**Programming Project Report**

Name: William Taylor

Date: 3/3/2020

**Academic Integrity Statement:** I pledge that I have neither given nor received unauthorized help on this programming assignment.

**Problem Statement:**

In this project our goal is to create a program that can “read a book” and then tell the user it’s estimated reading level. The program takes in a file name from the user, and the actual file, as well as the “words.txt” file. The error handling was for if the user entered an invalid file name or the file itself wasn’t readable. If the user doesn’t input a valid file name the program will report an error and return 0.

**Design:**

For this program we used binary search as a data structure. I also used for loops and while loops to implement algorithms. The pros of using binary search is that it’s one of the fastest data types for finding stuff. The con is that it’s my first time using binary search on my own. I decided not to create separate classes, because in the past projects the hardest part for me was navigating back and forth between different files.

**Implementation:**

I didn’t start with any sample code. For my midpoint I got my program to read in a “book” and “clean it” (remove invalid characters and make everything lowercase). After for the rest of the project I used Professor Gauch’s presentation slide titled “Binary Search Implementation” (slide 27/68 of chapter 4). After this I had to change around variable names to match my code. I also had to manipulate it a bit to work with the files I was given. My timeline was pretty spread out and constant compared to previous projects. I did the midpoint late, but after that I worked on the project about 1 to 2 hours everyday till

March 3rd (when it was due).

**Testing:**

A screenshot of a computer

Description automatically generatedFor this part I t tried to test my program using Turing. I successfully uploaded the files but couldn’t get them to run.

A screenshot of a computer

Description automatically generated

I also tested my program with online gdb. I tested them with the 5 example problems, and I also tested it with 3 of my own special files (included in zip file). The only one that threw me for a loop was “sample5.txt” This was because it used ‘!’ and gave my program a segmentation fault. I then went back and added ‘!’, ‘?’ to be deleted during “cleaning”. For the give samples my program returned :

1: 288.72

2: 381.445

3: 315.66

4: 278.557

5: 241.303

Special1.txt was the words “complex complex complex;”

It returned a reading level of ‘1001’

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generatedSpecial2.txt is my entire senior year (of high school) google doc. It got a rating of 381.445

Special3.txt is the phrase “The quick brown fox jumps over the lazy dog” in Hebrew.

A screenshot of a computer

Description automatically generatedThe difficulty is 1001, although I think it should be higher.

**Conclusions:**

The overall result was successful. The program was accurate to a degree compared to the examples we were given. The only thing that didn’t go well was using Turing to test the files.

Next time I would try to using Turing before, then go into office hours when it didn’t work. This project took around 10 hours to complete, however not 100% of attention was being used till the day it was due.