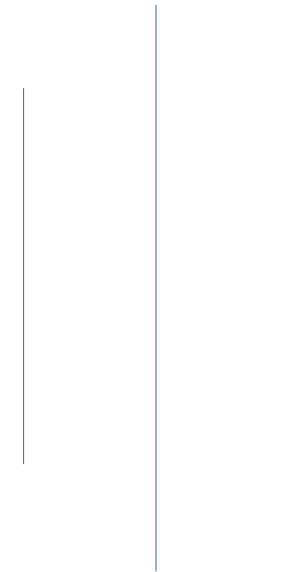


INSTITUTE OF ENGINEERING

Pulchowk Campus, Lalitpur



Subject: C Programming

Lab Report 12

Title: **Files Handling**

Submitted by:
Susheel Thapa 077BCT090

Submitted to:
Department of Electronics and
Computer Engineering

Checked by

Content of Lab Report:

Background Information

C Programming

Editor Used

Compiler

Files

Code and Output (Analysis)

Source Code

Output

Analysis

Conclusion

Background Information

What is C Programming?

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language.

Editor

Here, I have used Visual Studio Code as my editor. You can download the editor from [Download Visual Studio Code - Mac, Linux, Windows](#). Select your operating system and download it.

Compiler

Here, I have used **gcc** as my compiler provided by MinGWw64. You can download it via [Download MinGW-w64 - for 32 and 64 bit Windows](#) from SourceForge.net. Your download will start automatically. Run the downloaded .exe file. After, you have install MinGW-w64, you need to configure it.

1. In the Windows search bar, type 'settings' to open your Windows Settings.
2. Search for Edit environment variables for your account.
3. Choose the Path variable and then select Edit.
4. Select New and add the Mingw-w64 destination folder path to the system path. The exact path depends on which version of Mingw-w64 you have installed and where you installed it. If you used the settings above to install Mingw-w64, then add this to the path: **C:\Program Files\mingw-w64\x86_64-8.1.0-posix-seh-rt_v6-rev0\mingw64\bin.**
5. Select OK to save the updated PATH. You will need to reopen any console windows for the new PATH location to be available.

File:

Many applications require that information be written to or read from an auxiliary memory device. Such information is stored on the memory device in the form of a data file. The data files allow us to store information permanently and to access and alter that information whenever necessary.

Opening a file:

Before performing any input / output operation, file must be opened. While opening file, the following must be specified:-

- i. The name of file.
- ii. The manner in which it should be opened (that for reading ,writing ,both reading and writing ,appending at the end of file, overwriting the file)
- iii. when working with a stream oriented data file ,the first step is to establish a buffer area, where information is temporary stored while being transferred between the computers memory and data file .the buffer area is established by writing
`FILE *fpt;`

where File is a special structure type establishes the buffer area and ptvar is a pointer variable that indicates the beginning of the buffer area the library function `fopen` is used to open a file .This function is used to open a file .This function is typically written as

`fpt=fopen(file name, file type);`

where file name and file type are strings that represent the name of the data file and the manner in which the data file will be utilized.

Finally, a file can be closed at the end of the program. This can be accomplished with the library function `fclose`. The syntax is simply,
`fclose(fpt)`

Processing a file:

Most data file application requires that a data file be altered as it is being processed. For example, in an application involving the processing of customer records, it may be desirable to add new records to the file. To delete the existing records, to modify the contents or to rearrange the records.

File types:

DOS treats files in two different ways that as binary or text file. Files almost all UNIX systems do not make any distinction between the two. If a file is specified as the binary type, the file I /O functions do not interpret the contents of the file when they read from, or write to the file. But if the file is specified as the text type, the file I / O functions interpret the contents of the file. The basic differences in these two types of files are:

- i. ASCII 0*1a is considered as end of file character by the file I /O functions when reading from a text file and they assume that the end of the file has been reached.
- ii. In case of DOS, a new line is stored as the sequence 0*0a on the disk in case text files.

1. Write a program to read a year from the user and write it in the file if the year is leap year

Source Code

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

void leapYear(unsigned int year)
{

    unsigned int divisible_by_four, divisible_by_hundered, divisible_by_four_hundered;

    /*Below if check whether the year enter in after 1582 or not*/
    if (year >= 1582)
    {

        /*Needed tool for checking the variable*/
        divisible_by_four = year % 4;
        divisible_by_hundered = year % 100;
        divisible_by_four_hundered = year % 400;

        /*Check the condition to be a leap year*/
        if (divisible_by_four == 0)
        {
            if (!(divisible_by_hundered == 0) || divisible_by_four_hundered == 0)
            {
                printf("It is leap year.\n");

                printf("Writing to file 'Program01.txt'.\n");

                /*Writing data to files*/
                FILE *ptr_file; /*Files pointer*/

                ptr_file = fopen("Program01.txt", "a"); /*Pointing to the file*/
```

```

fprintf(ptr_file, "%d is leap year.\n", year);

fclose(ptr_file); /*Close the files pointer*/
printf("\n\nDONE...");
}
else
{
printf("It isn't leap year.");
}
}
else
{
printf("It isn't leap year.");
}
}
else
{
printf("You didn't give input as per asked in questions.");
}
}
int main()
{
    unsigned int year;

    system("cls");

    printf("Year(after 1582): ");
    scanf("%u", &year);

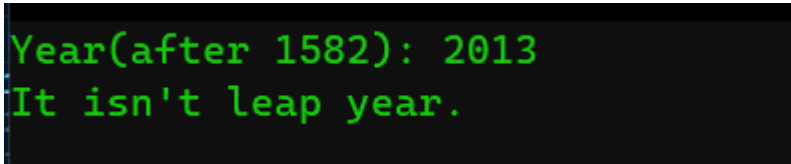
    leapYear(year);

    getch();
    return 0;
}

```

Output:

Screenshot of only one output



```

Year(after 1582): 2013
It isn't leap year.

```

Files data with various input.

```
2000 is leap year.  
2004 is leap year.  
2000 is leap year.
```

2.WAP to read words from the user until user hits 'NO' and write them to a file if the word is vowel free. Display the content in the files

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

int vowelCheck(char *word)
{
    "Check whether the word contain the vowel letter of not";

    char vowel[] = "aeiouAEIOU";

    while (*word != '\0')
    {
        for (int i = 0; i < 10; i++)
        {
            if (*word == vowel[i])
            {
                return 1;
            }
        }
        word++;
    }
    return 0;
}

int main()
```

```

{
    char word[100], file_content;
    FILE *ptr_word;
    int vowel_check_result;

    system("cls");

    do
    {
        printf("Words: ");
        scanf("%s", word);

        vowel_check_result = vowelCheck(word);

        if (vowel_check_result == 0)
        {
            ptr_word = fopen("Program02.txt", "a");

            fprintf(ptr_word, "~%s\n", word);

            fclose(ptr_word);
        }
    } while (strcmpi(word, "NO") != 0);

    /*Reading and printing the content of file*/
    printf("Content of file:\n");
    ptr_word = fopen("Program02.txt", "r");
    do
    {
        file_content = fgetc(ptr_word);

        printf("%c", file_content);
    } while (file_content != EOF);

    fclose(ptr_word);

    /*Deleting the content of files*/
    ptr_word = fopen("Program02.txt", "w");
    fclose(ptr_word);

    getch();
    return 0;
}

```


Output:

```
Words: sus
Words: he
Words: is
Words: whnn
Words: uy
Words: re
Words: no
Content of file:
~whnn
```

4. Write a program to open a new file, read roll number, name, address and phone number of students until user says "no" after reading the data write it to file then display the contents of files

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

int main()
{
    FILE *in, *out;
    char roll_number[100], name[100], address[100];
    long long int phone_number;

    // system("cls");
    in = fopen("Program03.txt", "a");
```

```

while (1)
{
// fflush(stdin);

printf("Roll Number: ");
scanf(" %[^'\n']", roll_number);

// printf("%s", roll_number);
// exit(0);

if (strcmp(roll_number, "no") == 0)
{
break;
}

printf("Name: ");
scanf(" %[^'\n']", name);

    printf("Address: ");
    scanf(" %[^'\n']", address);

    printf("Phone Number: ");
    scanf("%lld", &phone_number);
    fprintf(in, "%30s%30s%30s%15lld\n", roll_number, name, address, phone_number);
}

fclose(in);

system("cls");

out = fopen("Program03.txt", "r");

printf("\n%30s%30s%30s%15s\n", "Roll Number", "Name", "Address", "Phone Number");

while (1)
{
    char ch = fgetc(out);
    if (ch == EOF)
    {
        break;
    }
    printf("%c", ch);
}
fclose(out);

```

```

    getch();
    return 0;
}

```

Output:

Roll Number	Name	Address	Phone Number
1	Ram	Chitwan	9800000000
2	Shyam	Lamjung	9811111111
3	Hari	Nepaljung	9822222222
4	Geeta	Hetauda	9833333333
5	Sita	Kathmandu	9844444444
6	Balram	Sagarmatha	9855555555
7	Hanuman	Parbat	9866666666
8	EarthQuake	Nepal	9877777777
9	Laxman	Jhapa	9888888888
090	Susheel	Bharatpur-05, Chitwan	9816156426
023	Something	Parsa	9811564794
12	Noone	Heaven	9874562135

Data in files

1	Ram	Chitwan	9800000000
2	Shyam	Lamjung	9811111111
3	Hari	Nepaljung	9822222222
4	Geeta	Hetauda	9833333333
5	Sita	Kathmandu	9844444444
6	Balram	Sagarmatha	9855555555
7	Hanuman	Parbat	9866666666
8	EarthQuake	Nepal	9877777777
9	Laxman	Jhapa	9888888888
090	Susheel	Bharatpur-05, Chitwan	9816156426
023	Something	Parsa	9811564794
12	Noone	Heaven	9874562135

4. Write a program to input name, roll, address, telephone number and score of a student. Store the contents of the person in file first.txt After that, copy the contents of first.txt to second.txt and display the contents of second.txt In this program, you should use the text

files.[You can use structure for data handling]

Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

struct Data
{
    char name[100];
    int roll_number;
    char address[100];
    long long int telephone_number;
    float score;
};

int main()
{
    char ch;

    /*Structure variable*/
    struct Data user;

    /*Creating Files pointer*/
    FILE *in_first;
    FILE *out_first;
    FILE *in_second;
    FILE *out_second;

    system("cls");

    /*Taking input from user*/
    printf("Name: ");
    scanf("%s", user.name);

    printf("Roll Number: ");
    scanf("%d", &user.roll_number);

    printf("Address: ");
    scanf("%s", user.address);

    printf("Telephone Number:");
```

```

scanf("%lld", &user.telephone_number);

printf("IOE score: ");
scanf("%f", &user.score);

/*Opening first.txt in write mode and write the inputted data*/
in_first = fopen("First.txt", "w");
fprintf(in_first, "%s %d %s %lld %.2f", user.name, user.roll_number, user.address,
user.telephone_number, user.score);
fclose(in_first);

/*Opening First.txt in read mode and Second.txt in write mode*/
out_first = fopen("First.txt", "r");
in_second = fopen("Second.txt", "w");

/*Coping character from first.txt to second.txt character by character*/
while (1)
{

    /*Taking a character*/
    ch = fgetc(out_first);

    /*Condition if the character in the files has reached to end*/
    if (ch == EOF)
    {
        break;
    }
    fputc(ch, in_second); /*First argument character , Second argument File pointer*/
}

/*Closing the files*/
fclose(out_first);
fclose(in_second);

/*Opening the second.txt in read mode and reading charcter by character and printing*/
out_second = fopen("Second.txt", "r");
while (1)
{
    ch = fgetc(out_second);
    if (ch == EOF)
    {
        break;
    }
    printf("%c", ch);
}

```

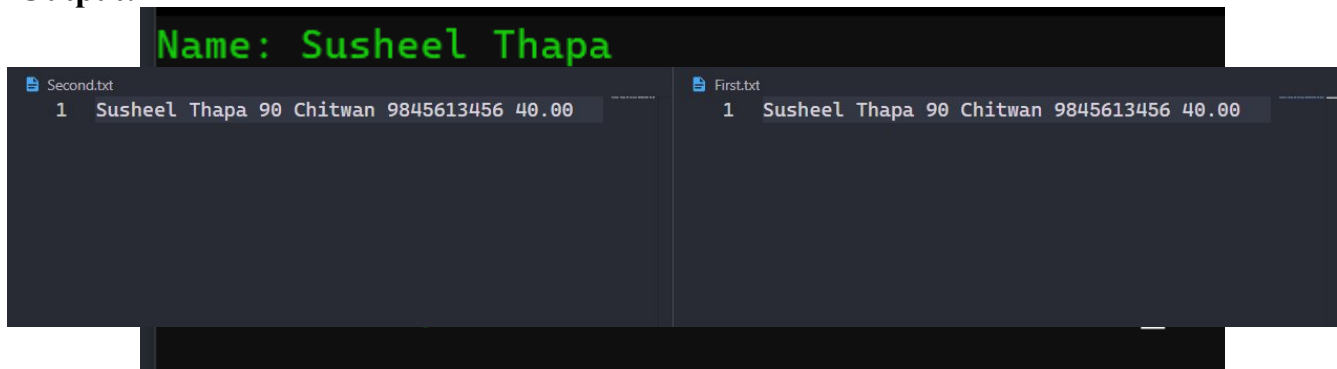
```

fclose(out_second);

getch();
return 0;
}

```

Output:



5. Modify question number 4, using binary file. You should use `fwrite()` function to write data into a text file and `fread()` function to read the file. [here, you must use structure for data handling]

Source Code:

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>

/*Creating a structure*/
struct Data
{
    char name[100];
    int roll_number;
    char address[100];
    long long int telephone_number;
    float score;
};

int main()
{
    /*Structure variable*/

```

```

struct Data user;

/*Creating Files pointer*/
FILE *in_first;
FILE *out_first;
FILE *in_second;
FILE *out_second;

system("cls");

/*Taking input from user*/
printf("Name: ");
scanf("%s", user.name);

printf("Roll Number: ");
scanf("%d", &user.roll_number);

printf("Address: ");
scanf("%s", user.address);

printf("Telephone Number:");
scanf("%lld", &user.telephone_number);

printf("IOE score: ");
scanf("%f", &user.score);

/*Opening first.txt in write binary mode and write the inputted data*/
in_first = fopen("First_05.txt", "wb");

/*Writing data that has been taken from user*/
fwrite(&user, sizeof(user), 1, in_first);

/*Closing the file*/
fclose(in_first);

/*Opening First.txt in read binary mode and Second.txt in write binary mode*/
out_first = fopen("First_05.txt", "rb");
in_second = fopen("Second_05.txt", "wb");

/*Reading a data and writing the read data to another file. Copying data from one file to
another*/
fread(&user, sizeof(user), 1, out_first);
fwrite(&user, sizeof(user), 1, in_second);

/*Closing the files*/

```

```

fclose(out_first);
fclose(in_second);

/*Opening the second.txt in read binary mode*/
out_second = fopen("Second_05.txt", "rb");

/*Reading the data*/
fread(&user, sizeof(user), 1, out_first);

/*Printing the data that has been read*/
printf("%s\n", user.name);
printf("%s\n", user.address);
printf("%lld\n", user.telephone_number);
printf("%d\n", user.roll_number);
printf("%.2f\n", user.score);
fclose(out_second);

getch();
return 0;
}

```

Output:

```

Name: Susheel Thapa
Roll Number: 156
Address: Bharatpur-05, Chitwan
Telephone Number:9845612378
IOE score: 50
Susheel Thapa
Bharatpur-05, Chitwan
9845612378
156
50.00

```


Analysis and Discussion

Here we have written a lot of programs and run it many times. Through this, we are able to know the respective syntax of the element like `fprintf()`, `fscanf()`, `fgets ()`, `fgetc()`, `fputc()` and many more.

Moreover, it provides us brief information about how can we write into files read into files and many more.

Conclusion

In nut shell we learn about file handling