**Tyler Holland**

12/3/09

Lab 8

No Cheating Signature:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Demo Signature:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**I did not use functions, so I do not have a main.h**

**The sections are commented in my code**

**main.c:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

**/\***

**\* I did not use functions, so I commented on when**

**\* each Part of the program was starting in my code**

**\*/**

struct grep

{

char lineOn[100];

int lineNumber;

int wordLocation;

struct grep \*next;

};

int main (int argc, const char \* argv[])

{

int i;

char file[100];

char word[100];

int check = 0;

**//Variables for Part 2**

FILE \*infile;

char lineString[100];

int lineLength = 0;

int lines = 1; //Assuming first line = line 1

char currentChar = 'z';

int counter = 0;

**//Variables for Part 3**

int atWord = 0; //For searching the tokens

char temp[100]; //Space for current line

char storage[100]; //copy of temp to use strtok on

char \*result = NULL;

int occurrences = 0;

struct grep \*topOccurrence = NULL;

struct grep \*current = NULL;

struct grep \*newGrep = NULL;

check = scanf("myGrep %s %s", file, word);

if(check < 2) //Did not scan entire line

{

if(check == 0) //myGrep or file name mistype

{

printf("Error: myGrep or file name not recognized\n");

}

else if(check == 1) //search word mistype

{

printf("Error: search word was not recognized\n");

}

printf("Correct Usage:\nmyGrep filename.txt searchword\n");

return 0;

}

else

{

printf("myGrep %s %s", file, word);

}

//Open the file

infile = fopen(file, "r");

//Start scanning

while (fgets(temp, 100, infile)!= NULL)

{

//Remove newline from end of temp

i = 0;

while (temp[i] != '\n')

{

i++;

}

temp[i] = '\0';

**//Part 2**

counter = 0;

currentChar = 'z';

while (currentChar!= '\0' && currentChar != '\n') //Line length check

{

currentChar = temp[counter];

counter++;

}

if (counter > lineLength) //LONGEST LINE WILL BE THE FIRST LINE when equal in length

{

lineLength = counter;

for (i = 0; i < 100; i++)

{

lineString[i] = temp[i];

}

}

**//Part 3**

atWord = 1;

for (i = 0; i < 100; i++)

{

storage[i] = temp[i];

}

result = strtok(storage, " ");

while (result != NULL)

{

if (strcmp(result, word) == 0)

{

newGrep = (struct grep\*) malloc(sizeof(struct grep));

for (i = 0; i < 100; i++) //Copy line

{

newGrep->lineOn[i] = temp[i];

}

newGrep->lineNumber = lines; //Copy line number

newGrep->wordLocation = atWord;

if (topOccurrence == NULL) //First occurrence

{

topOccurrence = newGrep;

}

else

{

(\*current).next = newGrep;

}

current = newGrep;

occurrences++;

}

atWord++;

result = strtok(NULL, " ");

}

lines++;

}

//Printout

**//Part 2**

printf("\nLongest Line: %s\nNumber of Characters in line: %d\nNumber of Total Lines: %d\n",lineString, (lineLength - 1), (lines - 1));

**//Part 3**

printf("Total Occurrences of word: %d\n", occurrences);

current = topOccurrence;

while (current != NULL)

{

printf("Line %d: Word %d: %s\n",

current->lineNumber, current->wordLocation, current->lineOn);

current = current->next;

}

fclose(infile);

return 0;

}