HW5 - OpenStreetMap

Ημερομηνία παράδοσης: Παρασκευή 4 Ιουλίου 2025

Σε αυτή την εργασία θα γράψετε ένα πρόγραμμα που υλοποιεί ένα γράφο, με τη βοήθεια πραγματικών δεδομένων που θα λάβετε από αρχεία της πλατφόρμας **OpenStreetMap**. Για την υλοποίηση του γράφου και όλων των λειτουργιών του θα πρέπει να χρησιμοποιήσετε έτοιμες κλάσεις από τη βιβλιοθήκη STL και όχι δικές σας υλοποιήσεις.

Ο γράφος που θα πρέπει να είναι υλοποιημένος ως λίστα γειτνιάσεως και ΟΧΙ ως πίνακας γειτνιάσεως. Για τον σκοπό αυτό απαιτείται να έχετε τα εξής:

- Μία κλάση με όνομα **`Vertex`** που περιγράφει τον κόμβο του γράφου. Ο κόμβος έχει ένα μοναδικό ακέραιο αναγνωριστικό τύπου **`unsigned long`**, και δύο **`double`** τιμές για το γεωγραφικό πλάτος (latitude) και γεωγραφικό ύψος (longitude). Επιπλέον, περιέχει μία λίστα **`std::list`** με ακμές όπως αυτές περιγράφονται από την κλάση **`Edge`**. Κάθε ακμή του κόμβου έχει ως κόμβο αφετηρίας τον κόμβο αυτό. Μπορείτε να προσθέσετε επιπλέον πεδία αν το κρίνετε απαραίτητο. Η κλάση του κόμβου κατ' ελάχιστον θα πρέπει να έχει τα εξής:
  - Ένα κατασκευαστή που λαμβάνει ως παραμέτρους το αναγνωριστικού του κόμβου και το γεωγραφικό μήκος και πλάτος.
  - Ένα copy-constructor με παράμετρο τύπου **`const Vertex&`**.
  - Συναρτήσεις για προσθήκη ακμής και αφαίρεση ακμής
  - Συναρτήσεις για διάβασμα της τιμής του ακέραιου αναγνωριστικού του κόμβου και των τιμών γεωγραφικού πλάτους και γεωγραφικού ύψους.
- Μία κλάση με όνομα **`Edge`** που περιγράφει τις ακμές του γράφου. Η ακμή έχει το αναγνωριστικό του κόμβου αφετηρίας (προαιρετικό πεδίο), το αναγνωριστικό του κόμβου προορισμού και μία τιμή **`double`** που περιγράφει την απόσταση ανάμεσα τους.
  - Ένα κατασκευαστή που λαμβάνει το αναγνωριστικό του κόμβου αφετηρίας, το αναγνωριστικό του κόμβου προορισμού και την απόσταση μεταξύ τους.
  - Ένα copy-constructor με παράμετρο τύπου **`const Edge&`**.
  - Συναρτήσεις για ανάγνωση και εγγραφή των αναγνωριστικών των κόμβων και της απόστασης.
- Η κλάση **`Graph`** του γράφου, περιγράφει τον δίκτυο των κόμβων και των ακμών που προσδιορίζουν το πολεοδομικό συγκρότημα στο οποίο αναφέρεται. Έχετε υπόψη ότι ο γράφος είναι κατευθυνόμενος. Περιέχει ένα **`std::vector`** από κόμβους. Για τη γρηγορότερη αναζήτηση των κόμβων εντός του **`std::vector`**, περιέχει και ένα **`std::unordered_map<unsigned long, unsigned int>`** όπου ως κλειδί καταχωρείται το ακέραιο αναγνωριστικό κάθε κόμβου και ως τιμή η θέση του κόμβου αυτού στο **`std::vector`**. Μπορείτε να προσθέσετε επιπλέον πεδία αν το κρίνετε απαραίτητο. Η κλάση του γράφου κατ' ελάχιστον θα πρέπει να έχει
  - Ένα κατασκευαστή που λαμβάνει το όνομα ενός αρχείο τύπου OSM (OpenStreetMap), με περιεχόμενα σε μορφή XML και δημιουργεί από αυτό τον γράφο. Αφού δημιουργηθεί ο γράφος, αφαιρούνται όλοι οι κόμβοι που είναι ασύνδετοι.
  - Μία συνάρτηση προσθήκης κόμβου στον γράφο. Κατά την προσθήκη κόμβου θα πρέπει να δίνεται το μοναδικό αναγνωριστικό του κόμβου και το γεωγραφικό πλάτος και ύψος.
  - Μία συνάρτηση διαγραφής κόμβου από τον γράφο, με παράμετρο το μοναδικό να αναγνωριστικό του κόμβου. Πριν από τη διαγραφή του κόμβου θα πρέπει να διαγραφούν και όλες οι ακμές που έχουν ως αφετηρία ή προορισμό τον κόμβο αυτό.
  - Μία συνάρτηση εύρεσης του συντομότερου μονοπατιού μεταξύ δύο σημείων στον γράφο, με τη βοήθεια του αλγορίθμου **dijkstra**. Η συνάρτηση λαμβάνει ως ορίσματα τα μοναδικά

αναγνωριστικά των δύο κόμβων (`start, end`) και επιστρέφει ένα `std::list` με το μονοπάτι από τον κόμβο αφετηρίας προς τον κόμβο προορισμού, συμπεριλαμβανομένων των κόμβων αυτών. Για την επιλογή του συντομότερου μονοπατιού, όταν συναντούμε δύο ή περισσότερους κόμβους που έχουν την ίδια απόσταση από το σημείο έναρξης της διαδρομής, επιλέγουμε τον κόμβο με το μικρότερο ακέραιο αναγνωριστικό.

- Μία συνάρτηση σύμπτυξης του γράφου. Εδώ θα πρέπει να γίνει η διάκριση μεταξύ ακμών που αναφέρονται σε δρόμους μίας κατεύθυνσης και σε δρόμους δύο κατευθύνσεων.
  - **Για δρόμους μίας κατεύθυνσης**, εάν ***α)*** ο κόμβος αποτελεί κόμβο προορισμού μιας ακμής με αφετηρία έναν άλλο κόμβο και κόμβο αφετηρίας μιας άλλης ακμής με ένα τρίτο κόμβο και ***β)*** δεν υπάρχουν άλλες ακμές από ή προς τον κόμβο αυτό, ***τότε*** ο ενδιάμεσος κόμβος πρέπει να διαγραφεί και οι δύο ακριανοί κόμβοι να συνδεθούν με μία νέα ακμή. Η απόσταση της ακμής αυτής θα πρέπει να επαν-υπολογιστεί.
  - **Για δρόμους δύο κατευθύνσεων**, εάν ο ***α)*** ο κόμβος αποτελεί κόμβο προορισμού δύο ακμών με αφετηρία δύο άλλους κόμβους και κόμβο αφετηρίας δύο άλλων ακμών με τους ίδιους κόμβους και ***β)*** δεν υπάρχουν άλλες ακμές από ή προς τον κόμβο αυτό, τότε ο ενδιάμεσος κόμβος πρέπει να διαγραφεί και οι δύο ακριανοί κόμβοι να συνδεθούν με δύο νέες ακμές. Η απόσταση κάθε ακμής πρέπει να επαν-υπολογιστεί.
- Μια συνάρτηση διαπέρασης του γράφου με βάση τον αλγόριθμο **BFS (breadth first search)**. Η συνάρτηση επιστρέφει μία λίστα `std::list<unsigned long>` με τα μοναδικά αναγνωριστικά των κόμβων με τη σειρά που τα διέτρεξε ο αλγόριθμος. Για την υλοποίηση του αλγορίθμου **BFS** απαιτείται η χρήση ενός **FIFO** που θα το υλοποιήσετε μέσω `std::queue`. Η σειρά διάτρεξης των γειτνιαζόντων κόμβων κάθε κόμβου είναι από το κόμβο με το μικρότερο ακέραιο αναγνωριστικό, προς τον κόμβο με το μεγαλύτερο ακέραιο αναγνωριστικό.
- Μια συνάρτηση διαπέρασης του γράφου με βάση τον αλγόριθμο **DFS (depth first search)**. Η συνάρτηση επιστρέφει μία λίστα `std::list<unsigned long>` με τα μοναδικά αναγνωριστικά των κόμβων με τη σειρά που τα διέτρεξε ο αλγόριθμος. Για την υλοποίηση του αλγορίθμου **DFS** απαιτείται η χρήση ενός **LIFO** (χρησιμοποιείστε την έτοιμη κλάση `std::stack`). Η σειρά επίσκεψης των γειτνιαζόντων κόμβων του κάθε κόμβου είναι από το κόμβο με το μικρότερο ακέραιο αναγνωριστικό, προς τον κόμβο με το μεγαλύτερο ακέραιο αναγνωριστικό.

Περιορισμοί: Τα πεδία κάθε κλάσης θα πρέπει να είναι δηλωμένα ως `private`..

Η συνάρτηση main

Η συνάρτηση `main` βρίσκεται στο αρχείο **HW5.cpp**. Η συνάρτηση `main` εκτυπώνει σε επανάληψη το παρακάτω μενού. Εκτυπώνονται επιπλέον χαρακτήρας αλλαγής γραμμής στην αρχή και στο τέλος.

```
-i <filepath>  :  Import Graph from <filepath>
-c             :  Compact Graph
-p <sid> <eid> :  Estimate the shortest path between start
                  node with <sid> and end node with <eid>
-b <sid>       :  Print bfs starting from node with <sid>
-d <sid>       :  Print dfs starting from node with <sid>
-q             :  Exit without memory leaks

Enter your choice:
```

Προσοχή: Στον κώδικα που θα υποβάλλετε στο autolab βάλτε σε σχόλια όλες τις γραμμές του μενού εκτός από την τελευταία, για λόγους ευκρινέστερης σύγκρισης των αρχείων ελέγχου.

Οι επιλογές που μπορεί να εισάγει ο χρήστης έτσι όπως περιγράφονται από το παραπάνω μενού είναι οι εξής:

- **`-i <filepath>`** : Εισάγει ένα νέο γράφο από το μονοπάτι **`<filepath>`**. Αν υπάρχει ήδη κάποιος γράφος αποθηκευμένος, αυτός διαγράφεται και αντικαθίσταται από τον νέο γράφο. Αφού αναγνωστεί ο γράφος αφαιρούνται όλοι οι κόμβοι που δε συνδέονται με άλλους κόμβους.
    - Σε περίπτωση επιτυχίας εκτυπώνεται το μήνυμα **"Graph OK"**, ακολουθούμενο από χαρακτήρα αλλαγής γραμμής.
    - Σε περίπτωση που το αρχείο δεν υπάρχει ή δεν μπορεί να ανοιχθεί εκτυπώνεται χαρακτήρας αλλαγής γραμμής και το μήνυμα **"Unable to open file: X"**, ακολουθούμενο από χαρακτήρα αλλαγής γραμμής, όπου **X** το όνομα του αρχείου που δόθηκε από τον χρήστη.
    - Σε περίπτωση που το αρχείο υπάρχει αλλά δεν περιέχει σωστή XML πληροφορία και δεν μπορεί να διαβαστεί από τη βιβλιοθήκη tinyxml2, τότε εκτυπώνεται χαρακτήρας αλλαγής γραμμής και το μήνυμα **"Invalid format for file: X"**, ακολουθούμενο από χαρακτήρα αλλαγής γραμμής, όπου **X** το όνομα του αρχείου που δόθηκε από τον χρήστη.
- **`-c`** : Συμπιέζει τον γράφο, όπως περιγράφεται από τη συνάρτηση σύμπτυξης του γράφου. Μετά τη συμπίεση εκτυπώνεται το μήνυμα **"Compact OK"**, ακολουθούμενο από χαρακτήρα αλλαγής γραμμής.
- **`-p <sid> <eid>`** : Εκτυπώνει το συντομότερο μονοπάτι από τον κόμβο με αναγνωριστικό **`<sid>`** έως τον κόμβο με αναγνωριστικό **`<eid>`**. Εκτυπώνεται μία λίστα με εγγραφές όπως παρακάτω:

  ```
  [5581419685 -> 5581419686] 77.346
  [5581419686 -> 355243761] 97.129
  [355243761 -> 355481056] 149.188
  ```

  Κάθε εγγραφή περιγράφει μία ακμή στο μονοπάτι. Συγκεκριμένα, περιέχει αριστερή αγκύλη '**`[`**', το μοναδικό αναγνωριστικό του κόμβου αφετηρίας, το σύμβολο **`->`**, το μοναδικό αναγνωριστικό του κόμβου προορισμού, δεξιά αγκύλη '**`]`**', κενό και την απόσταση του κόμβου προορισμού από την αφετηρία εκτυπωμένη με ακρίβεια 3 δεκαδικών ψηφίων, ως το άθροισμα των μηκών των ακμών της διαδρομής έως τον κόμβο αυτό.

  Στη συνέχεια, εκτυπώνονται δύο χαρακτήρες αλλαγής γραμμής και ακολουθεί το URL της διαδρομής στο Google Maps, ώστε να μπορείτε να επιβεβαιώσετε ότι η διαδρομή που προέκυψε είναι σωστή. Η μορφή του URL είναι η εξής:

  `https://www.google.com/maps/dir/lat1,lon1/lat2,lon2/lat3,lon3/…/latN,lonN/`

  όπου **`latX`**, **`lonX`** είναι η τιμές του *latitude* και του *longitude* για κάθε κόμβο της διαδρομής.

- **`-b <sid>`** : Εκτυπώνει τη διαπέραση **BFS** του γράφου ξεκινώντας από τον κόμβο με αναγνωριστικό **`<sid>`**. Εκτυπώνεται το ακέραιο αναγνωριστικό κάθε κόμβου ακολουθούμενο από χαρακτήρα αλλαγής γραμμής, ξεκινώντας από τον κόμβο **`<sid>`**.
- **`-d <sid>`** : Εκτυπώνει τη διαπέραση **DFS** του γράφου ξεκινώντας από τον κόμβο με αναγνωριστικό **`<sid>`**. Εκτυπώνεται το ακέραιο αναγνωριστικό κάθε κόμβου ακολουθούμενο από χαρακτήρα αλλαγής γραμμής, ξεκινώντας από τον κόμβο **`<sid>`**.
- **`-q`** : Τερματίζει αφού απελευθερώσει όλη τη δυναμικά δεσμευμένη μνήμη και κλείσει τα αρχεία που πιθανόν παραμένουν ανοιχτά.

Ανάγνωση του γράφου από αρχείο XML

Για την κατασκευή του γράφου θα πρέπει να χρησιμοποιήσετε πληροφορία από κατάλληλα αρχεία XML που δίνονται από την πλατφόρμα OpenStreetMap. Τα αρχεία αυτά θα τα διαβάσετε με τη βοήθεια της βιβλιοθήκης tinyxml2 για ανάγνωση XML. Τα αρχεία πηγαίου κώδικα της βιβλιοθήκης tinyxml2 θα τα βρείτε εδώ. Παράδειγμα αποσπάσματος ενός τέτοιου αρχείου μπορείτε να βρείτε παρακάτω:

```
<?xml version="1.0" encoding="UTF-8"?>
<osm version="0.6" generator="openstreetmap-cgimap 2.0.1 (3529600
spike-06.openstreetmap.org)" copyright="OpenStreetMap and contributors"
attribution="http://www.openstreetmap.org/copyright"
license="http://opendatacommons.org/licenses/odbl/1-0/">

<node id="355307728" visible="true" version="49" changeset="104599287"
timestamp="2021-05-12T22:00:16Z" user="Henosb" uid="11601547" lat="39.3746568"
lon="22.9470292">
    <tag k="highway" v="traffic_signals"/>
</node>
<node id="9497492531" visible="true" version="1" changeset="117276949"
timestamp="2022-02-11T08:20:33Z" user="konkyriakou" uid="13925068" lat="39.3731283"
lon="22.9424218"/>

<way id="381872739" visible="true" version="7" changeset="136476831"
timestamp="2023-05-24T01:14:43Z" user="nikospag" uid="609559">
<nd ref="355306786"/>
<nd ref="9497492531"/>
<tag k="highway" v="residential"/>
<tag k="name" v="Καραμπατζάκη"/>
<tag k="name:el" v="Καραμπατζάκη"/>
<tag k="name:en" v="Karabatzaki"/>
<tag k="oneway" v="yes"/>
</way>

</osm>
```

Το αρχείο περιέχει κόμβους (`<node>`), όπου κάθε κόμβος έχει ένα μοναδικό αναγνωριστικό `id` και τις γεωγραφικές συντεταγμένες `lat` και `lon`. Οποιαδήποτε άλλη πληροφορία σχετικά με τον κόμβο αγνοείται για τους σκοπούς της παρούσας εργασίας.

Το αρχείο περιέχει μονοπάτια (`<way>`), όπου κάθε μονοπάτι αποτελείται από ένα σύνολο κόμβων. Τα μονοπάτια που μας ενδιαφέρουν (τα υπόλοιπα αγνοούνται) είναι εκείνα που περιέχουν το tag με κλειδί `k="highway"`, διότι περιγράφουν δρόμους. Οι επιτρεπόμενες τιμές που λαμβάνει το κλειδί `highway` είναι `motorway, trunk, primary, secondary, tertiary, residential, living_street, unclassified, service, track`. Σε ένα μονοπάτι, η κατεύθυνση των ακμών ακολουθεί τη σειρά εμφάνισης των κόμβων στο μονοπάτι. Στο προηγούμενο παράδειγμα, ο κόμβος αφετηρίας είναι ο κόμβος με `id 355306786` και ο κόμβος προορισμού αυτός με `id 9497492531`. Εάν το μονοπάτι, περιέχει το κλειδί **oneway** με τιμή `"no"` (στο παράδειγμα μας είναι `"yes"`), τότε ο δρόμος είναι διπλής κατεύθυνσης και δημιουργούμε ακμές και προς τις δύο κατευθύνσεις του μονοπατιού. Η *default* τιμή για το κλειδί **oneway** είναι *false*, που σημαίνει ότι εάν το κλειδί απουσιάζει ο δρόμος είναι διπλής κατεύθυνσης. Η τιμή `"yes"` ή `"1"` σηματοδοτεί δρόμο μονής κατεύθυνσης.

Η απόσταση **D** της ακμής ανάμεσα σε δύο σημεία $\mathbf{p}_1$ και $\mathbf{p}_2$ υπολογίζεται από τη σχέση ***haversine distance*** που περιγράφεται από τις παρακάτω σχέσεις. **R** είναι η ακτίνα της γης και είναι ίση με **6378137** μέτρα.

$$D = R * c \quad , R = 6378137$$

$$c = 2 * atan2(\sqrt{a}, \sqrt{(1-a)})$$

$$a = sin^2(\Delta\varphi/2) + cos\ \varphi1 \cdot cos\ \varphi2 \cdot sin^2(\Delta\lambda/2)$$

όπου **ΔΦ = (p1.latitude - p2.latitude2)** και **Δλ = (p1.longitude - p2.longitude)**

Οι τιμές των *latitude* και *longitude* σας δίνονται σε μοίρες **deg** και μετατρέπονται σε ακτίνια **rad** από τη σχέση

$$rad = deg * (M\_PI/180)$$

Η απόσταση κάθε ακμής πολλαπλασιάζεται με μία σταθερά **factor** που είναι διαφορετική ανάλογα με την τιμή του κλειδιού **highway**. Συγκεκριμένα, η σταθερά **factor** λαμβάνει τις παρακάτω τιμές με βάση την τιμή του κλειδιού highway

- [**motorway, trunk**] **factor = 0.5**
- [**primary, secondary**] **factor = 0.75**
- **[tertiary, residential]** **factor = 1.0**
- **[living_street, unclassified]** **factor = 1.25**
- **[service, track]** **factor = 1.5**

Οδηγίες Αποστολής

Η αποστολή της εργασίας θα γίνει μέσω της πλατφόρμας *autolab*.

Συμπιέστε τα αρχεία της εργασίας σας σε μορφή zip, χωρίς να συμπεριλάβετε οποιονδήποτε κατάλογο. Τα αρχεία που πρέπει να περιέχονται στο zip file είναι τα εξής:

Makefile, Edges.hpp, Edge.cpp, Vertex.hpp, Vertex.cpp, Graph.hpp, Graph.cpp, HW5.cpp

Το αρχείο **HW5.cpp** περιέχει με συνάρτηση **main**. Συμπληρωματικά εφόσον το κρίνετε απαραίτητο μπορείτε να περιλάβετε ΜΟΝΟ μία επιπλέον κλάση. Θα πρέπει να παρέχετε κατάλληλο *Makefile* που να μεταγλωττίζει τον κώδικα σας. Το τελικό εκτελέσιμο έχει όνομα **hw5**. Όλα τα CPP αρχεία πρέπει υποχρεωτικά να μεταγλωττίζονται με την παράμετρο **-fsanitize=address**.

Τα αρχεία της βιβλιοθήκης **tinyxml2** υπάρχουν ήδη στο autolab και δε χρειάζεται να τα ενσωματώσετε στο zip.

Για την υποβολή ακολουθήστε τα εξής βήματα.

- **Υποβολή κώδικα:** Το αρχείο zip πρέπει να έχει όνομα **hw5.zip**. Συνδέεστε στο *autolab* και επιλέγετε το μάθημα **ECE326_2025 (S25)** και από αυτό την εργασία **HW5**.
- Για να υποβάλετε την εργασία σας κάνετε click στην επιλογή *"I affirm that I have compiled with this course academic integrity policy…"* και πατάτε submit. Στη συνέχεια επιλέγετε το αρχείο **hw5.zip** που δημιουργήσατε παραπάνω.