EXPERIMENT9: FILE HANDLING IN C

Experiment 1: Creating a File and Writing Text

Aim

To create a new file and write text into it using file handling functions in C.

Algorithm

1. Declare a file pointer
2. Open file in write mode
3. Write text using `fprintf()`
4. Close file

Pseudocode

```
DECLARE file pointer
OPEN file in write mode
WRITE text into file
CLOSE file
```

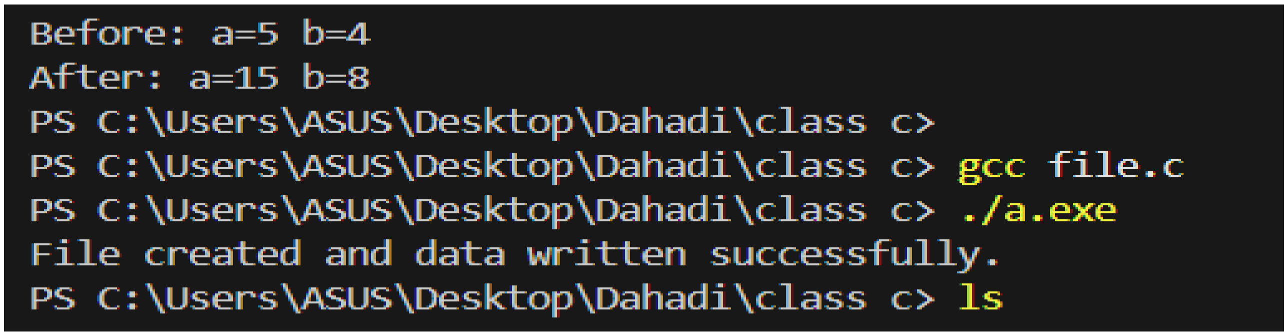
Flowchart (ASCII)

```
Start
   ↓
Open File (Write Mode)
   ↓
Write Text
   ↓
Close File
   ↓
 End
```

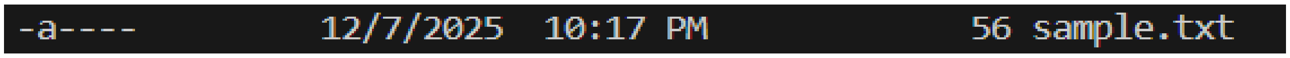
C Program

```c
#include <stdio.h>

int main() {
    FILE *fp;

    fp = fopen("sample.txt", "w");

    if (fp == NULL) {
        printf("File cannot be created\n");
        return 1;
    }

    fprintf(fp, "Welcome to File Handling in C.\n");
    fprintf(fp, "This is a sample file.\n");

    fclose(fp);

    printf("File created and data written successfully.\n");
    return 0;
}
```

Output



```
Before: a=5 b=4
After: a=15 b=8
PS C:\Users\ASUS\Desktop\Dahadi\class c>
PS C:\Users\ASUS\Desktop\Dahadi\class c> gcc file.c
PS C:\Users\ASUS\Desktop\Dahadi\class c> ./a.exe
File created and data written successfully.
PS C:\Users\ASUS\Desktop\Dahadi\class c> ls
```



```
-a----          12/7/2025  10:17 PM            56 sample.txt
```

Conclusion

The program successfully created a new file and wrote text into it using file handling functions.

Experiment 2: Reading File Content Character by Character

Aim

To read the contents of an existing file character by character and display it on the console.

Algorithm

1. Open file in read mode
2. Read characters using `fgetc()`
3. Display each character
4. Close file

Pseudocode

```
OPEN file in read mode
WHILE character != EOF
    READ character
    DISPLAY character
CLOSE file
```

Flowchart (ASCII)

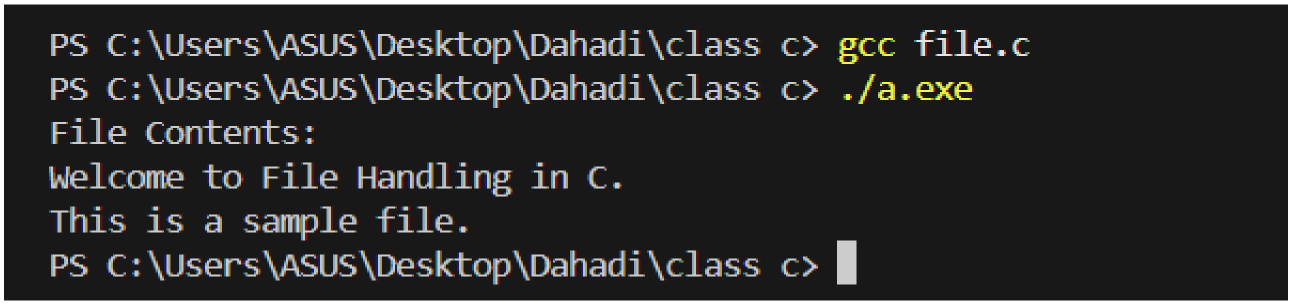
```
Start
   ↓
Open File (Read Mode)
   ↓
Read Character
   ↓
Is EOF?
   ↓ No
Display Character
   ↓
Repeat
   ↓ Yes
Close File
   ↓
```

```
 End
```

C Program

```
#include <stdio.h>

int main() {
    FILE *fp;
    char ch;

    fp = fopen("sample.txt", "r");

    if (fp == NULL) {
        printf("File cannot be opened\n");
        return 1;
    }

    printf("File Contents:\n");
    while ((ch = fgetc(fp)) != EOF) {
        putchar(ch);
    }

    fclose(fp);
    return 0;
}
```

Output



```
PS C:\Users\ASUS\Desktop\Dahadi\class c> gcc file.c
PS C:\Users\ASUS\Desktop\Dahadi\class c> ./a.exe
File Contents:
Welcome to File Handling in C.
This is a sample file.
PS C:\Users\ASUS\Desktop\Dahadi\class c> 
```

Conclusion

The program read and displayed file contents character by character successfully.

Experiment 3: Reading File Content Line by Line

Aim

To read the contents of a file line by line and display each line on the console.

Algorithm

1. Open file in read mode
2. Read lines using `fgets()`
3. Display each line

4. Close file

Pseudocode

```
OPEN file in read mode
WHILE fgets reads a line
    DISPLAY line
CLOSE file
```

Flowchart (ASCII)

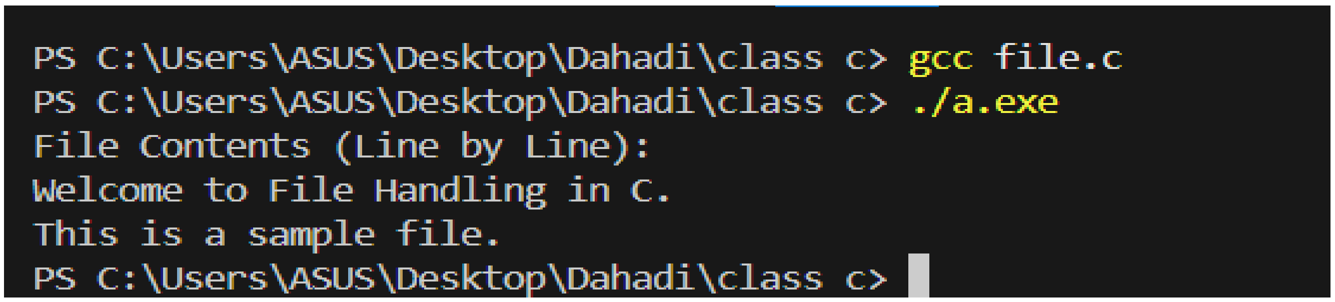
```
Start
   ↓
Open File
   ↓
Read Line
   ↓
Is EOF?
   ↓ No
Display Line
   ↓
Repeat
   ↓ Yes
Close File
   ↓
 End
```

C Program

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

```c
int main() {
    FILE *fp;
    char line[100];

    fp = fopen("sample.txt", "r");

    if (fp == NULL) {
        printf("File cannot be opened\n");
        return 1;
    }

    printf("File Contents (Line by Line):\n");

    while (fgets(line, sizeof(line), fp) != NULL) {
        printf("%s", line);
    }

    fclose(fp);
    return 0;
}
```

Output



```
PS C:\Users\ASUS\Desktop\Dahadi\class c> gcc file.c
PS C:\Users\ASUS\Desktop\Dahadi\class c> ./a.exe
File Contents (Line by Line):
Welcome to File Handling in C.
This is a sample file.
PS C:\Users\ASUS\Desktop\Dahadi\class c>
```

Conclusion

The program successfully read and displayed file contents line by line using file handling functions in C.