνάρτηση *move()* επιλέγει τη βέλτιστη κίνηση του παίκτη, μετακινεί τον παίκτη στη νέα θέση και επιστρέφει σε μορφή πίνακα τις νέες συντεταγμένες. Δημιουργούμε μία δομή του τύπου HashMap<Integer, Double>. Με τη συνάρτηση *possibleMoves()* βρίσκουμε τους αριθμούς των διαθέσιμων κινήσεων του παίκτη και αποθηκεύουμε στην παραπάνω δομή τον αριθμό και την αξιολόγηση κάθε πιθανής κίνησης. Για την επιλογή της καλύτερης κίνησης απαιτείται η εύρεση της μεγαλύτερης τιμής που περιέχει η δομή. Έτσι, βρίσκουμε τη μέγιστη τιμή και αποθηκεύουμε σε μία λίστα (ArrayList<Integer>) τους αριθμούς των κινήσεων που αντιστοιχούν στην καλύτερη αξιολόγηση (μέγιστη τιμή). Διακρίνουμε δύο περιπτώσεις:

- Αν η λίστα περιέχει ένα στοιχείο, τότε αυτό αντιστοιχεί στον αριθμό της κίνησης του παίκτη.
- Αν η λίστα περιέχει περισσότερα του ενός στοιχεία, τότε επιλέγεται τυχαία ένα από τα στοιχεία αυτά.

Μετά την εύρεση του αριθμού της κίνησης καλείται η συνάρτηση *newPosition()* για την εύρεση των συντεταγμένων της νέας θέσης και ο παίκτης μετακινείται σ' αυτή.

Το επόμενο μέρος της συνάρτησης αφορά την καταγραφή των πληροφοριών της κίνησης και την ενημέρωση της μεταβλητής path του παίκτη. Σε περίπτωση που στη νέα θέση υπάρχει όπλο (το οποίο μπορεί να πάρει ο παίκτης), μηδενίζονται οι συντεταγμένες του όπλου και ανάλογα με το είδος του όπλου ενημερώνεται το αντίστοιχο αντικείμενο του τύπου Weapon του παίκτη. Αν η νέα θέση περιέχει εφόδιο, τότε μηδενίζονται οι συντεταγμένες του εφοδίου και προστίθενται στους βαθμούς του παίκτη οι πόντοι του εφοδίου. Αντίστοιχα, αν στη νέα θέση υπάρχει παγίδα και ο παίκτης δεν έχει το κατάλληλο όπλο για να την αντιμετωπίσει, τότε αφαιρούνται από τους βαθμούς του παίκτη οι πόντοι της παγίδας.

Η συνάρτηση *statistics()* εκτυπώνει πληροφορίες για την κίνηση του παίκτη σε κάθε γύρο του παιχνιδιού. Εκτυπώνει τον αριθμό της κίνησης του παίκτη, του πόντους που κέρδισε, τον αριθμό των εφοδίων, παγίδων και όπλων (και το είδος τους) που συνέλεξε.

Η κλάση Game

Η κλάση Game αντιπροσωπεύει το παιχνίδι, το οποίο χαρακτηρίζεται από μία μόνο μεταβλητή: τον τρέχων γύρο (round).

Υλοποιούμε δύο συναρτήσεις αρχικοποίησης (constructors). Η πρώτη συνάρτηση δεν έχει ορίσματα και αρχικοποιεί τη μεταβλητή του αντικειμένου σε μηδέν και η δεύτερη δέχεται ως όρισμα έναν ακέραιο αριθμό, οποίος εκχωρείται στη μεταβλητή round.

Στις μεθόδους της κλάσης περιλαμβάνονται ακόμα οι συναρτήσεις εκχώρησης τιμής και επιστροφής της τιμής της μεταβλητής της κλάσης (συναρτήσεις get και set).

Η συνάρτηση *main()* είναι η κύρια συνάρτηση του προγράμματος. Δημιουργούμε ένα ταμπλό διάστασης 20×20 με 6 όπλα, 10 εφόδια και 8 παγίδες, τυχαία κατανεμημένα στο ταμπλό και ορίζουμε την ακτίνα (r) της περιοχής που μπορεί να δει ο παίκτης σε κάθε του κίνηση σε 3. Ορίζουμε δύο παίκτες· ο πρώτος είναι του τύπου HeuristicPlayer και βρίσκεται στην πάνω αριστερή γωνία του ταμπλό και ο δεύτερος είναι του τύπου Player και βρίσκεται στην κάτω δεξιά γωνία. Σε κάθε γύρο του παιχνιδιού οι παίκτες παίζουν εναλλάξ και μετακινούνται κατά μία θέση, εκτυπώνεται το ταμπλό και η κατάσταση του κάθε παίκτη (νέα θέση, αριθμός όπλων, βαθμοί). Μετά τη μετακίνηση του κάθε παίκτη καλείται η συνάρτηση *kill()*. Αν ένας από τους δύο παίκτες μπορεί να σκοτώσει τον αντίπαλό του, τότε εμφανίζεται κατάλληλο μήνυμα και το παιχνίδι τερματίζεται.

Ορίζουμε μία μεταβλητή resizeTime, η οποία αυξάνεται σε κάθε γύρο του παιχνιδιού και χρησιμοποιείται για να ελέγξουμε πότε πρέπει να μικρύνει το ταμπλό. Συγκεκριμένα, όταν η μεταβλητή resizeTime είναι μεγαλύτερη ή ίση του τρία, καλούμε τη συνάρτηση *resizeBoard()*. Αν οι διαστάσεις του ταμπλό μικρύνουν (αυτό συμβαίνει όταν κανένας παίκτης δε βρίσκεται στα εξωτερικά πλακίδια), τότε η τιμή της μεταβλητής resizeTime γίνεται μηδέν. Αν το ταμπλό δεν μικρύνει, τότε επαναλαμβάνεται ο έλεγχος στον επόμενο γύρο. Το παιχνίδι συνεχίζεται όσο οι διαστάσεις N, M του ταμπλό είναι μεγαλύτερες του τέσσερα ή μέχρις ότου ένας από τους δύο παίκτες σκοτωθεί. Στο τέλος του παιχνιδιού τυπώνεται το ταμπλό, οι συνολικοί γύροι που έπαιξαν οι παίκτες και οι βαθμοί του κάθε παίκτη. Νικητής είναι ο παίκτης που επιβίωσε ή ο παίκτης που συγκέντρωσε τους περισσότερους πόντους (σε περίπτωση που και οι δύο παίκτες είναι ζωντανοί).