In this project, I implemented three programs for reading a text file and counting the occurrences of the letter ‘a’ in them. The first program written in C++ uses ifstream for reading the file character by character. Similarly, the second program in C also reads the file character by character till the end of file is reached. The third program in C uses a mapping library to first map the file in memory, thus allowing the file to be accessed as an array in memory. This bypasses the need to refer to the file for its contents as the file contents are now stored in a portion of the processes’ virtual address space.

The C++ program proved to be the slowest. The C function without mapping was faster than the C++ function. This may have been due to the buffer in C program which reduces the number of file accesses. Buffers of fixed sizes copy the data into memory which is much faster than directly reading it from the file. Although making the program faster, the buffers have the disadvantage of needlessly copying data as they introduce an additional step between the file and the data.

The memory mapping implementation proved to be the fastest of the three. In memory mapping, the file contents are mapped to a pointer and data is filled in by the OS. This proves to be very fast as the file contents are mapped directly to memory, eliminating the problem of unnecessarily copying data. Memory mapping allows the user to not worry about remaining memory as the OS utilizes virtual memory to simulate additional memory. It grants users the freedom to load programs in memory which might be greater than RAM. Additional benefit of memory mapping is the reusability of ordinary, simple memory access instructions. Despite being very fast memory mapping has a tradeoff between speed and memory. The file data has to be loaded into

the memory and will consume a lot of memory for big files. Therefore, the speed comes at the cost of additional memory.

In my opinion, file sizes also play an important role in dictating the performance of the methods. The time for the first program would vary linearly, in a direct proportion to the file size. Memory mapping in small files will also provide a speed boost which may be negligible when compared with other methods of accessing file contents. However, for very large files, larger than RAM, the improvements in speed might not be significant as slower storage medium is used to mimic the much faster RAM. Still in my opinion, it would be best to use memory mapping either way due to its advantage of accommodating files larger than RAM.