Project 1 - Pacman - Κωνσταντίνα Έλληνα 1115201600046

Έχω βάλει σχόλια και στον κώδικα για καλύτερη κατανόηση του αλγορίθμου.

Question 1:
Στην αναζήτηση κατά βάθος χρησιμοποιώ στοίβα για να αποθηκεύονται τα παιδιά του κόμβου. Χρησιμοποιώ dictionary για να αποθηκεύω το μονοπάτι που φτιάχνεται κάθε φορά που γίνεται dfs. Ύστερα, υλοποίησα τον ψευδοκώδικα που υπάρχει στις διαφάνειες του μαθήματος.

Question 2:
Στην αναζήτηση κατά πλάτος χρησιμοποιώ ουρά για την αποθήκευση των παιδιών των κόμβων. Δεν αλλάζει κάτι άλλο, είναι πάλι η υλοποίηση του ψευδοκώδικα, αλλά με ουρά.

Question 3:
Σε αυτό το ερώτημα χρησιμοποιώ ουρά προτεραιότητας και τον αλγόριθμο του BFS. Το μόνο που αλλάζει είναι ότι βρίσκω και το κόστος που ζητείται.

Question 4:
Είναι ίδια λογική με το προηγούμενο ερώτημα, χρησιμοποιώ τον ίδιο αλγόριθμο αναζήτησης και απλά προσθέτω το heuristic στο κόστος κάθε επόμενης κίνησης.

Question 5:
Συμπληρώνω στο getStartState τη θέση που ξεκινάω και μία λίστα που μας λέει αν οι γωνίες έχουν εξερευνηθεί. Στο isGoalState επιστρέφουμε αν εξερευνήθηκαν οι γωνίες μας. Στο getSuccessors βρίσκουμε την κίνησή μας, ενημερώνουμε αν βρούμε γωνία και επιστρέφουμε το successors που περιέχει τη θέση μας, αν βρέθηκε γωνία, το action της for και το κόστος.

Question 6:
Για να είναι γρήγορος ο αλγόριθμος ορίζουμε την απόσταση manhattan με lambda συνάρτηση. Έχουμε τις ανεξερεύνητες γωνίες, υπολογίζουμε τις αποστάσεις της θέσης μας με τις γωνίες και επιστρέφουμε τη μεγαλύτερη από αυτές.

Question 7:
Είναι παρόμοιο με το προηγούμενο ερώτημα, αλλά εδώ υπολογίζουμε σε τετραγωνάκια λαβυρίνθου τις αποστάσεις, διότι δεν πρέπει να κάνει περιττές κινήσεις, πρέπει να κινείται μέχρι

εκεί που είναι το φαγητό.

Question 8:
Επιστρέφουμε το ucs, επιστρέφουμε αν στη θέση που είμαστε υπάρχει φαγητό(isGoalState).