Project created in Visual Studio 2022.

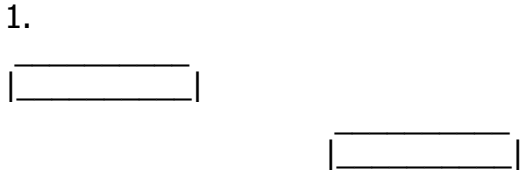
Source files:
DuplicateChecker\DuplicateChecker\

Solution documentation:
I assume the network.h is not supposed to be modified, however I took the liberty of changing the char arrays to strings for convenience. I figured the focus of the assigment will be on the algorithm itself, not the file processing part. I would take a slightly different approach if the header file was not modifiable. The contents of network.csv are parsed line by line and each line is split via the comma separators and then added to a vector of Networks. Not all of this data is needed to complete the assignment, however I did save a (later sorted) list of the contents for reference, so the readFile function stayed like this. For efficiency the unneeded data parsing can be skipped.

Algorithm:
My assumption is that a triplet of order8_name, street_name and street_type will uniquely define a street, so the approach is to sort the vector of networks using a custom comparator dependent on these 3 values. After sorting the addresses containing the same street will be next to each other, making it easier to digest in the next step. Actually the names containing hungarian characters get sorted after the ones not containing them, but for the purposes of this program this is not a problem.

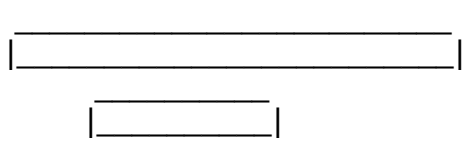
The house number indices will be treated as intervals. Meaning, the next step is to find all the overlapping intervals after loading the indices into 2 vectors (one for odd numbers, one for even numbers).
We sort the passed interval vector in findDuplicateIntervals, then theoretically we will have 3 options. We would have more than 3, but sorting by the first index limits the possibilities somewhat.

1.

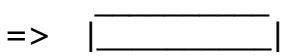


No overlap.

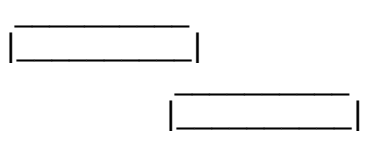
2.



Overlap, and we take the 2nd interval as duplicate.



3.



Overlap, we take the beginning of the 2nd interval and the end of the 1st interval as duplicate.

The duplicate search algorithm has a time complexity of O(n^2), which is not ideal, but since we are working with smaller chunks of data after sorting the initial list, it should not be a big deal. For long data I would look for a more efficient way.

The last step is merging the duplicate intervals since many of them will still overlap with each other.