

Don Bosco Institute of Technology, Kulra(W)
Department of Computer Engineering
CSL601: System Programming and Compiler Construction Lab2022-23

Experiment No.	: 01
Experiment Title	<p>Task A:</p> <p>Write a program that reads a text file and count number of lines, blank spaces, occurs and displays the results in another text file.</p> <p>Task B:</p> <p>Write Assembly language program on 8086 emulator machine for factorial of a number</p>
Student Name	Ronak Surve
Roll No.	64
Objectives :	<p>1) Students will be able to revisit the concepts of assembly language programming</p> <p>2) Students will be able to perform file handling operations.</p>
Theory /Algorithm :	<p><u>File Handling using Python</u></p> <p>Python supports file handling and allows users to handle files i.e., to read and write files, along with many other file handling options, to operate on files. The concept of file handling has stretched over various other languages, but the implementation is either complicated or lengthy, but like other concepts of Python, this concept here is also easy and short. Python treats files differently as text or binary and this is important. Each line of code includes a sequence of characters and they form a text file. Each line of a file is terminated with a special character, called the EOL or End of Line characters like comma {,} or newline character. It ends the current line and tells the interpreter a new one has begun. Let's start with the reading and writing files.</p> <p>Working of open() function</p> <p>Before performing any operation on the file like reading or writing, first, we have to open that file. For this, we should use Python's inbuilt function open() but at the time of opening, we have to specify the mode, which represents the purpose of the opening file.</p> <p>f = open(filename, mode)</p> <p>Where the following mode is supported:</p> <p>r: open an existing file for a read operation.</p> <p>w: open an existing file for a write operation. If the file already contains some data then it will be overridden but if the file is not present then it creates the file as well.</p> <p>a: open an existing file for append operation. It won't override existing data.</p> <p>r+: To read and write data into the file. The previous data in the file will be overridden.</p> <p>w+: To write and read data. It will override existing data.</p> <p>a+: To append and read data from the file. It won't override existing data.</p>
Program Code:	<p>Task A:</p> <pre>#include <stdio.h> #include <stdlib.h></pre>

```

int WinMain() {
    FILE *input, *output;
    char fileName[100];
    int lines = 0, spaces = 0, chars = 0;
    char c;

    // Open input file for reading
    printf("Enter input file name: ");
    scanf("%s", fileName);
    input = fopen(fileName, "r");
    if (input == NULL) {
        printf("Error opening input file!\n");
        return 1;
    }

    // Open output file for writing
    printf("Enter output file name: ");
    scanf("%s", fileName);
    output = fopen(fileName, "w");
    if (output == NULL) {
        printf("Error opening output file!\n");
        return 1;
    }

    // Read characters from input file and count lines, spaces, and characters
    while ((c = fgetc(input)) != EOF) {
        chars++;
        if (c == '\n') {
            lines++;
        } else if (c == ' ') {
            spaces++;
        }
    }

    // Write results to output file
    fprintf(output, "Number of lines: %d\n", lines);
    fprintf(output, "Number of spaces: %d\n", spaces);
    fprintf(output, "Number of characters: %d\n", chars);

    // Close input and output files
    fclose(input);
    fclose(output);

    printf("Results written to output file successfully!\n");
    return 0;
}

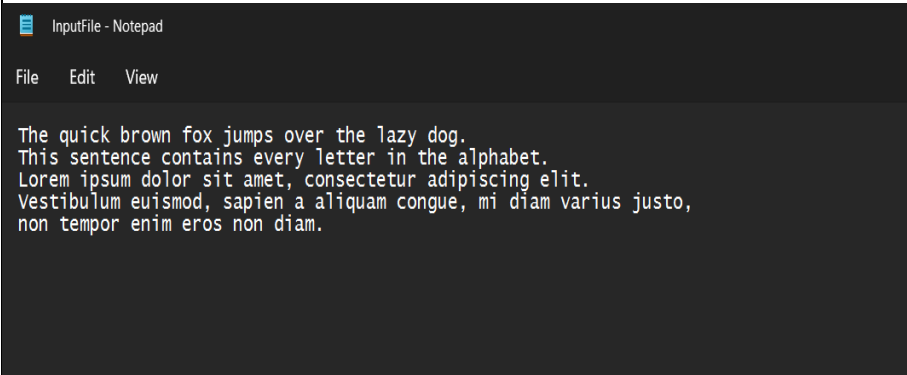
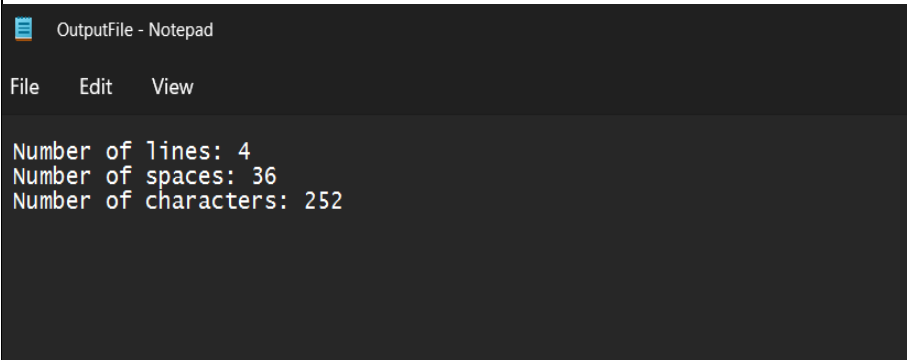
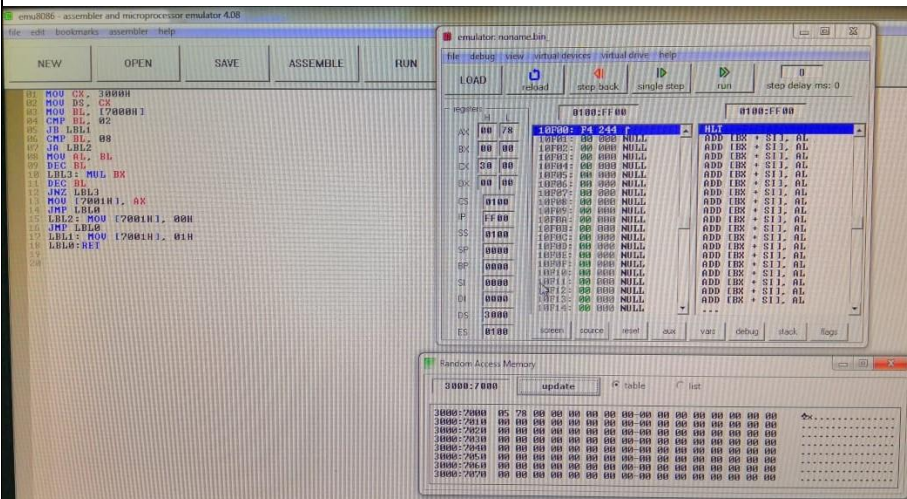
```

Task B:

```

MOV CX, 3000H
MOV DS, CX
MOV BL, [7000H]
CMP BL, 02
JB LBL1
CMP BL, 08
JA LBL2
MOV AL, BL
DEC BL
LBL3: MUL BX
DEC BL
JNZ LBL3
MOV [7001H], AX

```

	<pre> JMP LBL0 LBL2: MOV [7001H], 00H JMP LBL0 LBL1: MOV [7001H], 01H LBL0: RET </pre>
Input to the Program:	<p>Task A:</p>  <p>Task B:</p> <p>Input for factorial: 05H</p>
Output of the program:	<p>Task A:</p>  <p>Task B:</p> 
Outcome of the Experiment:	<p>In this experiment we were able to understand how file handling can be used for performing certain operations on files and storing the results in another file. We were also able to get factorial of a number with the use of assembly language.</p>

References:	File Handling in Python - GeeksforGeeks

Course in-charge-Mayura Gavhane