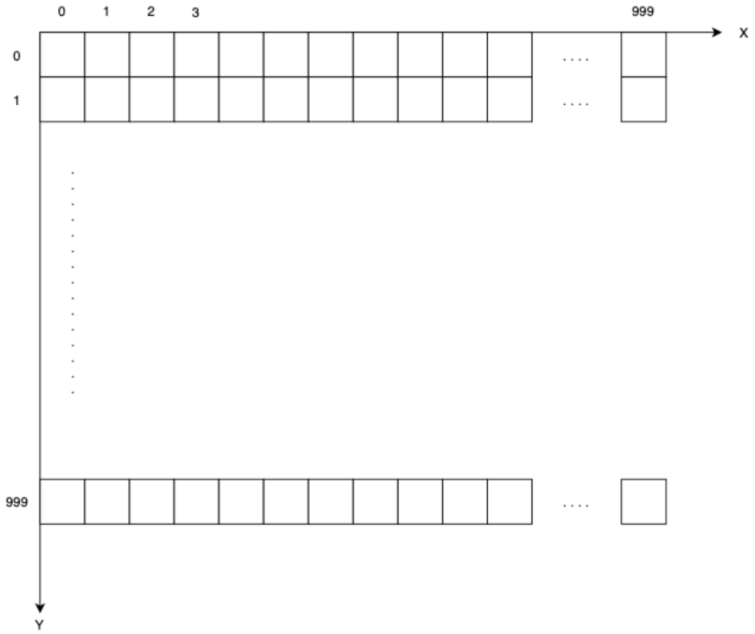
CSCE3304 – Spring 2025
Project - Maze Router

The objective of this project is to develop a maze router (using a language of your choice) that implements ~~both~~ Lee's algorithm~~s~~. The router connects pins that belong to the same net together using the available routing resources. There are two routing layers (M1: horizontal and M2: Vertical) routing grid is up to 1000x1000 cells foreach layer.



The input to the router is a text file that specifies the size of the grid, the obstacles and lists the nets to be routed. Net's definitions follow the following syntax

```
Net_name (pin_1_layer, pin_1_x, pin_1_y) (pin_2_layer, pin_2_x, pin_2_y) …
```

For example

```
100x200
OBS (33, 44)
OBS (55, 77)
net1 (1, 10, 20) (2, 30, 50) (1, 5, 100)
net2 (2, 100, 200) (1, 300, 50)
net3 (1, 100, 50) (2, 300, 150) (2, 50, 50) (1, 2, 2)
.
.
```

The output from the router is a text file that lists the cells used by each net. Each line of this file looks like:

```
Net_name (cell_1_layer, cell_1_x, cell_1_y) (cell_2_layer, cell_2_x, cell_2_y) …
```

For example

```
net1 (1, 10, 20) (1, 11, 20) (1, 12, 20) …
net2 (2, 100, 200) (2, 100, 201) (2, 100, 202) …
```

Your algorithm should minimize the usage of vias (to move between layers), for that assign high cost to vias. You may route wires vertically on M1 and horizontally on M2 but do that only when there is no other choice (give a high cost for that). Via usage and non-preferred routing direction costs must be specified by the user.

Finally, you need to develop a script to visualize the output file to examine the routed nets.

Bonus (15% - Pick one only):

- Implement a net re-ordering heuristic
- Add support for rip-up and re-route to your router when it fails to route some nets.

Rules

- Work in a group of 3 students.
- Use GitHub (from day 1)
- MS1 Deliverables:
    - A demo of a Semi-functional router with at least 3 test cases.
    - Bugs are allowed
    - A single layer routing is allowed
- Final submission must include:
    - Documented source code
    - Readme file outlining the assumptions, how to use, limitations, ….
    - Examples/Test cases
    - Power point presentation
        - Technical background as well as design and implementation aspects
        - Examples
        - Problems/Limitations
        - Conclusions
- Grading
    1. -10% for not using GitHub
    2. 25%: MS1 Deliverables
    3. 10% for the Power Point presentation.
    4. 15% for the test cases (at least 8); different grid sizes, different number of nets, different complexities (e.g., 1 layer only, 2 layers, …), …
    5. 50% for Source code and the correctness of the o/p
        - 15% The results visualization
        - 30% correctness of the o/p
            - -15% for not supporting 2 layers
        - 5% for GitHub Readme file and the source code comments

Deadline:

- MS1 demo and deliverables: May 11.
- Final demo and deliverables: May 21.