Assignment 1

Kirollos Saleh Farah

202002552

Operating System

Section 1A

Overview

This C code provides a simple implementation of the Unix ‘cat’ command with the ability to read from, write to, and edit text files. The code reads input from the user and performs various actions based on the given input. The code uses a variety of functions to open files, update files, remove characters from strings, append to files, and overwrite files.

Data Structures

The code uses the following data structures:
char str[100]: An array of characters to store user input.
char *str[10][10]: A two-dimensional array of characters to store the file names passed in by the user.
char *filenames[30]: An array of pointers to characters to store the names of the files to be operated on.
char buffer[256]: An array of characters to store the contents of a file.

Functions

The code uses the following functions:

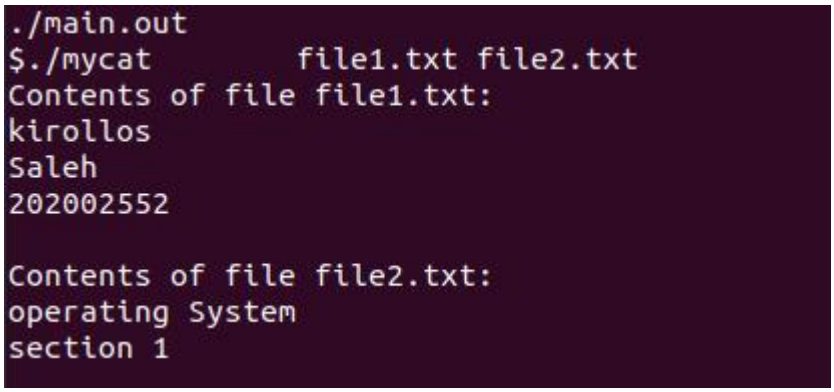
void openfiles(char* x)

This function opens files specified in the input string x for reading and prints their contents to the console.

Inputs

char *x: A pointer to a character array containing the file names.

Outputs



Assumptions

The input string contains the names of one or more text files separated by spaces.

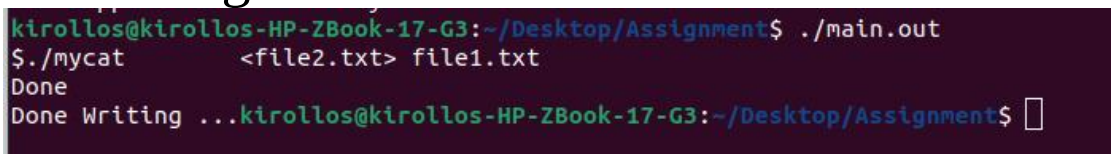
void updatefile(char* x)

This function opens the first file specified in the input string x for reading and the second file specified for writing. It then writes the

contents of the first file to the second file.

Inputs

char *x: A pointer to a character array containing the file names.



Outputs

Assumptions

The input string contains the names of two text files separated by <> characters for the file that we write its content in the second file (ex:

the content of file2.txt was written in the file1.txt)

void remove_chars(char *str, char c1, char c2)

This function removes characters c1 and c2 from the input string str.

Inputs

char *str: A pointer to a character array.
char c1: A character to remove from the string.
char c2: Another character to remove from the string.

Outputs

The string (‘str’) without the ‘<’ and ‘>’

Assumptions

The input string contains one or more c1 and c2 characters.

void remove_char(char *str, char c1)

This function removes character c1 from the input string str.

Inputs

char *str: A pointer to a character array.
char c1: A character to remove from the string.

Outputs

The string (‘str’) without the ‘<’ or ‘>’

Assumptions

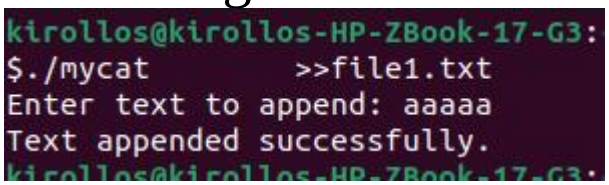
The input string contains one or more c1 characters.

void append(char* filename)

This function appends text to the end of the file specified by filename.

Inputs

char *filename: A pointer to a character array containing the name of the file to be appended.



Outputs

Assumptions

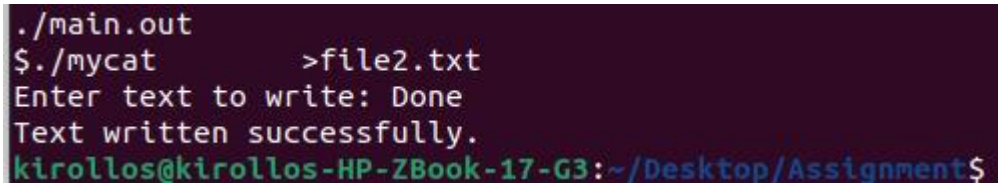
The file specified by filename exists and is writable.

void overwrite(char* filename)

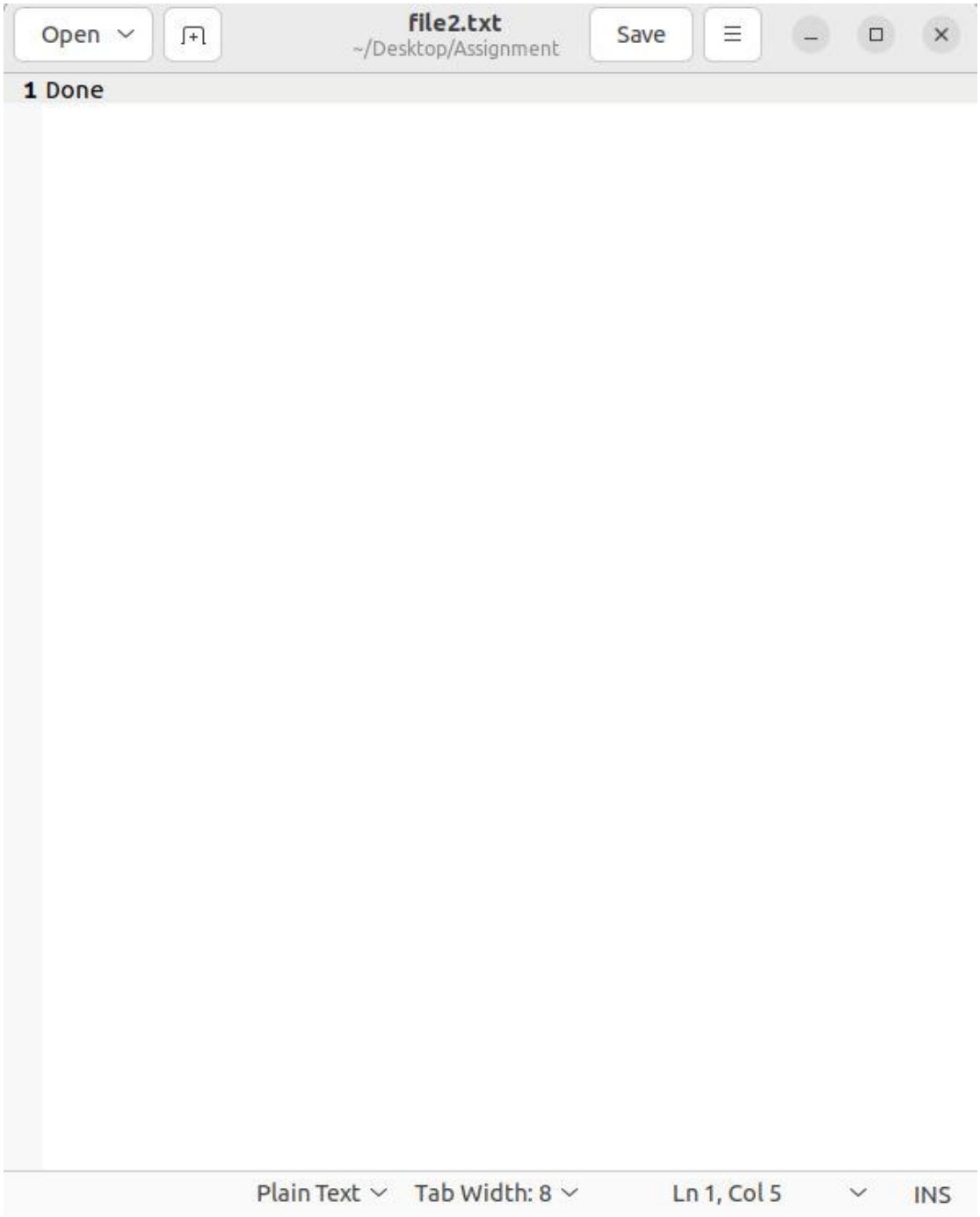
This function overwrites the contents of the file specified by filename with new text.

Inputs

char *filename: A pointer to a character array containing the name of the file to be overwritten.



Outputs



Assumptions

The file specified by filename exists and is writable.

int main()

This function is the main entry point for the program. It reads input from the user and calls the appropriate functions to perform the requested action.

Inputs

Input from the user applied on the mycat command

Outputs

Output based on the command or the function.

The Pseudocode

```
void openfiles(string:x){
    char *y= x;
    char str[10][10]; // 2d array to store the file
names
    strcpy(*str,y); passed the filename and put it in
the array
    char* filenames[30];
for loop- i< number of files then:
check if the input of the text file ended by ‘.txt’ if
true check the filename was available for opening
if true then open and save the content in the buffer
but if not display ‘"Error: Could not open file’

close file
}
```

Void updatefile(string x)
char *y= x;
    char str[10][10]; // 2d array to store the file names
    strcpy(*str,y); passed the filename and put it in the array
    char* filenames[30];
check the filename was available for opening if true then open the file then append the content of the file putting between <> in the second file then display ("Done Writing ..."); but if not display ‘"Error: Could not open file’

close file1
close file2
}

Void remove_chars(string str , char c1, char c2){
For loop until the end of the string :
If the string had < and > then skipped and loop on the second character so we have the string without the characters removed or skipped
}

Void remove_chars(string str , char c1){
For loop until the end of the string :
If the string had < or > then skipped and loop on the second character so we have the string without the characters removed or skipped
}

void append(char* filename){
char text[100];
 printf("Enter text to append: ");
then take the input
if the file was not available then display "Error opening file”
if the file was available then append the text in the file specified
then display "Text appended successfully”
then close the file
}

void overwite(char* filename){
char text[100];
 printf("Enter text to append: ");
then take the input
if the file was not available then display "Error opening file”
if the file was available then overwrite the text in the file specified
then display "Text written successfully”
then close the file
}