

Task 6: File I/O Operations in C++

Objective:

The objective of this task is to understand and implement basic file input and output (I/O) operations in C++. This task demonstrates how to create a text file, write data into it, and then read the contents back and display them on the console.

Problem Statement

Implement file handling in C++ by performing the following steps:

- Create a text file and write a few lines of text into it
- Read the contents of the file
- Display the file contents on the console

Concepts Used

- File Handling in C++
- ofstream (Output File Stream)
- ifstream (Input File Stream)
- Reading and Writing Text Files

Source Code (C++)

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;

// Function to write data to a file
void writeToFile() {
    ofstream outFile("sample.txt");

    if (outFile.is_open()) {
        outFile << "This is line 1 written to the file." << endl;
        outFile << "This is line 2 written to the file." << endl;
        outFile << "File I/O operations in C++ are simple." << endl;
        outFile.close();
    } else {

```

```

        cout << "Error opening file for writing." << endl;
    }
}

// Function to read data from a file
void readFromFile() {
    ifstream inFile("sample.txt");
    string line;

    if (inFile.is_open()) {
        while (getline(inFile, line)) {
            cout << line << endl;
        }
        inFile.close();
    } else {
        cout << "Error opening file for reading." << endl;
    }
}

int main() {
    writeToFile();

    cout << "Reading contents from the file:" << endl;

    readFromFile();

    return 0;
}

```

Explanation of the Code

1. The `<fstream>` header file is included to enable file handling operations.
2. The `writeToFile()` function uses `ofstream` to create and write data into a text file named `sample.txt`.
3. The `is_open()` function checks whether the file has been successfully opened.
4. Several lines of text are written into the file using the insertion operator (`<<`).
5. The file is closed after writing to ensure data is saved properly.
6. The `readFromFile()` function uses `ifstream` to open the same file for reading.
7. The `getline()` function reads the file line by line and prints it to the console.
8. In `main()`, both write and read functions are called sequentially.

Program Output

```
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/ClassWork/day19$ ll
total 16
drwxrwxr-x  3 student student 4096 Jan  9 00:18 .
drwxrwxr-x 24 student student 4096 Jan  8 23:49 ..
drwxrwxr-x  2 student student 4096 Jan  8 23:49 cw/
-rw-rw-r--  1 student student  930 Jan  9 00:18 file_io_operations.cpp
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/ClassWork/day19$ g++ file_io_operations.cpp
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/ClassWork/day19$ ./a.out
Reading contents from the file:
This is line 1 written to the file.
This is line 2 written to the file.
File I/O operations in C++ are simple.
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/ClassWork/day19$ █
```

Conclusion

This task demonstrates basic file input and output operations in C++. By using `ofstream` and `ifstream`, programs can easily store data permanently and retrieve it when required. File handling is a crucial concept for real-world applications such as logging, data storage, and configuration management.