CSCE3304 – FALL 2024
Project - Maze Router

Efficient routing of connections between components is essential in modern integrated circuit (IC) design, as it directly impacts performance and manufacturability. This project aims to develop a **maze router**—a type of routing algorithm designed to establish paths between multiple points on a grid-based layout while navigating obstacles and minimizing routing costs. This maze router is specifically tailored for a two-layer routing environment, where it will manage multi-point connections (nets) and optimize routes across two layers, M0 and M1, which have distinct directional preferences: M0 is optimized for horizontal routing, and M1 for vertical.

The routing area is organized as an NxM grid of uniformly sized cells, each with an associated cost. Deviating from the preferred directions on each layer incurs a *bend penalty*, while transitions between layers (vias) come with an additional *via penalty*. This router will intelligently account for these penalties, generating optimized paths for each net. As a result, it serves as a foundational tool for IC layout design, addressing both physical constraints and cost considerations integral to the routing process.

Input Format:
The router's input is a text file containing the following information:

```
100, 200, 20, 5
OBS (0, 33, 44)
OBS (1, 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)
.
.
```

- Line 1: Specifies the grid's dimensions (number of horizontal and vertical cells), along with penalties for bends and vias.
- Lines 2-3: Define obstructions. Cells at coordinates (33, 44) on M0 and (55, 77) on M1 are blocked and cannot be used.
- Lines 4 onwards: Define each net. The format for each net is:

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

Output Format:
The router generates a text file that lists the sequence of cells used by each net. Each line follows this format:

```
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) …
```

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

Visualization:
You will also create a script to visualize the routed nets from the output file, allowing for easy examination of the routing paths.

Bonus (10% - Pick one only):
- Implement a net ordering heuristic
- Add support for rip-up and re-route to your router when it fails to route some nets.

Project Rules

- Group Work: Form a group of 3 students for this project.
- GitHub Development: Use GitHub to manage and develop your project.
  - The readme.md file will serve as your report, detailing the implementation, compilation, and execution instructions. Also, incliude the challenges and the contributions of each member.
  - Include at least 8 test cases. The test cases must provide meaningful scenarios. For example, 2-pin nets, multi-pin nets, different values for the penalities, etc. Implementing a bonus requires extra test cases (at least 3 more).
  - Ensure the source code is well-documented.
  - Follow a team workflow, which should be outlined in the readme file.
  - Each team member's contributions will be reviewed, and individual grades will reflect these contributions.

Grading Breakdown

1. GitHub Repository Organization: 5%
2. Readme File: 10%
3. Test Cases (at least 8): 10%
4. Intermediate checkpoint: 20%
   - 3%: a GH repo showing a consistent progress and contrubutions from all members.
   - 3%: At least 4 test cases
   - 12%: The router can produce meanigful output; however bugs may exist.
   - 2%: Plans for the visulaization
5. Source Code and Output Correctness: 55%
   - 15%: Visualizing results
   - 20%: Passing private test cases
   - 5%: Source code organization and comments
   - 15%: Interview assessment

Timeline

- Nov 24: Intermediate Interview
- Dec 8: Final Interview