Vietnam National University – HCMC
Hochiminh University of Technology
Faculty of Computer Science and Engineering

Data Structure & Algorithm

Assignment 01: Transportation System

1 Introduction

Transportation system is always a concern wherever you go. It is a complicated system with various subsystems, sometimes does not connect. Building applications that help people find the optimal route for traveling is an important task in a smart city. In this assignment, the student will build an application to manage the subway system of the cities. Below is a part of Tokyo transportation system

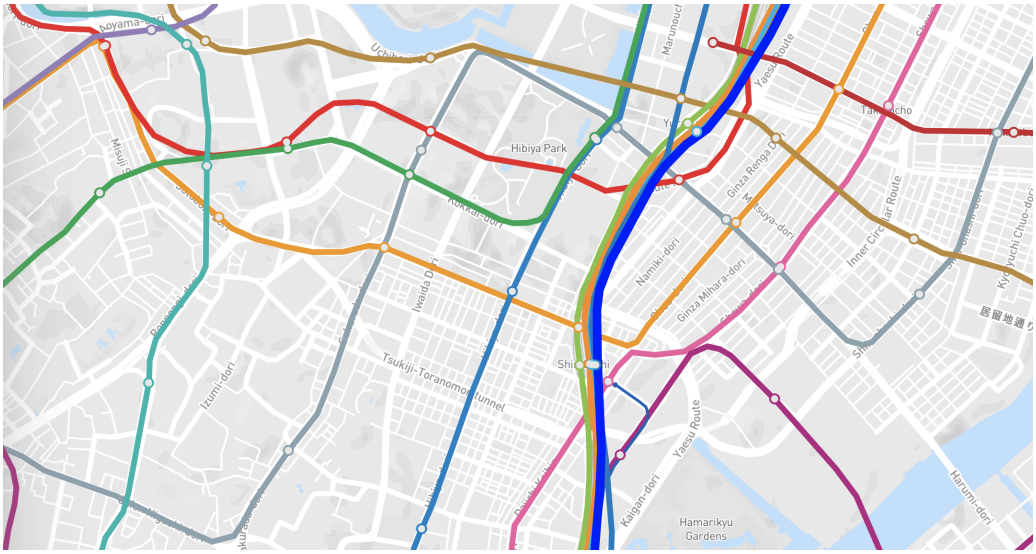

Figure 1: Tokyo transportation system.

The student should perform a deep analysis of problem requirements before start implementing. Also, the student is encouraged to follow the problem-solving process to do well in the assignment. In this assignment, the student must design the data structure and utilize them in the project with the highest performance. The linked list is the target data structure in this assignment. The student must implement the required features using only this data structure with no additional libraries. All required libraries have been included in the initial code. The detailed requirements are given in the next section.

2 Requirements

2.1 Dataset

Students have access to a simple dataset of stations of the subways system of big cities in the world. The dataset includes several tables in the form of CSV files: “cites.csv”, “station_lines.csv”, “stations.csv”, “systems.csv”, “lines.csv”, “lines.csv”, “track_lines.csv”, “tracks.csv”. This dataset (citylines.co) is open data licensed under the Open Database License (ODbL). Any rights in individual contents of the database are licensed under the Database Contents License (DbCL).

2.1.1 cities.csv

Cities.csv includes coordinates, start year, and other geographic information. This table has the following columns:

- id: integer
- name: string
- coords: string
- start_year: integer
- url_name: string
- country: string
- country_state: string

A sample of this file is given as follows:

Listing 1: cities.csv

```
id,name,coords,start_year,url_name,country,country_state
5,Aberdeen,POINT(-2.15 57.15),2017,aberdeen,Scotland,
6,Adelaide,POINT(138.6 -34.91666667),2017,adelaide,Australia,
7,Algiers,POINT(3 36.83333333),2017,algiers,Algeria,
9,Ankara,POINT(32.91666667 39.91666667),2017,ankara,Turkey,
16,Belem,POINT(-48.48333333 -1.466666667),2017,belem,Brazil,
10,Asuncion,POINT(-57.66666667 -25.25),2017,asuncion,Paraguay,
11,Athens,POINT(23.71666667 37.96666667),2017,athens,Greece,
12,Auckland,POINT(174.75 -36.86666667),2017,auckland,New Zealand,
13,Bangkok,POINT(100.5 13.75),2017,bangkok,Thailand,
```

The student should define the data structure according to the description above. You can ignore some information if they are unnecessary for processing requests.

2.1.2 lines.csv

Lines.csv includes coordinates, start year, and other geographic information. This table has the following columns:

- id: integer
- city_id: integer
- name: string
- url_name: string
- color: string
- system_id: integer
- transport_mode_id: integer

A sample of this file is given as follows:

Listing 2: lines.csv

```
id,city_id,name,url_name,color,system_id,transport_mode_id
43,4,Linea 2,43-linea-2,#ffbe2e,267,4
75,34,Linea 3 Metro de Caracas,75-linea-3-metro-de-caracas,#000,119,
107,126,Linea 1,107-linea-1,#434343,249,
604,74,La navette,604-la-navette,#009ab9,346,
```

```
61,66,Linea 1 (Tramo 1A),61-linea-1,#49aa43,250,
1471,91,Tokaido Shinkansen,1471-tokaido-shinkansen,#0000ff,551,1
5,1,A,a,#00b3da,254,4
168,261,Linea 1,168-linea-1,#f58223,256,
219,110,14,219-red-line,#d0021b,257,4
```

2.1.3 station_lines.csv

Station_lines.csv describes the relationship between lines and stations. This table has the following columns:

- id: integer
- station_id: integer
- line_id: integer
- city_id: integer
- created_at: date
- updated_at: date

A sample of this file is given as follows:

Listing 3: station_lines.csv

```
id,station_id,line_id,city_id,created_at,updated_at
47,7754,570,74,2017-11-21 00:00:00,2017-11-21 00:00:00
48,7771,571,74,2017-11-21 00:00:00,2017-11-21 00:00:00
49,7764,571,74,2017-11-21 00:00:00,2017-11-21 00:00:00
50,7763,571,74,2017-11-21 00:00:00,2017-11-21 00:00:00
51,7729,571,74,2017-11-21 00:00:00,2017-11-21 00:00:00
52,7772,571,74,2017-11-21 00:00:00,2017-11-21 00:00:00
53,8523,582,70,2017-11-21 00:00:00,2017-11-21 00:00:00
54,8525,582,70,2017-11-21 00:00:00,2017-11-21 00:00:00
55,93,10,1,2017-11-21 00:00:00,2017-11-21 00:00:00
```

2.1.4 stations.csv

Stations.csv describes detailed information on stations. This table has the following columns:

- id: integer
- name: string
- geometry: string
- buildstart: integer
- opening: integer
- closure: integer
- city_id: integer

A sample of this file is given as follows:

Listing 4: stations.csv

```
id,name,geometry,buildstart,opening,closure,city_id
7694,Keisei Tsudanuma,POINT(140.024812197129 35.6837744784723),1921,1921,999999,114
6003,Kossuth Lajos ter,POINT(19.0462376564033 47.5054880717671),0,0,999999,29
7732,Saint-Charles,POINT(5.3801556 43.3024646),1973,1977,999999,74
7695,Keisei Makuhari-Hongo,POINT(140.042146725175 35.6726021159981),1991,1991,999999,114
```

```
7726,Chartreux,POINT(5.4014815 43.309129),1973,1977,999999,74
7696,Keisei Makuhari,POINT(140.056077093286 35.6605591225961),1921,1921,999999,114
7719,Malpasse,POINT(5.4165033 43.3209501),1973,1977,999999,74
7697,Kemigawa,POINT(140.066304589107 35.6526280375642),1921,1921,999999,114
1,Caseros,POINT(-58.3989075634122 -34.6358418393779),2001,2007,,1
```

2.1.5 systems.csv

This file contains the name of the transportation system with the following fields:

- id: integer
- city_id: integer
- name: string

A sample of this file is given as follows:

Listing 5: systems.csv

```
id,city_id,name
1,5,
2,6,
3,7,
4,8,
5,9,
6,16,
7,10,
8,11,
9,12,
```

Please note that some systems don't have a name, so several fields will be left empty. Students can decide to use this table or not.

2.1.6 tracks.csv

This table describes detailed information of tracks, including a sequence of points stored in "geometry" field. It has the following columns:

- id: integer
- geometry: string
- buildstart: integer
- opening: integer
- closure: integer
- length: integer
- city_id: integer

A sample of this file is given as follows:

Listing 6: tracks.csv

```
id,geometry,buildstart,opening,closure,length,city_id
```

```
2  | 1911,"LINESTRING(19.0817752 47.5005079,19.0817355 47.5004893,19.0807974
   |     47.5000068,19.0803989 47.4998011,19.079491 47.4991458,19.0791327 47.4986762,19.0772964
   |     47.4955602,19.0766446 47.4945492,19.0760385 47.49405,19.0752771 47.493605,19.0746593
   |     47.4933489,19.0738419 47.4931273,19.0724272 47.4928741,19.0714014 47.4926904,19.0705006
   |      47.4925309,19.0698621 47.4923834,19.0690296 47.4921049,19.0618357
   |     47.4893223,19.0613785 47.4891046,19.060975 47.4889126,19.0593017 47.4879414,19.0587136
   |     47.4874644,19.0584927 47.4872223,19.0578868 47.4861782,19.057606 47.4857793,19.0573097
   |     47.4854394,19.0554189 47.4838484,19.054754 47.4832741,19.054014 47.482635,19.0534836
   |     47.4823146,19.0527699 47.4819185,19.0509559 47.4807818,19.0503245 47.4803739,19.0497254
   |      47.4799532,19.0491567 47.4794995,19.048538 47.4788871,19.0480326 47.478316,19.0476105
   |     47.4778133,19.0474229 47.4772914,19.0472593 47.4766126,19.0471328 47.4760528,19.0462516
   |      47.4735442,19.0459836 47.4728946,19.0454836 47.471696,19.0448219 47.4705031,19.0437705
   |      47.4693496,19.0422614 47.468224,19.0399778 47.4671626,19.0376454 47.4661899,19.0352558
   |      47.4655756,19.0334721 47.4651736,19.0319979 47.4648632,19.0307579
   |     47.4646427,19.0290703 47.4644266,19.0272172 47.4642833,19.0253275 47.4641549,19.0246336
   |      47.4641389,19.0239994 47.4641582,19.0226824 47.4642448,19.0222187 47.4642709,19.020656
   |      47.4643687,19.0195835 47.4644471)",0,0,999999,6719,29
3  | 2563,"LINESTRING(16.4151057 48.1907238,16.4156455 48.190389,16.4170845 48.1895171)
   |     ",0,0,999999,199,118
4  | 2557,"LINESTRING(16.4164437 48.1839655,16.4161534 48.1836515,16.4158173
   |     48.1833488,16.4155745 48.1831673,16.4153602 48.1830237,16.4151342 48.1828881,16.4148982
   |      48.1827604,16.4146926 48.1826603,16.4144833 48.1825675,16.4142825
   |     48.1824861,16.4141186 48.1824167,16.4140102 48.1823673,16.4139056 48.1823162,16.4137529
   |      48.182233,16.4135925 48.1821472,16.4134211 48.1820376,16.4131191 48.1818536,16.4128041
   |      48.1816269,16.4125639 48.1814033,16.4123453 48.181168,16.4120298 48.1808047,16.4116795
   |      48.1803475,16.4113292 48.1798903,16.4109949 48.1793831,16.4108121
   |     48.1789775,16.4107516 48.178733,16.4107157 48.1784378,16.4107385 48.1780427,16.4108399
   |     48.1776321,16.4111028 48.177078)",0,0,999999,925,118
5  | 2558,"LINESTRING(16.4164901 48.1839473,16.416198 48.1836313,16.4158591
   |     48.1833262,16.4156137 48.1831427,16.4153974 48.1829977,16.4151692 48.1828607,16.414931
   |     48.182732,16.4147234 48.1826309,16.4145123 48.1825371,16.4143219 48.1824491,16.414152
   |     48.1823599,16.4139933 48.18227,16.4138128 48.1821679,16.4136257 48.1820299,16.4134848
   |     48.1819043,16.4132958 48.1817187,16.4130732 48.1814777,16.4128921 48.1812542,16.4127223
   |      48.181028,16.412486 48.1806772,16.4121931 48.1801668,16.4119012 48.1796582,16.4117692
   |     48.1794077,16.4115911 48.1790904,16.4114857 48.1788783,16.4114265 48.1787169,16.4113737
   |      48.178553,16.4113528 48.1783256,16.4113766 48.1780488,16.4114572 48.1777929,16.4115651
   |      48.1775514,16.4117489 48.1772184)",0,0,999999,881,118
6  | 2564,"LINESTRING(16.415259 48.1908074,16.4153634 48.190746,16.4156079
   |     48.1905985,16.4162929 48.1901825,16.4167818 48.189893,16.4171761 48.1896436,16.417375
   |     48.1895146)",0,0,999999,213,118
7  | 2565,"LINESTRING(16.4120893 48.1927723,16.4130719 48.1921607,16.4132932
   |     48.1920235,16.413609 48.1918264,16.4141822 48.1914748,16.4147524 48.191125,16.4149727
   |     48.1909823,16.415259 48.1908074)",0,0,999999,321,118
8  | 2566,"LINESTRING(16.412035 48.1927423,16.4129962 48.1921133,16.4132141
   |     48.1919707,16.4135239 48.1917679,16.4140784 48.1914073,16.4146349 48.1910454,16.4147239
   |      48.1909826,16.4148596 48.1908865,16.4151057 48.1907238)",0,0,999999,320,118
9  | 2567,"LINESTRING(16.3949694 48.2005582,16.3984035 48.1985083,16.3993181
   |     48.1979618,16.3998962 48.197622,16.4004715 48.1972838,16.4011364 48.1969538,16.401813
   |     48.1966319,16.4032509 48.1960011,16.4039223 48.1957086,16.4059947 48.1948059,16.4067773
   |      48.194462,16.4072888 48.1942372,16.4075663 48.1941084,16.4080565 48.1938927,16.4083274
   |      48.1937854,16.4085071 48.1937282,16.4086963 48.1936746,16.4093969
   |     48.1935505,16.4103781 48.1933857,16.4110952 48.19324,16.4114368 48.1931169,16.4116053
   |     48.1930368,16.4120893 48.1927723)",0,0,999999,1555,118
10 | 3434,"LINESTRING(139.387327940585 35.3676500392482,139.386198840205
   |     35.3684306545863,139.385290650795 35.3688509827827,139.384308824406
   |     35.3693914015494,139.383400634996 35.3697516787171,139.382345171626
   |     35.3700118778978,139.381019705987 35.3703121066745,139.380185153557
   |     35.3707124099687,139.379276964116 35.3713328761492,139.378516048667
   |     35.3723736474238,139.378098772467 35.3733343474647,139.377853315847
   |     35.3743550787274,139.378074226787 35.375375797067,139.378466957367
   |     35.3767767620105,139.379535244568 35.3800839081475,139.379682518528
   |     35.3806442643415,139.379731609828 35.3817649650663,139.379608881548
   |     35.3828456260258,139.379363424928 35.3852270314683,139.379030739788
   |     35.3878792563645,139.378834374498 35.3886796866738,139.378515280928
   |     35.3894600985856,139.378328584795 35.3898890395009,139.378005028921
   |     35.3909847023726,139.377886621726 35.3917740554985,139.377903214371
```

```
    35.3924300870029,139.377928103256 35.393316059214,139.377936399578
    35.3937353709307,139.378027658919 35.3941073391642,139.378384400041 35.3948918483589)
    ",1926,1926,999999,3534,114
```

2.1.7 track_lines.csv

This table tells the line that each track belongs to. It has the following columns:

- id: integer
- section_id: integer
- line_id: integer
- created_at: date
- updated_at: date
- city_id: integer

A sample of this file is given as follows:

Listing 7: tracklines.csv

```
1  id,section_id,line_id,created_at,updated_at,city_id
2  2494,1278,343,2017-11-21 00:00:00,2017-11-21 00:00:00,252
3  4124,4477,779,2017-11-21 00:09:55.135507,2017-11-21 00:09:55.135507,63
4  2495,21,9,2017-11-21 00:00:00,2017-11-21 00:00:00,1
5  2496,940,228,2017-11-21 00:00:00,2017-11-21 00:00:00,79
6  4129,4478,793,2017-11-21 17:44:39.765832,2017-11-21 17:44:39.765832,48
7  4139,4483,797,2017-11-23 21:51:04.515237,2017-11-23 21:51:04.515237,114
8  4180,4522,657,2017-11-30 17:36:52.150194,2017-11-30 17:36:52.150194,1
9  4181,4523,657,2017-11-30 18:02:36.400933,2017-11-30 18:02:36.400933,1
10 4182,4524,657,2017-11-30 18:02:36.429198,2017-11-30 18:02:36.429198,1
```

2.2 Requests

In this assignment, the application will receive a sequence of requests. These requests are represented in the form of input text as follows:

Request	Output	Description
CL	integer	Count the number of lines in the dataset.
CL <city_name>	integer	Count the number of lines in the given city. If the city does not exist, return -1.
LSC <city_name>	integer array	List stations (station_id) of a city (given <city_name>). The order of stations is determined by their appearance in stations.csv.
LLC <city_name>	integer array	List lines (line_id) of a city (given <city_name>). The order of this list is given in lines.csv.
LSL <line_id>	integer array	List stations (station_id) of a line with <line_id>. The order of station is determined by its appearance in station_lines.csv.
FC <city_name>	integer	Find a city with the given name. Return the first city_id if found, -1 otherwise.
FS <station_name>	integer	Find a station with the given name. Return the first station_id if found, -1 otherwise.
SLP <station_id> <track_id>	integer	Find the position of a station in a track. Return the index of that station if found, -1 otherwise. The order of station is determined by LINESTRING in tracks.csv.
IS <csv_description>	integer	Insert a station into the dataset. The information of the station is given in csv_description, which includes everything you see in stations.csv except the id and city_id. If this operation success, we have to receive station id as the return value. Note that we keep track the maximum id of every entities in the dataset so that when you insert a new instance, the allocated id should be the $id_{max}^{type} + 1$.
RS <station_id>	integer	Remove a station from the dataset. As a consequence, every records related to the station must be removed. Return 0 if success, return -1 if the station does not exist or can not be removed.
US <station_id> <csv_description>	integer	Update the information of a station with id <station_id>. Return 0 if success, and -1 otherwise.
ISL <station_id> <line_id> <pos>	integer	Insert a station <station_id> to a line <line_id> at index <pos>. Note that the first position has index 0. Return 0 if success, and -1 otherwise. NOTE: The expected result of this request is that when we request LSL, the new station must stay at position <pos>. If the station exists in the line, this operation will fail and no change should be made.
RSL <station_id> <line_id>	integer	Remove a station <station_id> from a line <line_id>. Return 0 if success, and -1 otherwise.

The requests are given in the form of strings, and the location of the output is given in an input address. The student must implement the function to load data from the given files into the defined data structure.

3 Implementation

An initial code package is provided with some source files. The main.cpp file contains the main function of the application and should not be modified since it will be overwritten during the grading process. Two files processData.cpp and processData.h will be used for implementing features for this program. The data structure should be defined and implemented in dsalib.h, dbLib.h and dblib.cpp. The implemented methods in these files must remain intact since it can affect the framework operations. Students can implement, update, and add more members and methods for their data structure.

4 Regulation

4.1 Evaluation

The program output will be compared with the expected output from the solution. A test case passes if everything in the program output matches to the solution. The test case will fail if the program runs too slow (timeout error). This issue should not occur in this assignment but the student must be aware.

4.2 Submission

The student should follow the submission instruction on the course site (e-learning site). One file src.zip contains processData.cpp, processData.h, dsalib.h, dbLib.h, and dblib.cpp must be submitted during the given timeline. The OS system that builds and run code is Linux (Ubuntu). Please do not use any special functions that do not exist on Linux, otherwise, your code cannot be built. Student can build their code on Linux before submit to ensure that the code works.

The deadline for this assignment is 23-09-2019. The grading system will open at the deadline.

4.3 Rules

The student must perform this assignment themselves. Copying code is strictly prohibited and be considered as an ethic code violation. In such a case, the course result will be zero regardless of the upcoming assignments. Please protect your code carefully. Student can help their friends by giving advise, not code.