Name: K Srikanth Roll Number: 17ETCS002124

Laboratory 2

Title of the Laboratory Exercise: Programs using file management system calls

1. Introduction and Purpose of Experiment

   A system call is a programmatic way in which a computer program requests a service from the kernel of the operating system it is executed on. There are different types of system calls developed for various purposes. They are mainly classified as process management, file management, directory management. By solving the problems students will be able to apply file management system calls

   Aim and Objectives

   Aim

   - To develop programs involving file management system calls

   Objectives

   At the end of this lab, the student will be able to

   - Use different file management system calls
   - Apply different system calls wherever required
   - Create C programs using file management system calls

2. Experimental Procedure

   i. Analyse the problem statement

   ii. Design an algorithm for the given problem statement and develop a flowchart/pseudo-code

   iii. Implement the algorithm in C language

   iv. Compile the C program

   v. Test the implemented program

   vi. Document the Results

   vii. Analyse and discuss the outcomes of your experiment

3. Questions

   Implement the following command in C

   Implement copy command (cp) to copy a file content to other file using file management system calls

4. Calculations/Computations/Algorithms

Case - 1

Copying single line from input file to a output text file **(Image 1)**

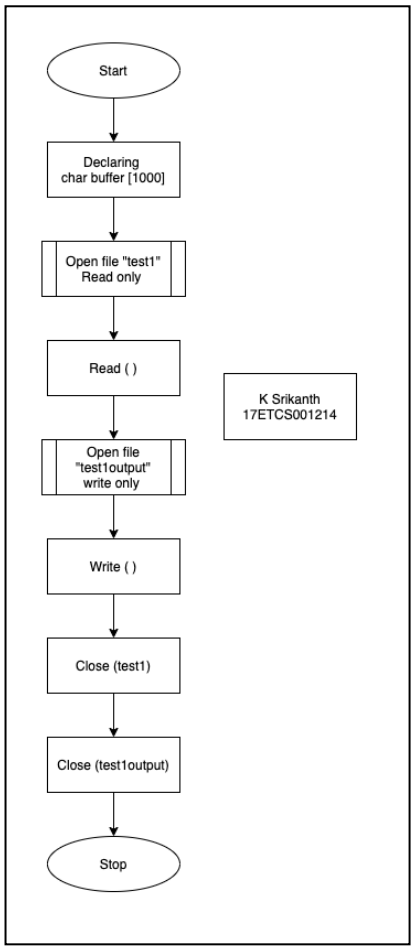

Image 1 Flowchart of C Program Case 1

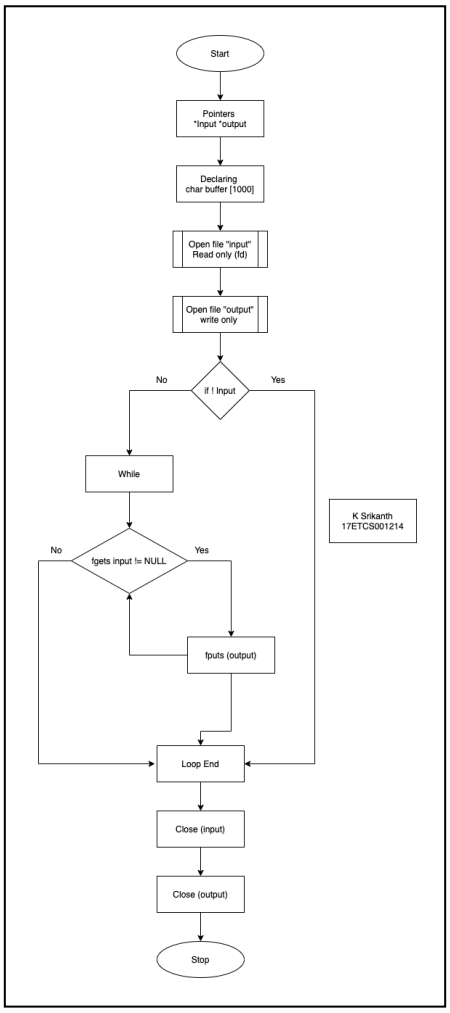

Image 2 Flowchart of C Program Case 2

Case - 2

Copying multiple line from input file to an output text file **(Image 2) .**

Case - 3

Copying from source file (input) when it doesn't exist to an output text file **(Image 3)** .

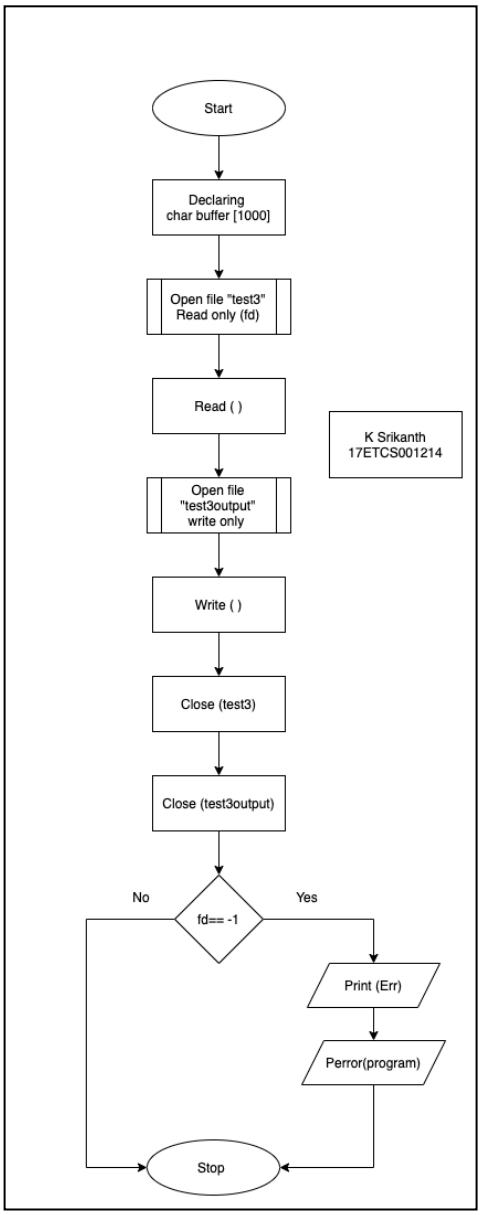

Image 3 Flowchart of C Program Case 4

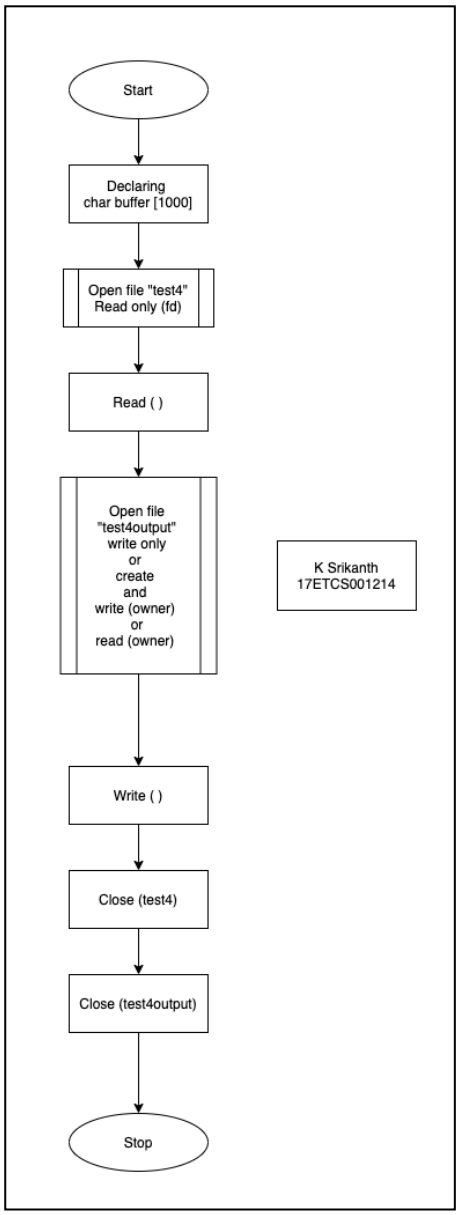

Image 4 Flowchart of C Program Case 4

Case - 4

Copying from input text file when destination (Output file) doesn't exist **(Image 4)** .

Case - 5

Copying input text file from a different directory to a different directory output text file **(Image 5) .**

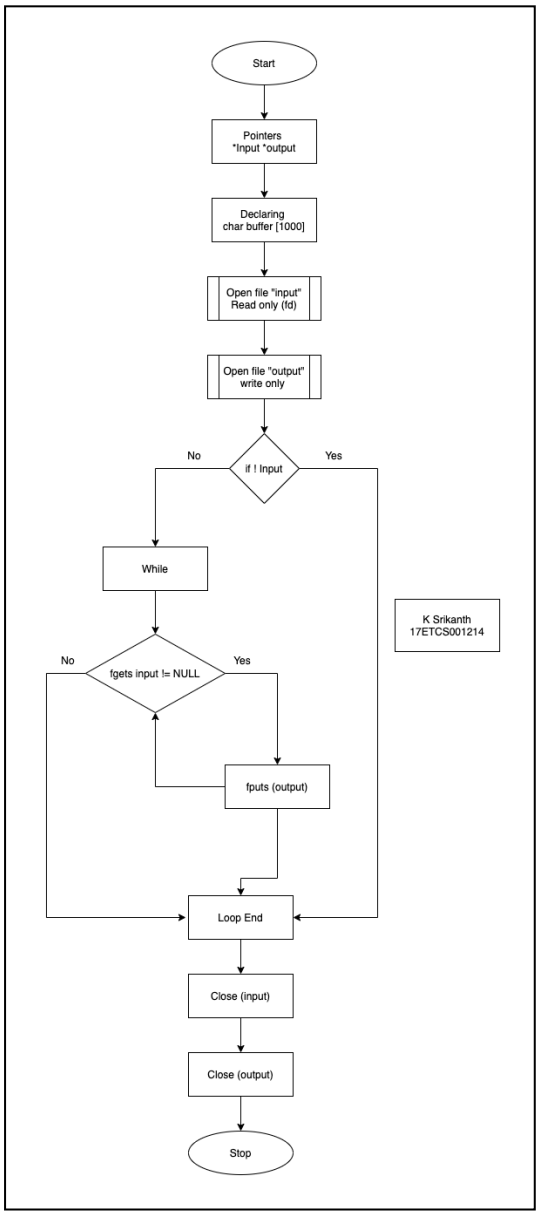

Image 5 Flowchart of C Program Case 5

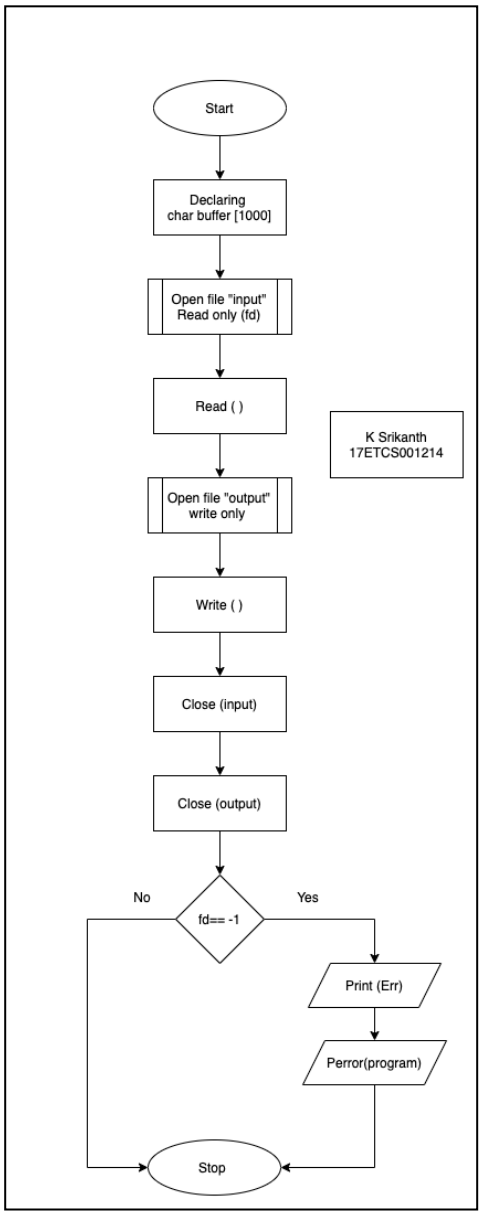

Image 6 Flowchart of C Program Case 6

Case - 6

Copying input text file from no source to output file no destination **(Image 6)** .

5. Presentation of Results

Case - 1

Copying single line from input file to a output text file

Code

```c
#include<stdio.h>
#include<fcntl.h>
#include <unistd.h>
#include <stdlib.h>
int main()
{
    printf("K Srikanth 17ETCS002124\n");
    printf("--------- CASE-1 ---------\n");
    printf("*************************\n");
    char buff[1000] ;
    int fd = open("/Users/srikanthkandarp/Desktop/Os_texts/Case_1/test1.txt", O_RDONLY );
    read(fd, buff, sizeof(buff));
    printf("fd = %d \n", fd);
    int fd1 = open("/Users/srikanthkandarp/Desktop/Os_texts/Case_1/test1output.txt", O_WRONLY );
    write(fd1, buff, sizeof(buff));
    close(fd);
    close(fd1);
}

```

Image 7 C Program to Copy single line from input file to an output text file

Outputs

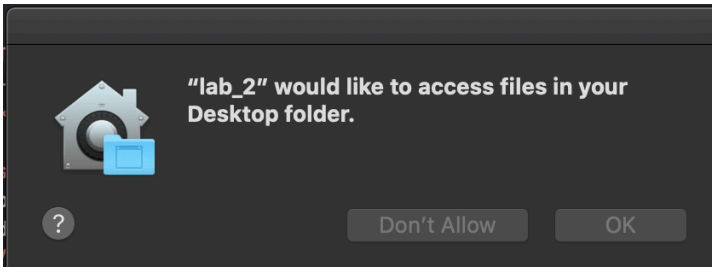

Image 8 System Pop-up to access desktop files.

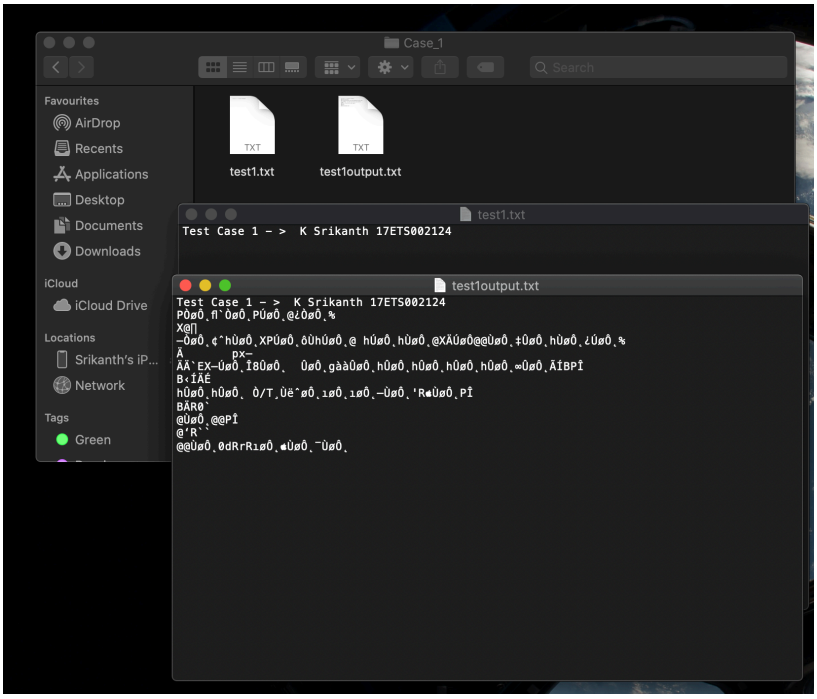

Image 9 C Program Output for copying single line from input file to a output text file

Case - 2

Copying multiple line from input file to a output text file

Code

```c
int main()
{
    printf("K Srikanth 17ETCS002124\n");
    printf("--------- CASE-2 ---------\n");
    printf("*************************\n");
    FILE *input_file, *output_file;
    char buffer[100];
    input_file = fopen("/Users/srikanthkandarp/Desktop/Os_texts/Case_2/test2.txt","r");
    output_file = fopen("/Users/srikanthkandarp/Desktop/Os_texts/Case_2/test2output.txt","w");

    if (!input_file)
        exit(1);

    while (fgets(buffer,100, input_file)!=NULL)
        fputs(buffer, output_file);


    fclose(input_file);
    fclose(output_file);
    return 0;;
}
```

Image 10 C Program to copy multiple line from input file to an output text files.

Outputs

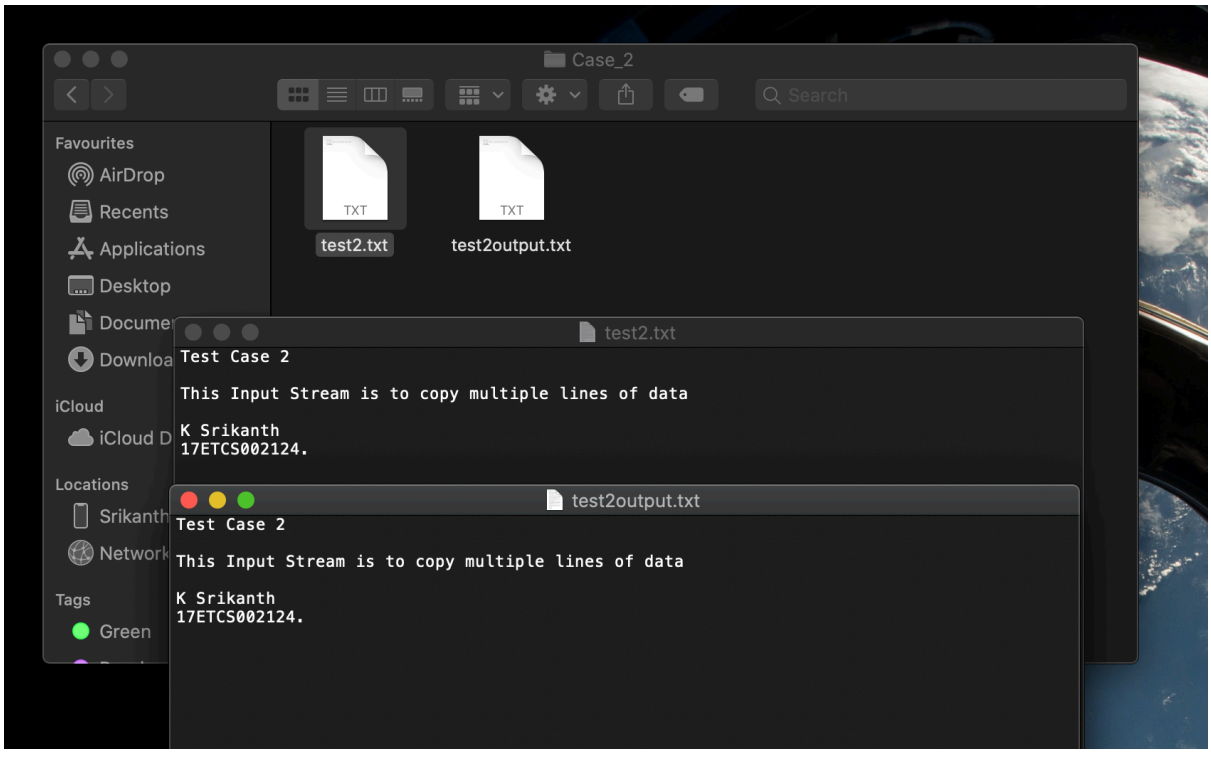

Image 11 C Program output for copying multiple line from input file to an output text files.

Case - 3

Copying from source file (input) when it doesn't exist to a output text file

Code

```c
extern int errno;
int main()
{
    printf("K Srikanth 17ETCS002124\n");
    printf("--------- CASE-3 ---------\n");
    printf("*************************\n");
    char buff[15];
    int fd = open("/Users/srikanthkandarp/Desktop/Os_texts/Case_3/test3.txt", O_RDONLY );
    int fd1 = open("/Users/srikanthkandarp/Desktop/Os_texts/Case_3/test3output.txt", O_WRONLY );
    read(fd, buff, sizeof(buff));
    write(fd1, buff, sizeof(buff));
    close(fd);
    close(fd1);
    if (fd ==-1)
    {
        printf("C Program Error Number % d\n", errno);
        perror("Program");
    }
    return 0;
}
```

Image 12 C Program to copy source file (input) when it doesn't exist to a output text file

Output

```
K Srikanth 17ETCS002124
--------- CASE-3 ---------
*************************
C Program Error Number  9
Program: Bad file descriptor
Program ended with exit code: 0
```

Image 13 C Program output to copy source file (input) when it doesn't exist to a output text file

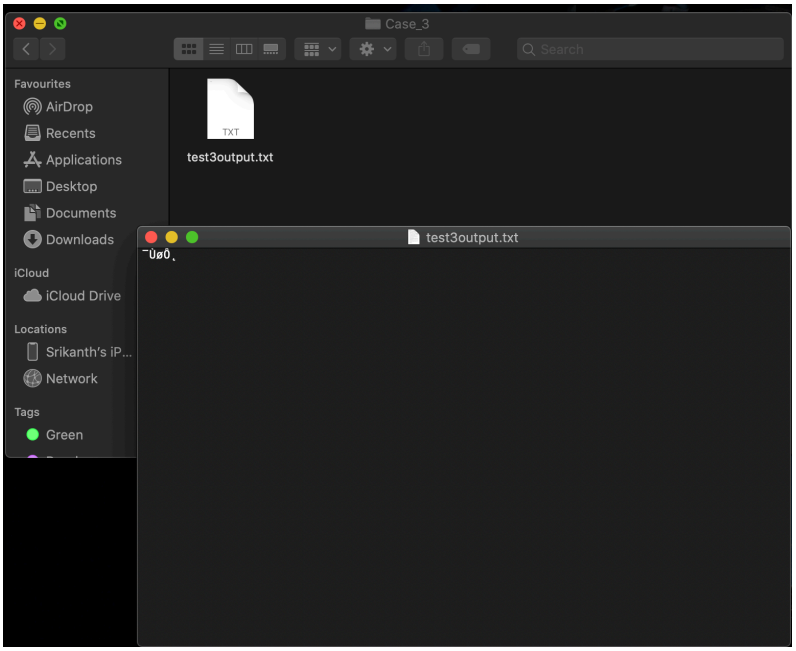

Image 14 C Program output to copy source file (input) when it doesn't exist to a output text file

Case - 4

Copying from input text file when destination (Output file) doesn't exist

Code

```c
extern int errno;
int main()
{
    printf("K Srikanth 17ETCS002124\n");
    printf("--------- CASE-4 ---------\n");
    printf("**************************\n");
    char buff[1000] ;
    int fd = open("/Users/srikanthkandarp/Desktop/Os_texts/Case_4/test4.txt", O_RDONLY );
    read(fd, buff, sizeof(buff));
    int fd1 = open("/Users/srikanthkandarp/Desktop/Os_texts/Case_4/test4output.txt", O_WRONLY | O_CREAT, S_IWRITE | S_IWUSR);
    write(fd1, buff, sizeof(buff));

    close(fd);
    close(fd1);

    return 0;
}
```

Image 15 C Program to copy input text file when destination (Output file) doesn't exist

Output

Input text file **"test4.txt".**

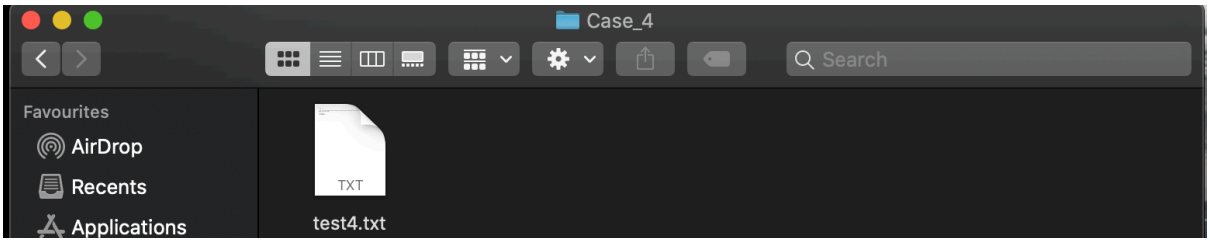

Image 16 User Folder before running the code (Image 15).

After running the program you get an output file **"test4output.txt"**

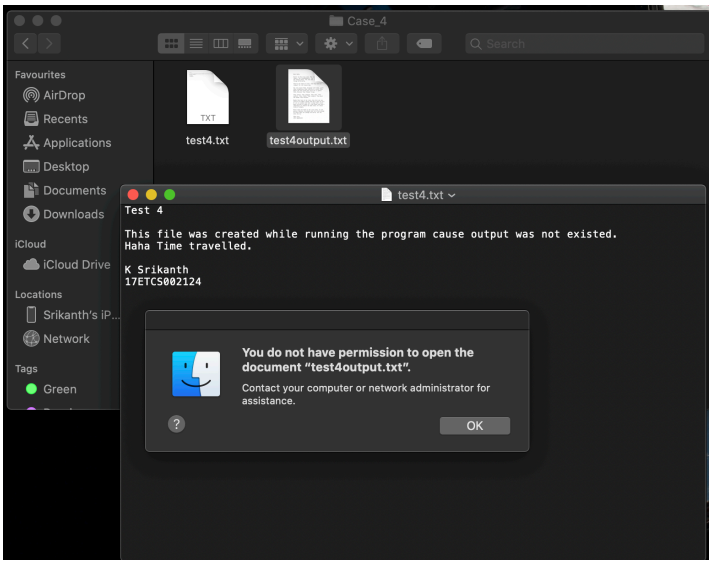

Image 17 C Program output created but can't be accessed.

Now you don't have permission to open it cause it's in write mode so we change the permissions of the file to public.

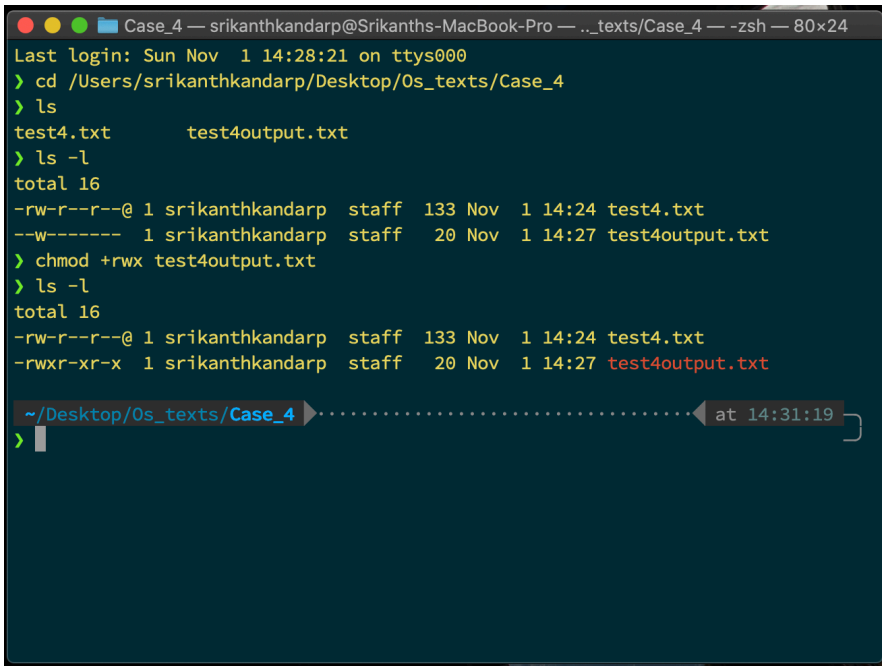

Image 18 User Terminal Overriding the file restrictions

Voila, now we can open the text file "test4output.txt".

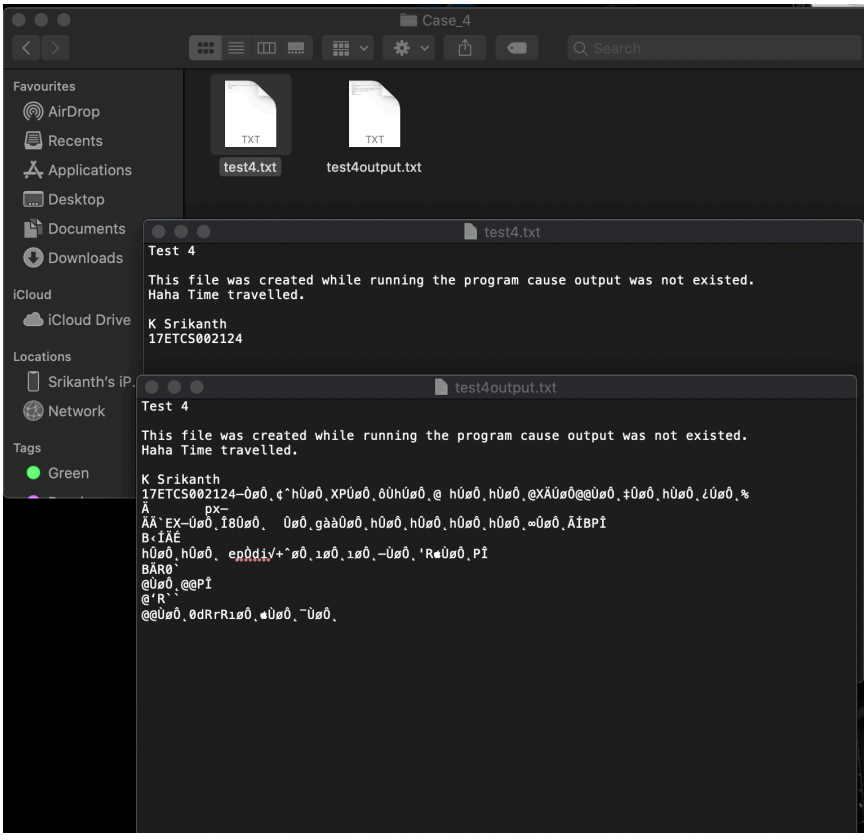

Image 19 C Program output for copying input text file when destination (Output file) doesn't exist

Case - 5

Copying input text file from a different directory to a different directory output text file

Code

```c
int main()
{
    printf("K Srikanth 17ETCS002124\n");
    printf("---------- CASE-5 ----------\n");
    printf("**************************\n");
    FILE *input_file, *output_file;
    char buffer[100];
    input_file = fopen("/Users/srikanthkandarp/Desktop/Os_texts/Case_5/test5.txt","r");
    output_file = fopen("/Users/srikanthkandarp/Documents/Case_5/test5output.txt","w");
    if (!input_file)
        exit(1);
    while (fgets(buffer,100, input_file)!=NULL)
        fputs(buffer, output_file);
    fclose(input_file);
    fclose(output_file);
    return 0;;
}
```

Image 20 C Program to copy input text file from a different directory to a different directory output text file

Output

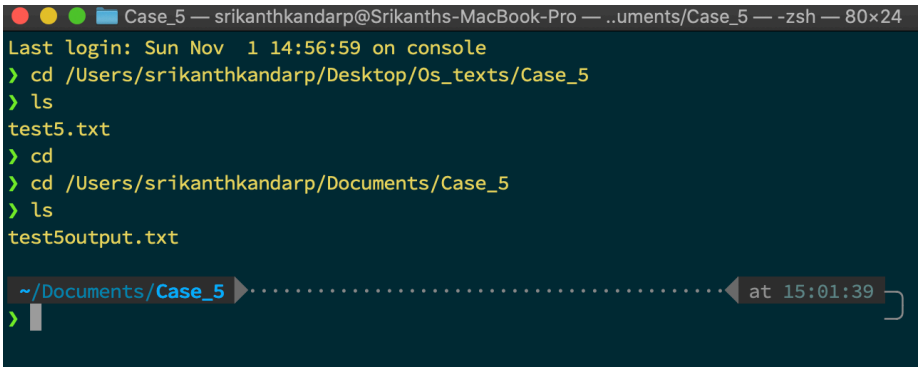

Image 21 User Terminal Displaying that two files are in different directories.

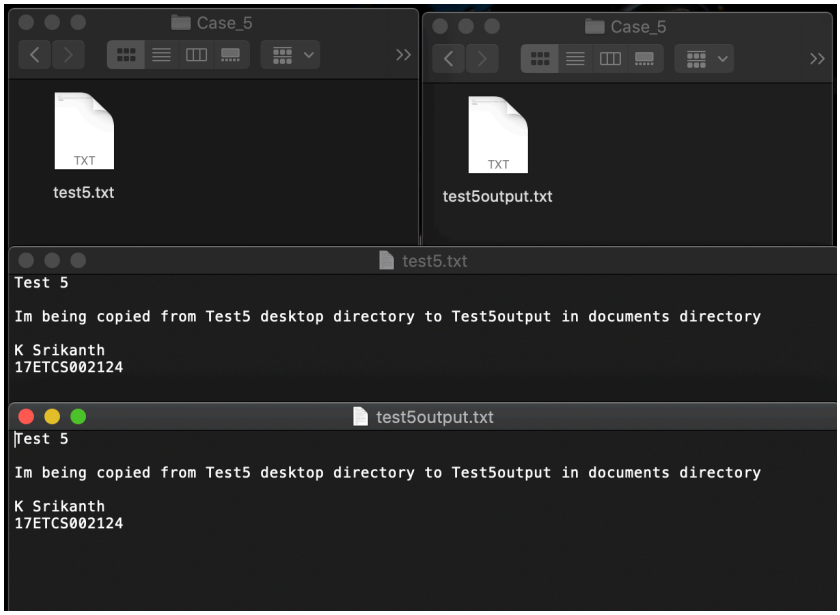

Image 22 C Program output for copying input text file from a different directory to a different directory output text file

Case - 6

Copying input text file from no source to an output text file no destination.

Code

```c
extern int errno;
int main()
{
    printf("K Srikanth 17ETCS002124\n");
    printf("--------- CASE-6 ---------\n");
    printf("**************************\n");
    char buff[15];
    int fd = open("/Users/srikanthkandarp/Desktop/Os_texts/Case_6/test6.txt", O_RDONLY );
    int fd1 = open("/Users/srikanthkandarp/Desktop/Os_texts/Case_6/test6output.txt", O_WRONLY );
    read(fd, buff, sizeof(buff));
    write(fd1, buff, sizeof(buff));
    close(fd);
    close(fd1);
    if (fd ==-1)
    {
        printf("C Program Error Number % d\n", errno);
        perror("Program");
    }
    return 0;
}
```

Image 23 C Program to copy input text file from no source to an output text file no destination.

Outputs

```
K Srikanth 17ETCS002124
--------- CASE-6 ---------
**************************
C Program Error Number  9
Program: Bad file descriptor
Program ended with exit code: 0
```

Image 23 C Program output for copying input text file from no source to an output text file no destination.

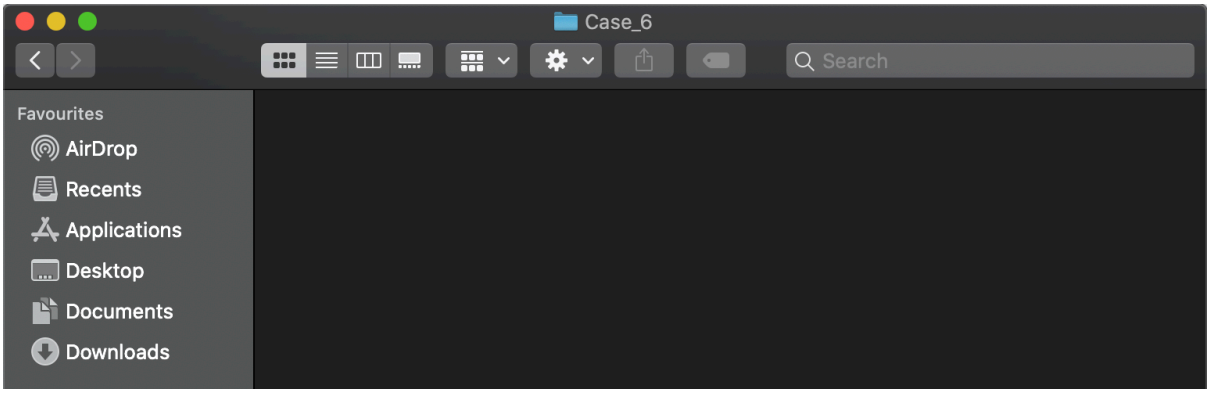

Image 25 User Folder stating that files doesn't exist.

6. Analysis and Discussions

Case 1

In this case we are supposed to copy single line from input file (*test1) to output file (*test1output) so first we create a buffer with **1000 characters** then we will be opening a input file using **open( )** function in a read only mode and then we will read the whole file using **read ( )** function where the arguments are first the file then your buffer and last argument is size of the buffer. After doing this part now we write into a file for output so we open a file using **open( )** function in write mode and write it using. **write ( )** function here the arguments are first the file then your buffer and last argument is size of the buffer at last we close the both files using **close ( )** function.

Case 2

In this case we are supposed to copy multiple lines from input file (*test2) to output file (*test2output) so first we create a buffer with **1000 characters** then we will be opening a input file using **open( )** function in a read only mode and we open a new output file using **open( )** function in write mode and write it using now we have a condition if the input file doesn't exist then we exit the program to read what's on the file we use while loop with condition **fgets ( )** with arguments first being buffer then an integer value stating how much characters you wanna read and finally your input file which we compared it not being null if this condition satisfy then we write into the output file with **fputs ( )** function with arguments buffer and output file at last we close the both files using **close ( )** function

Case 3

In this we are supposed to from source file (input) when it doesn't exist to an output text file so it should return an error so we firstly define a external variable **err type INT.** we create a buffer with **15 characters** then we will be opening a input file using **open( )** function in a read only mode and we open a new output file using **open( )** in write only mode and we read the input file using **read()** function and using **write() f**unction we will be writing the data into the output file at last we close the both files using **close ( )** function. But as we know that output file doesn't exist so we check if out input file is equal to **"-1" using a if statement** if yes then we print what type of error number it is using err variable that we declared earlier and also we will be printing **perror("Program") to know what kind of error it is.**

Case 4

In this case we are supposed to copy single line from input file **(*test4)** to output file **(*test4output)** which doesn't exist yet. so first we create a buffer with 1000 characters then we will be opening a input file using **open( )** function in a read only mode and then we will read the whole file using **read ( )** function after doing this part now we have to create a new file so we use **open( )** function to open the file in write only if it doesn't exist then we create a new file with the name that user specified while opening the file and we will be giving it **write user access** using **write ( )** function we write the data Into the file that we created now at last we close the both files using **close ( )** function. Now when you run the code you get a output file which can't be opened from a user access **(Image 17)** now we have to give permissions so we open that file directory in terminal and typing this command **"chmod +rwx filename"(Image 18)** so this gives read write and execute access to the user ,Now output can be seen in **image 19.**

Case 5

In this case we are supposed to copy multiple lines from input file (*test5) from a different directory to output file (*test5output) in different directory so first we create a buffer with **1000 characters** then we will be opening a input file using **open( )** function in a read only mode and we open a new output file using **open( )** function in write mode and write it using now we have a condition if the input file doesn't exist then we exit the program to read what's on the file we use while loop with condition **fgets ( )** we compared it not being null if this condition satisfy then we write into the output file with **fputs ( )** function at last we close the both files using **close ( )** function

Case 6

in this we are supposed to from source file (input) when it doesn't exist to an output text file which also doesn't exist so it should return an error so firstly we define a external variable **err type INT.** we create a buffer with **15 characters** then we will be opening a input file using **open( )** function in a read only mode and we open a new output file using **open( )** in write only mode and we read the input file using **read()** function and using **write() f**unction we will be writing the data into the output file at last we close the both files using **close ( )** function. But as we know that output file doesn't exist so we check if out input file is equal to **"-1" using a if statement** if yes then we print what type of error number it is using err variable that we declared earlier and also we will be printing **perror("Program") to know what kind of error it is.**

7. Conclusions

File management calls is a systematic way of opening, creating, reading, writing and closing of a file through system calls. This way we can create a program that can open, read, and write on files. We opened a file in write-mode and wrote the data read from the other file that has been opened in read-mode, thus copying text from one file to other.

8. Comments

1. Limitations of Experiments

The program uses the fore-mentioned file names, instead of asking the user to locate the file in the system.

2. Limitations of Results

Program doesn't display the number of characters copied, to the user, User unaware of how copy process is going

3. Learning happened

File management calls and applying them to access files in the system and work on them.