CS 409 - UG SOFTWARE LAB
Lab Assignment 2 Weightage: 10%
Submission Deadline: 13-Feb-2022 11:59pm

General Instructions:

- You must use only C, C++, Python or JAVA for this assignment.
- Prescribed specifications must be strictly followed. Failure to do so may lead to substantial loss of points.
- Make sure your code is well written (self explanatory variable names) and documented. You are likely to lose points if your TA cannot understand your code.
- You must submit a README file along with your code with clear instructions on how to run the code.
- Any assumptions made must be clearly stated in a README. Suitable ones would be accepted and graded accordingly.
- **You are not allowed to use any specialized libraries in this assignment. You are allowed to use only the primitive data types in the language such as arrays, structures, classes, hash_map, associative array, etc.**
- **Your code must not have any directory structure in it. All the code files must be inside a single directory. Also, you must include a readme file on how to run your code.**

Question 1 (100 points):

In this assignment you would be creating a partitioning of a road network dataset which would be used in the subsequent assignments. Consider the road network dataset which has been loaded with this assignment. This dataset models an underlying road network as a graph. Nodes represent road intersections, whereas edges represent road networks. Each node is associated with a x and y coordinates representing its spatial locations. Each edge is associated with a notion of edge length.

Partitioning Algorithm:

Determine the x_min, x_max, y_min, y_max among all the node coordinates. The rectangle defined by (x_min,y_min) as bottom left and (x_max,y_max) as upper right corner would define a box fitting the entire dataset. Start overlaying cells of size **kXk** (k to be varied during evaluation) over this rectangle to partition the graph. It is conceivable that the cells (being square in size) would not fit the rectangle perfectly. To address this case, you may extend the x_max and/or y_max boundary line to fit them. This would create some empty space in the cells at the boundary, but you can choose to ignore it. The portion of the graph inside a cell would be stored in a single disk block (along with overflow blocks as needed) which is simulated as a file in your code. In your code, you need to define a disk block as a file which can store only **B number of entries (distinct nodes or edges).** Do not hard code the value of **B.** It may be changed during the evaluation.

Consider an arbitrary cell **C** in this grid. This cell **C** is defined by its upper and lower limits on its x and y coordinates. All the nodes which are spatially inside this cell would be stored in the disk block corresponding to this cell. In addition, all the edges which are amongst the nodes inside a cell would also be stored in the same disk block. And lastly, the boundary edges (i.e., edges which have one node in **C** and another outside) would also be stored in the same disk block. Note that in your data structure boundary information (nodes and edges) would effectively be stored twice. In case all the required entries for a cell does not fit in a disk block, you must create an overflow disk flow (as simulated as a file with B number of entries).

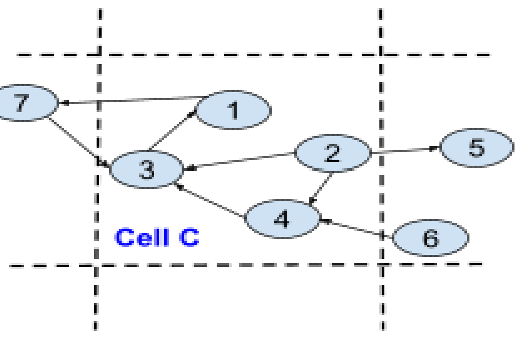

Figure 1: Sample Location Cell C

Following would be the disk block corresponding to cell C. Here, we are assuming that disk block size is 10 entries.

```
------------------
1  <x coordinate, y coordinate>
2  <x coordinate, y coordinate>
3  <x coordinate, y coordinate>
4  <x coordinate, y coordinate>
## <special character indicating start of edges within cell>
3 1 <edge length>
2 3 <edge length>
2 4 <edge length>
4 3 <edge length>
** <special character indicating start of boundary information>
5  <x coordinate, y coordinate>
6  <x coordinate, y coordinate>
?? <special character indicating start of overflow page> FIle name of overflow disk block

<Contents of Overflow disk block>
?? File name of the main disk block
7  <x coordinate, y coordinate>
%% <special character indicating start of boundary edges information>
1 7 <edge length>
7 3 <edge length>
2 5 <edge length>
6 4 <edge length>
-------------------------
```

Your code must also have the following: (a) a map from node id → x and y coordinate; (b) A function to determine the appropriate cell id given an x and y coordinate. For this function, you should internally have a data structure which stores the x and y range values for each cell. (c) a map between the cell ids and the file names corresponding to their disk blocks.

Evaluation:
Your code would be evaluated for both small and large datasets. On small datasets, the TAs would check the correctness of the code. Scalability would be tested on the large graphs. You must also implement a small "visualizer" function which would print the disk block (and its associated overflow block) contents of any given node id. For example, the function should print the above mentioned disk block (and the associated overflow block) contents if the query node is 1, 2, 3 or 4.