Εργασία 2

Ονοματεπώνυμο: Μαραντίδης Θεοφάνης

ΑΜ: 1115201800106

Question 1

Στην πρώτη ερώτηση του Pacman φτιάχνουμε την evaluationFunction για την επιλογή του καλύτερου state του pacman.Λαμβάνουμε υπόψη το τρέχον score του pacman, την θέση του και τις θέσεις των υπολειπόμενων φαγητών.

Υπολογίζουμε λοιπόν την απόσταση του κοντινότερου φαγητού από την τρέχουσα θέση του pacman καθώς και την απόσταση του pacman από το κοντινότερο φάντασμα.

Ελέγχουμε στην συνέχεια αν ο pacman βρίσκεται σε θέση με φαγητό και του αποδίδουμε στο τρέχον score ένα σταθερό reward το οποίο έχουμε ορίσει στην τιμή 5.

Διακρίνουμε και τις περιπτώσεις όπου ένα ghost βρίσκεται πολύ κοντά οπότε μειώνουμε το score κατά 100 μονάδες ως penalty.

Τέλος επιστρέφουμε την τιμή του score προσθέτοντας μία εκτίμηση του σταθερού reward δια την κοντινότερη απόσταση.

Question 2

Στο ερώτημα αυτό καλούμαστε να υλοποιήσουμε τον αλγόριθμο minimax. Η υλοποίηση βασίζεται σε μεγάλο βαθμό στις διαφάνειες του μαθήματος.Συγκεκριμένα υλοποιούμε τις συναρτήσεις min_value , max_value και την getAction η οποία καλεί την max_value με index το 0 (παίζει πρώτα ο pacman) και το αντίστοιχο βάθος.

Τόσο ο maximizer όσο και ο minimizer επιστρέφουν μία τιμή

(του max και του min αντίστοιχα) αλλά και την αντίστοιχη κίνηση (action).

Ο maximizer για κάθε επιτρεπτή κίνηση αναπαράγει έναν απόγονο και υπολογίζει το score του με την βοήθεια της minimizer αφού ο απόγονος βρίσκεται στο min επίπεδο.

Επιστρέφουμε την μεγαλύτερη τιμή των απογόνων καθώς και την κίνηση που κάνουμε για να φτάσουμε εκεί.

Αντίστοιχα στο minimizer ελέγχουμε αν υπάρχουν ακόμα διαθέσιμα ghost να παίξουν (min player's) και κάνουμε την αντίστοιχη διαδικασία με τους απογόνους μέχρι να έχουν τελειώσει την σειρά τους τόσο όλα τα ghosts όσο και ο pacman.Για τον τελευταίο καλείται η maximizer αφου πρόκειται για max κόμβο.Επιστρέφουμε πάλι την αντίστοιχη τιμή με την κίνηση που κάνουμε να φτάσουμε εκεί.

Question 3

Στο ερώτημα 3 καλούμαστε να υλοποιήσουμε τον αλγόριθμο alpha beta pruning. Η υλοποίηση γίνεται βάση των διαφανειών του μαθήματος. Έχουμε μία βελτιωμένη έκδοση του minimax αλγορίθμου από το προηγούμενο ερώτημα με την διαφορά ότι δεν εξετάζουμε όλες τις καταστάσεις διότι κλαδέβουμε ορισμένους κόμβους βάση των παραμέτρων α,β που έχουμε ορίσει και των περιορισμών τους.

Συγκεκριμένα έχουμε τις συναρτήσεις MaxValue και MinValue.Η τελευταία καλείται μέσω την getAction για τον Pacman (index = 0) και με alpha,beta -∞ και +∞ αντίστοιχα. Και αυτή με την σειρά της καλεί την MinValue για όλους τους απογόνους του Pacman.

Εν τέλει επιστρέφεται η καλύτερη επόμενη κίνηση για τον MAX (Pacman) που μπορούμε να φτάσουμε από την τρέχουσα κατάσταση.

Question 4

Στο ερώτημα αυτό υλοποιούμε τον αλγόριθμο Expectimax.

Αντίστοιχα με το minimax υπολογίζουμε τις αναμενόμενες τιμές αυτή τη φορά ως τον μέσο όρων των τιμών για όλα τα δυνατά αποτελέσματα των κόμβων τύχης.

Έχουμε τις συναρτήσεις maxValue,expValue,getAction.

Για κάθε επιτρεπτή κίνηση του pacman υπολογίζουμε την αναμενόμενη τιμή μέσω της expValue.

Στην τελευταία ελέγχουμε αν έχουμε εναπομείναντα ghosts να παίξουν και καλούμε αναδρομικά την expValue για καθένα από τους απογόνους των ghosts και τις δυνατές ενέργειες τους.

Εν τέλη επιστρέφουμε το άθροισμα των τιμών αυτών δια το πλήθος των επιτρεπτών ενεργειών για να πάρουμε την αντίστοιχη αναμενόμενη τιμή.

Εάν όλα τα ghosts έχουν παίξει επιστρέφουμε το άθροισμα της maxValue για κάθε έναν από τους απογόνους του Pacman στο επόμενο επίπεδο και για όλες τις δυνατές ενέργειές τους.

Διαιρούμε τελικά αυτό το άθροισμα πάλι με το πλήθος των επιτρεπτών ενεργειών για να σχηματίσουμε την αναμενόμενη τιμή μας.

Question 5

Στο ερώτημα αυτό υλοποιούμε μία καλύτερη εκδοχή της evaluationFunction.

Επιστρέφουμε την εκτίμηση για μία κατάσταση παιχνιδιού με βάση τα βάρη των χαρακτηριστικών που χρησιμοποιούμε.

Αναλυτικά χρησιμοποιούμε την απόσταση προς το κοντινότερο φαγητό,την απόσταση προς το κοντινότερο φάντασμα αλλά και την απόσταση από την κοντινότερη κάψουλα.Επίσης λαμβάνουμε υπόψη τα φαγητά και τις κάψουλες που έχουν απομείνει.

Πιο αναλυτικά, αν ένα φάντασμα βρίσκεται πολύ κοντά στο pacman (απόσταση <=1) ή έχουμε κατάσταση νίκης ή ήττας αντίστοιχα επιστρέφουμε μία πολύ μεγάλη τιμή για να αποφύγουμε αυτή την κατάσταση.

Τέλος επιστρέφουμε τον γραμμικό συνδυασμό των χαρακτηριστικών όπου το κάθε χαρακτηριστικό έχει το κατάλληλο βάρος.

```
score = score
score += 6.0 / closestfood
score += - 4.5  * len(currentGameState.getCapsules())
score += - 10 * len(food_list)
score += - 3.5 * closestghost
score += 8.0 / closestcapsule
```

Επιστρέφουμε λοιπόν το score προσθέτοντας τους κατάλληλους συνδυασμούς των χαρακτηριστικών και των βαρών τους ανάλογα με το πόσο σημαντικά είναι.

Το μεγαλύτερο βάρος το έχει η λίστα που μας δείχνει πόσα food έχουν απομείνει για μία κατάσταση νίκης.

Επομένως πρέπει να έχει μεγαλύτερη επιρροή στην επιστρεφόμενη τιμή και άρα στην απόφαση του pacman.

*(Οι τιμές από τα βάρη του κάθε χαρακτηριστικού βρέθηκαν πειραματικά).