Οικονομικό Πανεπιστήμιο Αθηνών, Τμήμα Πληροφορικής
Μάθημα: Τεχνητή Νοημοσύνη
Ακαδημαϊκό έτος: 2022–23
Διδάσκων: Ι. Ανδρουτσόπουλος

1η Εργασία

Αναπτύξτε πρόγραμμα σε Java, C++, Python (ή άλλη γλώσσα που θα σας επιτρέψει ο βοηθός του φροντιστηρίου σας) που να λύνει **ένα από τα ακόλουθα προβλήματα** (πρέπει να επιλέξετε ακριβώς ένα πρόβλημα).

α) **Κύβος του Rubik**.[1] Το πρόγραμμά σας θα πρέπει να ψάχνει ακολουθία κινήσεων (περιστροφών) που να οδηγεί στην τελική κατάσταση, ξεκινώντας κάθε φορά από τυχαία αρχική κατάσταση. Θα πρέπει να χρησιμοποιήσετε τον τον αλγόριθμο A* (με ή χωρίς κλειστό σύνολο) και ευρετικές που θα περιγράψετε στο έγγραφο που θα παραδώσετε. Για να γίνει ευκολότερο το πρόβλημα, κατά την κλήση του προγράμματος θα ορίζεται ως παράμετρος, έστω K, ο αριθμός των πλευρών του κύβου (1 ως 6) που θα πρέπει να έχουν ενιαίο χρώμα για να θεωρήσουμε ότι βρισκόμαστε σε τελική κατάσταση. Ενδεικτικά, για K = 2 αν καταφέρουμε δύο πλευρές να έχουν ενιαίο χρώμα (π.χ. η μια πλευρά να είναι όλη πράσινη, η άλλη όλη κόκκινη), τότε βρισκόμαστε σε τελική κατάσταση ανεξαρτήτως του τι συμβαίνει στις άλλες τέσσερις πλευρές. Στην πρωτότυπη μορφή του παιχνιδιού, K = 6 (ισοδύναμα K = 5). Δεδομένης της τιμής του K, το πρόγραμμά σας θα πρέπει να προσπαθεί να βρει τη βέλτιστη λύση (το συντομότερο μονοπάτι κινήσεων). Θα πρέπει να συμπεριλάβετε στο έγγραφο που θα παραδώσετε παραδείγματα εκτέλεσης του προγράμματός σας με ενδεικτικές αρχικές καταστάσεις και τις λύσεις που βρήκε (αν βρήκε) το πρόγραμμα, για ενδεικτικές τιμές του K (π.χ. 1, 2, 3).

γ) **Reversi** (ή την παραλλαγή του Othello).[2] Το πρόγραμμά σας θα πρέπει να επιτρέπει στον χρήστη να παίξει Reversi (ή Othello) με αντίπαλο τον υπολογιστή. Για την επιλογή των κινήσεων του υπολογιστή, το πρόγραμμά σας πρέπει να χρησιμοποιεί τον αλγόριθμο MiniMax, κατά προτίμηση με πριόνισμα α-β. Κατά την έναρξη του παιχνιδιού, ο χρήστης θα πρέπει να μπορεί να επιλέξει το μέγιστο βάθος αναζήτησης του αλγορίθμου MiniMax. Ο χρήστης θα πρέπει να μπορεί να επιλέξει, επίσης, αν θα παίξει πρώτος ή όχι. Το πρόγραμμα πρέπει να απορρίπτει κινήσεις που παραβιάζουν τους κανόνες του παιχνιδιού. Αν ο παίκτης του οποίου είναι η σειρά να παίξει δεν μπορεί να τοποθετήσει πουθενά νέο πούλι χωρίς να παραβιάσει τους κανόνες, το πρόγραμμα πρέπει να εμφανίζει αυτόματα σχετικό μήνυμα και να ζητά να παίξει ο άλλος παίκτης. Μετά από κάθε κίνηση, το πρόγραμμα θα πρέπει να δείχνει την κατάσταση του παιχνιδιού (π.χ. τυπώνοντας κενά, X και O). Δεν μας ενδιαφέρει σε αυτό το μάθημα η διεπαφή χρήστη (π.χ. μην αφιερώσετε χρόνο στην ανάπτυξη γραφικής διεπαφής). Εναλλακτικά μπορείτε να ασχοληθείτε με κάποιο άλλο παιχνίδι δύο αντιπάλων με ή χωρίς ζάρια (π.χ. τάβλι, σκάκι, όχι όμως π.χ. απλή τρίλιζα) που θα σας επιτρέψει ο βοηθός

[1] Βλ. https://en.wikipedia.org/wiki/Rubik%27s_Cube.
[2] Μπορείτε να βρείτε τους κανόνες του παιχνιδιού στη διεύθυνση: http://en.wikipedia.org/wiki/Reversi.

του φροντιστηρίου σας (και το οποίο θα περιγράφετε στο έγγραφο που θα παραδώσετε). Θα πρέπει να περιλάβετε στο έγγραφο που θα παραδώσετε και ενδεικτικά παραδείγματα παρτίδων που έπαιξε το πρόγραμμά σας με αντίπαλο άνθρωπο ή προαιρετικά και με προγράμματα άλλων ομάδων.

Η προθεσμία παράδοσης της εργασίας θα ανακοινωθεί στο e-class. **Διαβάστε προσεκτικά και το έγγραφο με τις γενικές οδηγίες των εργασιών του μαθήματος** (βλ. e-class).