The compilation process step by step, using terminal commands on Windows:

Assuming our .c file is named hello.c and contains the classic "Hello, World!" program:

```
#include <stdio.h>

int main() {
    printf("Hello, World!\n");
    return 0;
}
```

We'll use the MinGW compiler on Windows. Open the Command Prompt or PowerShell:

Step 1: Preprocessing

```
C:\>gcc -E hello.c -o hello.i
```

This command runs the preprocessor, expanding macros and includes, and saves the output to hello.i.

Verification:

```
C:\>type hello.i
```

This will display the preprocessed code, including the expanded includes and macros.

Step 2: Compilation

```
shell
C:\>gcc -S hello.i -o hello.s
```

This command compiles the preprocessed code into assembly code, saving it to hello.s.

Verification:

```
shell
```


```
shell
C:\>gcc -S hello.i -o hello.s
```

This command compiles the preprocessed code into assembly code, saving it to hello.s.

Verification:

```
shell
C:\>type hello.s
```

This will display the assembly code.

Step 3: Assembly

```
shell
C:\>as hello.s -o hello.o
```

This command assembles the assembly code into object code, saving it to hello.o.

Verification:

```
C:\>dumpbin /headers hello.o
```

This will display the object file headers, indicating it's an object file.

Step 4: Linking

```
C:\>gcc hello.o -o hello.exe
```

This command links the object code with library code (if necessary) to create an executable file hello.exe.

Verification:

```
C:\>dumpbin /headers hello.exe
```

This will display the executable file headers, indicating it's an executable file.

Step 5: Loading and Execution

```
C:\>hello.exe
```

This command loads the executable file into memory and executes it, printing "Hello, World!" to the console.

Verification:

```
C:\>echo %ERRORLEVEL%
```

This will display the exit status of the program, which should be 0 indicating successful execution.