Δομές Δεδομένων

Άσκηση

Διδάσκων **Δημήτρης Μιχαήλ**
Ακ. Έτος **2021-2022**

Οδηγίες Παράδοσης

1. Η άσκηση μπορεί να γίνει σε ομάδες 2 ή 3 φοιτητών.
2. Η παράδοση της άσκησης πρέπει να γίνει ηλεκτρονικά μέσω email.
3. Θα πρέπει να παραδώσετε ένα zip αρχείο που πρέπει να περιέχει
   (a) ένα φάκελο **src** με τον πηγαίο κώδικα της άσκησης
   (b) ένα .pdf αρχείο με την αναφορά.

   Το αρχείο πρέπει να περιέχει μόνο τον πηγαίο κώδικα και όχι και τα εκτελέσιμα αρχεία.
4. Η αναφορά πρέπει να περιέχει εισαγωγή στο θέμα, λεπτομερή ανάλυση της λύσης που υλοποιήσατε μαζί με τον κώδικα που γράψατε καθώς και παραδείγματα εκτέλεσης του.
5. Σε περίπτωση αντιγραφής θα μηδενίζονται **όλες** οι εμπλεκόμενες ασκήσεις.

1 Άσκηση

Στόχος της άσκησης αυτής είναι η εξοικείωση σας με α) τα γραφήματα ως δομή δεδομένων και β) την συγγραφή πολύ απλών αλγορίθμων επάνω σε γραφήματα. Καλείστε να υλοποιήσετε ένα πρόγραμμα που να αναπαριστά ένα λαβύρινθο ως γράφημα και να κατασκευάζει διαδρομές διαφυγής από τον λαβύρινθο.

Για απλότητα θα χρησιμοποιήσουμε πίνακες και χαρακτήρες ASCII για να κωδικοποιήσουμε έναν λαβύρινθο, όπως φαίνεται στο παρακάτω παράδειγμα:

```
labyrinth=[
  '########',
  '#X#    #',
  '# # ## #',
  '#   ## #',
  '# ##   #',
  '# ## ###',
  '#  #    ',
  '########'
]
```

Στο παράδειγμα αυτό βλέπουμε ένα πίνακα 8×8 που αναπαριστά έναν λαβύρινθο. Το σύμβολο `#` αναπαριστά τοίχο, το κενό αναπαριστά ελεύθερο χώρο ενώ το σύμβολο `X` αναπαριστά μία τοποθεσία ενδιαφέροντος.

1.1 Αναπαράσταση ως Γράφημα

Στο πρώτο μέρος της άσκησης θα πρέπει να υλοποιήσετε την δομή μη-κατευθυνόμενο γράφημα με την χρήση λιστών γειτνίασης (adjacency lists). Για την άσκηση αυτή το γράφημα δεν θα αλλάζει παρά μόνο κατά την διάρκεια φόρτωσης του στην μνήμη. Για απλότητα μπορείτε να χρησιμοποιήσετε πίνακες κατακερματισμού (dict) αντί για πίνακες για την αποθήκευση των λιστών.

Χρησιμοποιήστε μία κλάση για να αναπαραστήσετε το γράφημα. Ως κόμβους μπορείτε να χρησιμοποιήσετε ακέραιους ή οτιδήποτε άλλο αντικείμενο βολεύει. Για παράδειγμα μπορεί να βολεύει να χρησιμοποιήσετε κάποιο αντικείμενο που αναπαριστά ένα κελί του λαβύρινθου με τις συντεταγμένες του. Η κλάση αυτή πρέπει να έχει το παρακάτω interface:

- `add_vertex(v)` προσθέτει έναν κόμβο στο γράφημα αν δεν υπάρχει ήδη
- `add_edge(u,v)` προσθέτει μία ακμή στο γράφημα αν δεν υπάρχει ήδη
- `contains_vertex(v)` απαντά true ή false ανάλογα με το αν ο κόμβος v ανήκει στο γράφημα
- `contains_edge(u,v)` απαντά true ή false ανάλογα με το αν η ακμή $\{u, v\}$ ανήκει στο γράφημα
- `num_vertices()` επιστρέφει τον αριθμό των κόμβων
- `num_edges()` επιστρέφει τον αριθμό των ακμών
- `vertex_set()` επιστρέφει ένα σύνολο με όλους τους κόμβους του γραφήματος. Μία άλλη πιθανή υλοποίηση είναι να επιστρέφετε έναν iterator και να ονομάσετε την συνάρτηση `vertex_it()`.
- `neighbors(v)` επιστρέφει ένα σύνολο με όλους τους γείτονες του κόμβου v Μία άλλη πιθανή υλοποίηση είναι να επιστρέφετε έναν iterator και να ονομάσετε την συνάρτηση `neighbors_it()`.
- `degree(v)` επιστρέφει τον βαθμό του κόμβου v (αριθμό γειτόνων)

1.1.1 Φόρτωση

Η είσοδος σας θα είναι ένα αρχείο κειμένου που θα περιέχει τον λαβύρινθο σε CSV μορφή όπως φαίνεται παρακάτω. Αν δεν υπάρχει τίποτα μεταξύ δύο διαχωριστών (κόμμα) τότε είναι το ίδιο με το να υπάρχει κενός χώρος.

```
#,#,#,#,#,#,#,#
#,X,#, , , , ,#
#, ,#, ,#,#, ,#
#, , , ,#,#, ,#
#, ,#,#, , , ,#
```

```
#, ,#,#, ,#,#,#
#, , ,#, , , ,
#,#,#,#,#,#,#,#
```

θα πρέπει να διαβάσετε το αρχείο αυτό και να κατασκευάσετε το γράφημα. Μπορείτε είτε να διαβάσετε πρώτα σε πίνακα και μετά να κατασκευάσετε το γράφημα ή να κατασκευάσετε το γράφημα απευθείας. Για να φτιάξετε το γράφημα πρέπει να σκεφτείτε μία αντιστοίχιση κελιών του λαβυρίνθου σε κόμβους. Ακμές στο γράφημα υπάρχουν όταν κάποιος μπορεί να μετακινηθεί από ένα κελί προς διπλανό κελί.

1.2 Διαφυγή

Για τους σκοπούς της εργασίας θέλουμε να δούμε αν υπάρχει διαδρομή διαφυγής από τον λαβύρινθο ξεκινώντας από το κελί με το σύμβολο `X`. Διαφυγή θα πει να μπορούμε να κατασκευάσουμε ένα μονοπάτι στον λαβύρινθο που να βγαίνει στον εξωτερικό χώρο. Για σωστή μοντελοποίηση θα πρέπει να προσθέσετε στον γράφημα σας και έναν κόμβο που να αναπαριστά τον εξωτερικό χώρο.

Για να απαντήσετε το ερώτημα αυτό καλείστε να υλοποιήσετε τον αλγόριθμο Depth First Search (DFS). Γράψτε μία συνάρτηση η οποία θα πέρνει ως είσοδο ένα γράφημα G και δύο κόμβους s και t και θα επιστρέφει ένα μονοπάτι $s - t$ αν υπάρχει ή `None` σε περίπτωση που δεν υπάρχει μονοπάτι από τον κόμβο s στον κόμβο t. Η συνάρτηση αυτή θα πρέπει να κάνει διάσχιση κατά βάθος και να σταματάει την εκτέλεση της είτε αν βρει τον κόμβο t είτε αν εξερευνήσει όλο το γράφημα και δεν καταφέρει να βρει τον κόμβο t. Ως αποτέλεσμα πρέπει να επιστρέφει μία λίστα με κόμβους όπου πρώτος πρέπει να είναι το s και τελευταίος το t.

Αφού υλοποιήσετε την συνάρτηση αυτή, χρησιποιήστε την για να απαντήσετε το ερώτημα διαφυγής στον λαβύρινθο καθώς και για να τυπώσετε το μονοπάτι διαφυγής σε συντεταγμένες. Μπορείτε να θεωρήσετε πως η πάνω αριστερά γωνία του λαβύρινθου είναι το `(0,0)`.

1.3 Τεχνολογίες

Για την υλοποίηση σας θα πρέπει να χρησιμοποιήσετε την γλώσσα προγραμματισμού Python. Προσοχή πως **δεν** επιτρέπεται η χρήση κάποιας έτοιμης βιβλιοθήκης γραφημάτων, θα πρέπει να υλοποιήσετε μόνοι σας την δομή δεδομένων γράφημα. Επίσης η Python θα πρέπει να είναι τουλάχιστον έκδοση 3.6 και σε περίπτωση χρήσης κάποιας εξωτερικής βιβλιοθήκης θα πρέπει να παρέχετε μαζί και ένα αρχείο requirements.txt όπου να περιέχει όλα τα dependencies που χρησιμοποιήσατε. Κοιτάξτε το εργαλείο pip και πιο συγκεκριμένα την εντολή `pip` και πιο συγκεκριμένα την εντολή `pip freeze`. Για το διάβασμα των παραμέτρων από την γραμμή εντολών όπως π.χ το αρχείο εισόδου, η Python παρέχει την βιβλιοθήκη `argparse`.

Βαθμολογία

Κριτήρια

- Καλή μοντελοποίηση.
- Σωστή ονοματολογία μεταβλητών, συναρτήσεων, κλάσεων.
- Σωστή λειτουργικότητα.
- Αποδοτική υλοποίηση.
- Εύκολη μεταγλώττιση και εκτέλεση.
- Ολοκληρωμένη και σωστή τεκμηρίωση και περιγραφή στην αναφορά.

Εκτός από τα αρχεία με τον κώδικα πρέπει να γράψετε και μια αναφορά. Η αναφορά πρέπει να εξηγεί τις διάφορες επιλογές που κάνατε, γιατί μοντελοποιήσατε έτσι το πρόβλημα καθώς και να σχολιάζει τον κώδικα σας. Η αναφορά πρέπει να είναι υποχρεωτικά σε μορφή *pdf*. Επίσης φροντίστε ο κώδικας σας να περιέχει και ένα README αρχείο που να εξηγεί με ακρίβεια πως κάνει κάποιος compile και πως το εκτελεί. Ακρίβεια εδώ σημαίνει πως απλά κάποιος κάνει copy-paste μία εντολή και την εκτελεί χωρίς να είναι απαραίτητη κάποια αλλαγή. Για την εκτέλεση με τα αρχεία ως είσοδο μπορείτε να υποθέσετε πως τα αρχεία θα βρίσκονται στον ίδιο φάκελο που βρίσκονται και τα `.py` αρχεία σας.