Task 1H: Using Predefined Macros and Logging Macro in C

Objective:
To understand and use predefined macros in C such as __FILE__, __LINE__, __DATE__, __TIME__, and __func__. Further, to design a custom logging macro that prints function name and line number for debugging purposes.

1. Predefined Macros Used

__FILE__	→ Current source file name
__LINE__	→ Current line number
__DATE__	→ Compilation date
__TIME__	→ Compilation time
__func__	→ Current function name

2. Program Code

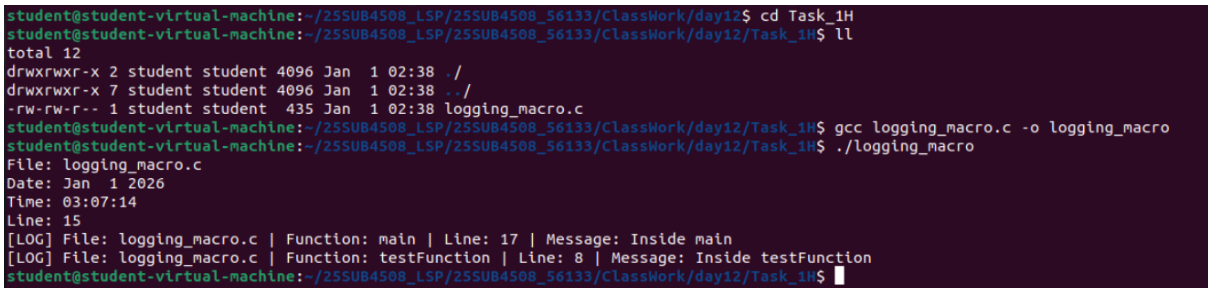
```
#include <stdio.h>

 #define LOG(msg) \
        printf("[LOG] File: %s | Function: %s | Line: %d | Message: %s\n", \
        __FILE__, __func__, __LINE__, msg)

 void testFunction() {
        LOG("Inside testFunction");
 }

 int main() {
        printf("File: %s\n", __FILE__);
        printf("Date: %s\n", __DATE__);
        printf("Time: %s\n", __TIME__);
        printf("Line: %d\n", __LINE__);

        LOG("Inside main");
        testFunction();
        return 0;
 }
```

3. Compilation Instructions

Compile the program normally using GCC:

```
gcc logging_macro.c -o logging_macro
```

4. Program Output



```
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/ClassWork/day12$ cd Task_1H
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/ClassWork/day12/Task_1H$ ll
total 12
drwxrwxr-x 2 student student 4096 Jan  1 02:38 ./
drwxrwxr-x 7 student student 4096 Jan  1 02:38 ../
-rw-rw-r-- 1 student student  435 Jan  1 02:38 logging_macro.c
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/ClassWork/day12/Task_1H$ gcc logging_macro.c -o logging_macro
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/ClassWork/day12/Task_1H$ ./logging_macro
File: logging_macro.c
Date: Jan  1 2026
Time: 03:07:14
Line: 15
[LOG] File: logging_macro.c | Function: main | Line: 17 | Message: Inside main
[LOG] File: logging_macro.c | Function: testFunction | Line: 8 | Message: Inside testFunction
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/ClassWork/day12/Task_1H$
```

5. Observations & Explanation

1. Predefined macros are replaced by the compiler at compile time.
2. __FILE__ shows the current source file name.
3. __LINE__ changes dynamically based on macro usage location.
4. __func__ provides the function name where the macro is expanded.
5. The LOG macro helps in debugging without manually printing details.

6. Conclusion

This task demonstrates the usefulness of predefined macros and logging macros for debugging and tracing program execution. Logging macros are widely used in real-world software systems for error tracking and diagnostics.