```
##############################################
##    Λεπτομέρειες Υλοποίησης Project 2:     ##
##         Κανελλόπουλος Στέφανος            ##
##               sdi1200050                  ##
##############################################

**************************************
* Question 1 (4 points): Reflex Agent *
**************************************

Η λογική στην οποία βασίστηκα ήταν η εξής:
Προσπάθησα για κάθε νέα θέση που θα έχει ο Pacman να βρίσκω αφενός το
κοντινότερο φαγητάκι,καθώς επίσης να
αποφεύγω τα φαντάσματα.Όταν κάποια πιθανή επόμενη θέση φαντάσματος
ταυτίζεται με τη νέα θέση του Pacman,
τότε επιστρέφω μια πολύ μικρή τιμή γι'αυτή την action ώστε να μην
επιλεγεί ποτέ απο την getAction της
ReflexAgent κλάσης. Όταν η νέα θέση του βρίσκεται πάνω σε φαγητάκι τότε
επιστρέφω 0 , αλλιώς η evaluationFunction
θα επιστρέψει την κοντινότερη απόσταση προς κάποιο φαγητάκι.
(Επειδή όμως η καλύτερη κίνηση του Pacman έχει να κάνει με τη "μικρότερη"
απόσταση που πρέπει να διανύσει,
κι επείδη η getAction της ReflexAgent , επιλέγει τη μεγαλύτερη τιμή από
αυτές που επιστρέφει η συνάρτηση
αξιολόγησης μου , επιστρέφω αρνητικές τιμές!)

**********************************
* Question 2 (5 points): Minimax *
**********************************

Ακολούθησα πιστά τη λογική των διαφανειών στην υλοποίηση της MiniMax
συνάρτησης φτιάχνωντας τα εξής:
-miniMax Function ( η οποία αντιπροσωπεύει την πρώτη κίνηση ενός max
player ( Pacman στην προκειμένη ))
-maxValue Function ( αφορά τις κινήσεις του max player )
-minValue Function ( αφορά τις κινήσεις των min player )
Καθώς επίσης και τις συναρτήσεις:
-Result : επιστρέφει για ενα δοσμένο state για κάποιο συγκεκριμένο agent
, αφού κάνει κάποιο action..ποιές θα
είναι οι διάδοχες καταστάσεις του.
-TerminalTest : ελέγχει το πότε πρέπει να σταματήσει η αναδρομή ( είτε
όταν κερδίζει , χάνει , ή φτάνει το
επιθυμητό βάθος )
-Utility : αξιολογεί την τερματική κατάσταση
$$ Ακόμα περισσότερες λεπτομέρειες βρίσκονται σε σχόλια μέσα στον κώδικα
για τον τρόπο που αυτές λειτουργούν! $$

******************************************
* Question 3 (5 points): Alpha-Beta Pruning *
******************************************

Ακολούθησα και εδώ πιστά τη λογική των διαφανειών και κατά την υλοιποίηση
μου έκανα τα εξής:
Πήρα τις maxValue & minValue από τη miniMax συνάρτηση που είχα φτιάξει,
και πρόσθεσα τη συνάρτηση value η οποία
παρέμβαλεται ανάμεσα στις 2 προαναφερθήσες συναρτήσεις , και καθορίζει
ουστιαστικά τη διαδικασία του κλαδέματος.
Τέλος θα τρέξω ουσιαστικά για όλες τις δυνατές κινήσεις του pacman
καλώντας τη value κ έπειτα θα ελέγχω για την
καλύτερη δυνατή κίνηση την οποία και θα επιστρέψω!
```

$$ Ακόμα περισσότερες λεπτομέρειες βρίσκονται σε σχόλια μέσα στον κώδικα για τον τρόπο που αυτές λειτουργούν! $$

****
* Question 4 (5 points): Expectimax *
****

Για την υλοποίηση της Expectimax , χρειάστηκα σαν κορμό την υλοποιήση της miniMax που είχα απο πριν και το μονο που
τροποποίησα για να δουλεύει ο κώδικας ως "Expectimax" ήταν στο κομμάτι της minValue (και πλεον για αυτό το ερώτημα
chancePlayer) να πάψω να κρατώ τη μικρότερη τιμή από αυτές που έπαιρνα γιατί πολύ απλά η Expectimax (στο σημείο
που παίζει ο Chance Player) θέλει να κρατήσει όλα τα values και στο τέλος να τ διαιρέσει με τον παράγοντα τύχης και
αυτή νάναι η τιμή που θα επιστρέψει στην Expectimax (η οποία με τη σειρά της θα επιλέξει την καλύτερη κίνηση με βάση
τα values που έχει στη διάθεση της).
$$ Ακόμα περισσότερες λεπτομέρειες βρίσκονται σε σχόλια μέσα στον κώδικα για τον τρόπο που αυτές λειτουργούν! $$

****
* Question 5 (6 points): Evaluation Function *
****

Για την υλοποίηση του τελευταίου ερωτήματος η αλήθεια είναι πως τελικά χρειάστηκα απροσδόκητα λίγα πράγματα.
Καταρχάς , σαν βάση του τελικού σκορ μου , χρειάστηκα το σκορ του currentGameState και με βάση αυτό, κάθε φορά προσέδιδα
ή αφαιρούσα αξία ανάλογα με κάποιους παράγοντες που έλαβα υπόψην.
Ενώ αρχικά πέτυχα άριστα αποτελέσματα (6/6) μέσω ενός γραμμικού συνδιασμού των :

- (πρόσθεση) min αποστάσεων μεταξύ pacman - φαντασμάτων (θέλοντας να κάνω τον pacman να απομακρύνεται όταν βρισκόταν
κόντα σε κάποιο φάντασμα)
- (αφαίρεση) min αποστάσεων μεταξύ pacman - εναπομείναντων φαγητών (θέλοντας να παροτρύνω τον pacman να πηγαίνει προς
αυτά)
- (αφαίρεση) μήκος εναπομείναντων φαγητών (ώστε οι κινήσεις που θα συμπερίλαμβανουν φάγωμα φαγητού να είναι προτιμητέες
σε σχέση μαυτές που δεν τρώνε)
- (αφαίρεση) μήκος εναπομείναντων καψουλών (ώστε όταν βρίσκεται κοντά σε κάψουλα να τον παρακινήσω να πάει προς τα εκεί
με το σκεπτικό ότι τρώγοντας την κάψουλα (θα κυνηγήσει να φάει φαντάσματα) και κατά συνέπεια θα πάρω bonus στο τελικό σκορ.!!

... (και φυσικά όλα τα παραπάνω με κατάλληλους συντελεστές)

Παρατήρησα μετά από λίγα πειράματα (και ειδικά ως προς το τι βάρος έπρεπε να δώσω στο κάθετι),
ότι μπορούσα να πετύχω πολύ καλύτερα σκορς αφαιρώντας από το current σκορ μόνο τη min απόσταση φαγητού και το
μήκος εναπομείναντων καψουλών.
Γι'αυτό και το παραδοτέο αρχείο μου περιέχει αυτό τον κώδικα και απλά άφησα σε σχόλια οτι αφορούσε τα φαντάσματα ,γι'αυτό το
ερώτημα ,μιας και δε μου χρείαστηκε!

##########
Τέλος Pdf