Ονοματεπώνυμο : Δημήτριος Φούντας
A.M:1112201600236

ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

(PROJECT1)

Στο πρώτο θέμα στον dfs αλγόριθμο χρησιμοποιώ μια στοίβα από το πακέτο util μέσα στην οποία αρχικά βάζω μια λίστα με πλειάδες που περιλαμβάνει την πλειάδα με την αρχική θέση, τις κινήσεις που έχω κανει ως τώρα και το κόστος τους. Πλέον η στοιβα αυτή είναι το σύνορο μου.αρχικοποιώ επίσης ένα σετ το οποίο θα κρατάει τις καταστάσεις που έχω ήδη εξερευνήσει. Μέχρι το σύνορο μου να αδείασει βγάζω ένα από τα στοιχεία της και για να βρω σε ποιά θέση βρίσκομαι στο προηγούμενο βήμα παίρνω το πρώτο στοιχείο της τελευταίας πλειάδας της λίστας που έχω εξάγει από την στοίβα. Αν αυτή η θέση είναι κατάσταση στόχου, τότε γυρνάω τις μέχρι τώρα κινήσεις που έχω πραγματοποιήσει για να φτάσω ως εκεί . Σε περίπτωση που η κατάσταση που βρισκόμαστε δεν είναι κατάσταση στόχου, τότε θα εξερευνήσω τα παιδία του και θα τον προσθέσω στο σετ μου ως εξερευνημένο. Αν τα παιδιά του δεν έχουν ήδη εξερευνηθεί ,τότε φτιάχνω μια λίστα που μου έχει τις προηγούμενες πλειάδες και άλλη μία η οποία αποθηκεύει τις πληροφορίες του παιδιού(την κατάστασή του,τις ενέργειες που κάναμε για να φτάσουμε, και το συνολικο κόστος που κάναμε για να φτάσουμε εκεί). Συνεχίζω αυτη την διαδικασία μέχρι να βρω κατάσταση στόχου και να επιστρέψω τις κινήσεις μου ή μεχρι ο αλγόριθμος μου να αποτύχει και να γυρισει False(αν αδειάσει το σύνορο πριν βρω στόχο).Στον αλγόριθμο bfs κάνω ακριβώς την ίδια διαδικασία με την προηγούμενη απλά σαν σύνορο τώρα έχω ουρά. Εξαιρετικά όμοια λογική ακολουθώ και στην ucs αναζήτηση με την διαφορά ότι χρησιμοποιώ ουρά προτεραιώτητας ως σύνορο με προτεραιώτητα το συνολικό κόστος του κάθε μονοπατιού που φτιάχνω.Τέλος ο αλγόριθμος A* ειναι όμοιος με τον ucs , με την διαφορά πως στο σύνορό μου η προτεραιότητα είναι πλέον το συνολικό κόστος που έχω κανει για να φτάσω στην συγκεκριμένη κατάσταση + την ευριστική του συνάρτηση

Στην προσπάθεια μου να κάνω τον pac man να επισκεφτεί όλες τις γωνίες χρησιμοποίησα μια πλειάδα στην οποία κάθε φορά πρόσθετα τις γωνίες που έχω ηδη επισκεπτεί.Όταν αυτή θα περιέχει όλες τις γωνίες, τότε θα έχει τελειώσει το πρόβλημα μας. Δημιούρησα επίσης για τον σκοπό μου μια μέθοδο η οποία βρίσκει την μετάβαση του pac man από μια κατάσταση σε μια άλλη. Ελέγχει κάθε φορά αν η κίνηση μας είναι νόμιμη ή αν χτυπάει σε τοίχο και κάθε φορά ελέγχο ο pac man να μην κατευθύνεται σε γωνία την οποία έχει ήδη επισκεπτεί.

Στην ερώτηση 7 ξεκινάω βρίσκοντας ποιες γωνίες δεν έχω ήδη επισκευτεί . Από την θέση που ξεκινάω μετράω την ευκλείδια απόσταση της από την πιο κοντινη γωνία, κατευθύνομαι προς τα εκεί και αποθηκεύω την ευκλείδια απόσταση σε μια μεταβλητή συνολικού κόστους. Τώρα βρίσκομαι σε μιά απο τις γωνίες,κάνω πάλι την ίδια διαδικασία και κατευθύνομαι προς την επόμενη γωνία και προσθετω την ευκλειδια απόσταση των δύο γωνιών στην μεταβλητή συνολικού κόστους. Συνεχίζω την ίδια διαδικασία μέχρι να επισκεπτώ ολες τις γωνίες.Η συγκεκριμένη ευρετικη είναι συνεπής γιατι ο pac man πρέπει τουλάχιστον να διανύσει αυτή την απόσταση για να φτάσει στον στόχο , όμως αυτό δεν θα γίνει ποτέ καθώς πρώτον δεν μπορώ να μετακινηθώ διαγώνια,οπότε η ελάχιστη απόσταση συχνά απο ένα σημείο σε ένα άλλο θα ειναι η μανχάταν και επιπροσθέτως θα έχει εμπόδια η διαδρομή , κάτι που θα αναγκάσει τον pac man να κάνει επιπλέον κινήσεις.

Η ευρεστική συνάρτηση που χρησιμοποιώ για να βρω γρήγορα και αποτελεσματικά όλες τις τροφές χρησιμοποιώ όμοια ευρετική με την προηγούμενη η οποία υπολογίζει την μικροτερη ευκλειδια απόσταση απο την θέση μας στην κοντινότερη τροφή.Και μετά απο την κατάσταση της τροφής ,στην επόμενη πιο κοντινή τροφή , μέχρι να μην υπάρχει τροφή που ο πακ μαν δεν εχει φάει.

Στην τελευταία ερώτηση προσπαθώ να επισκεφτώ μια μεμονομέμη τροφή με άπληστο τρόπο. Για τον σκοπό αυτο χρησιμοποιώ τον άπλειστο αλγόριθμο uniform cost search. Ο αλγόριθμος μου θα ψάξει το μονοπάτι μέσο του συγκεκριμενου αλγορίθμου και σταματάει όταν η κατάσταση που βρισκόμαστε είναι κατάσταση στόχου.