Experiment No: 01

Experiment Name: File Handling: Copying C Source Code File using Python

Objectives:

- To understand basic file operations in Python: reading from one file and writing to another.
- To practice handling C source code files as plain text.
- To ensure accurate copying of the content without modification.
- To simulate the file manipulation process, a fundamental step in many compiler stages like preprocessing.

Algorithm:

1. Open the input C source file (ccl_1_2254_input.c) in read mode.
2. Read the entire content of the file into a string variable.
3. Open or create an output file (ccl_1_2254_output.c) in write mode.
4. Write the read content into the output file.
5. Close both files.
6. Print a success message to confirm the file copy process.

Code: Input file:

ccl_1_2254_input.c

```c
#include <stdio.h>

int main() {
    int a, b, sum, product;

    printf("Enter first number: ");
    scanf("%d", &a);

    printf("Enter second number: ");
    scanf("%d", &b);

    sum = a + b;
    product = a * b;

    printf("Sum: %d\n", sum);
    printf("Product: %d\n", product);

    return 0;
}

```

Figure01: ccl1_2254_input.c

Main file:

ccl_1_2254_main.py > ...

```python
# Open and read the input file
with open('input.c', 'r') as input_file:
    content = input_file.read()

# Write the content to the output file
with open('output.c', 'w') as output_file:
    output_file.write(content)

print("File copied successfully from input.c to output.c.")

```

Figure02: ccl1_2254_main.py

Output file:

ccl_1_2254_output.c

```c
#include <stdio.h>

int main() {
    int a, b, sum, product;

    printf("Enter first number: ");
    scanf("%d", &a);

    printf("Enter second number: ");
    scanf("%d", &b);

    sum = a + b;
    product = a * b;

    printf("Sum: %d\n", sum);
    printf("Product: %d\n", product);

    return 0;
}

```

Figure03: ccl1_2254_output.c

Discussion:

In this experiment, I implemented a simple Python program to copy the contents of a C source code file from one location to another. The program reads the entire content of the input file and writes it directly into the output file without any modifications. This straightforward file handling process is fundamental in many compiler tasks, especially when dealing with preprocessing or intermediate code generation, where files are frequently read, transformed, and saved. The exercise helped me understand the importance of file I/O operations and how to handle text data efficiently in Python. It also ensures that the copied file retains the exact syntax and formatting of the original source code, which is critical for further compilation steps. The success message printed at the end confirms that the file was copied without errors, providing assurance that the file operations were executed correctly.