Design Document for A1 Part B: Fingerprint Groups

Sofia Mancini
Due: 31st February, 2025
Course: CSC 411 Machine Organization

With good input:

The fingerprint groups will be stored in a HashMap<String, Vec<String>>, where the key (String) will store the fingerprint.
The fingerprints will be one or more non-whitespace characters. I will use the Rust macro split_whitespace() to separate finerprints from names.
The names will be stored in the Vec<string> variable. These will contain all subsequent data on the same line, including whitespaces.
The for loop over 'for line in lines' will iterate after this name.

I will use a HashMap<String, u64> to store the number of instances where a fingerprint is encountered.
The (String) will contain the fingerprint itself, and the (u64) will store the number of instances of that fingerprint.
The invariant that holds true during this loop is;
At any point during input processing, the HashMap<String, Vec<String>> will store fingerprints as keys and their associated names as values.
The HashMap<String, u64> will maintain a count of occurrences of each fingerprint.

Another loop will print the associated names, each on their own line, of any fingerprint group with at least two members.
Different fingerprint groups will be separated by a newline.

With bad input:

If the input line is malformed, an error message will be printed to stderr, the line will be discarded, and the function will more to the next line.
If the given name is too long, an error message will be printed to stderr, the name will be truncated and the subsequent name will be used.
A function will iterate through the initial list of fingerprints and names and remove any duplicates.

To handle performance:

A single for loop will handle multiple functions, including duplicate checking and the storage of fingerprint groups.
A for loop over a HashMap should be O(n) complexity. The operations conducted on the HashMap (insert, count, etc) would require O(1) complexity.
This should result in an overal complexity of O(2n).