Diagram, engineering drawing

Description automatically generatedHomeWork

Logo

Description automatically generatedCPP

nng.com

CPP DEVELOPER HOMEWORK

House number

**Task Description – Detecting Duplicate House Numbers**

Your task is to identify and list house numbers that appear **more than once** in a street network dataset.

**Problem Context**

A house number is considered **duplicate** if it appears in **multiple segments** of the **same street**. For example:

* If "Kossuth utca" appears on the **odd side** from 43–47 in one segment, and again from 47–49 in another, the house number 47 is duplicated.
* If there’s also a segment from 45–59, then **45 through 49** are all duplicated.
* You do **not** need to treat triplicates or higher overlaps differently—any number that appears more than once counts as a duplicate.

Mixed/odd/even overlaps can be handled as you see fit, as long as your solution is logical. It is acceptable if a mixed overlap is reported separately as odd and even in the output.

**Input**

You are provided with a file named network.mid.  
Each row represents a street segment and includes:

* + Street name
  + Parity scheme (O for odd, E for even, M for mixed)
  + Starting house number (from)
  + Ending house number (to)

Only the necessary columns for solving the task need to be read and processed.

Example:

|  |
| --- |
| 514450,391593,391582,"6",3,"","",58,"F","NNNNNNN",50,0,30,0,0,0,"Kossuth Lajos","utca","","","",0,0,"",0,0,"Budaörs","Budaörs","","",244,244,"2040","2040" |

**Output**

Write the results to **standard output** using the following format:

|  |
| --- |
| [Street Name]: [Scheme] [From] [To] |

Example:

|  |
| --- |
| Kossuth utca: O 45 49  Kossuth utca: E 12 12 |

**Performance Requirements**

* Your solution’s **runtime must not depend on the magnitude** of the house numbers.
* For example, multiplying all house numbers by 100 or 101 should not significantly impact performance.

**Technical Requirements**

* Use **C++**.
* Implement the solution as a **console application**.
* The input file is network.mid, and output goes to standard output.
* Keep data loading and processing (algorithm) logic **clearly separated** in your code.
* You are free to implement the data loading in any way you prefer, but the algorithm should be easy to understand.

**Submission & Version Control Requirements**

* The project must be version-controlled using **Git**.
* All code should be committed **regularly** with **meaningful commit messages**.
* The complete project must be hosted on a **public GitHub repository**.
* The repository should include:
  + A README.md file with a brief project description and usage instructions
  + The full source code
  + Any sample JSON inputs used for testing (if applicable)
* Submit the project by sharing the **GitHub repository URL**.