# Module 0: Your First Steps in C (Days 1-2)

## 1. Philosophy Focus: Done-for-you Training Plan

Welcome! The goal of this course is to make you a comfortable and proficient C programmer. You don't need to worry about *what* to learn next or *how* to practice it. Just follow the steps laid out for each day. We'll start with the absolute basics: getting your tools ready and running your first program.

### **DAY 1: Setting Up Your Environment**

**Goal:** Install a C compiler and a text editor so you can write and run code.

**1. The Tools You Need**

* **A Text Editor:** This is where you write your code. We recommend **Visual Studio Code (VS Code)**. It's free, modern, and has great support for C.
* **A C Compiler:** This is a program that translates your human-readable C code into machine-readable instructions that your computer can execute. The most common compiler is **GCC** (GNU Compiler Collection).

**2. Installation Instructions**

* **Windows:** The easiest way to get GCC is by installing **MinGW-w64**.
  1. Go to the official MinGW-w64 website and download the installer.
  2. Follow the installation instructions. A crucial step is adding the compiler's bin directory to your system's PATH environment variable. This allows you to run gcc from any command prompt. A quick search for "add to PATH windows 10/11" will give you a detailed guide.
* **macOS:** You already have a C compiler! Open the Terminal app and type xcode-select --install. This installs the Xcode Command Line Tools, which includes the clang compiler (which works just like gcc for our purposes).
* **Linux (Debian/Ubuntu):** Open a terminal and run the command: sudo apt update && sudo apt install build-essential. This installs gcc and other essential development tools.

3. Verify Your Installation

Open your command prompt (Terminal on Mac/Linux, Command Prompt or PowerShell on Windows) and type:

gcc --version

If you see a message with the GCC version number, you're all set! If you get an error like "command not found," revisit the installation steps, especially the PATH setup on Windows.

### **DAY 2: "Hello, World!" - Your First Program**

**Goal:** Write, compile, and run the simplest C program. This cycle is the fundamental workflow of a C programmer.

1. Write the Code

Open VS Code and create a new file named hello.c. The .c extension is important! Type the following code exactly as it appears:

#include <stdio.h>  
  
int main(void) {  
 // This line prints the text to the screen  
 printf("Hello, World!\n");  
   
 return 0;  
}

**2. Understand the Code (Briefly)**

* #include <stdio.h>: This line includes the "Standard Input/Output" library. We need it to use the printf function.
* int main(void): This is the main "function" where every C program begins execution.
* printf(...): This is the function we're using to print text.
* \n: This is a special character that means "new line".
* return 0;: This tells the operating system that the program finished successfully.

3. Compile the Code

Go back to your command prompt. Make sure you are in the same directory where you saved hello.c. You can use the cd (change directory) command to navigate. Once there, type:

gcc hello.c -o hello

Let's break that down:

* gcc: Calls the compiler.
* hello.c: The input file (your source code).
* -o hello: The output file. This tells the compiler to name the executable program hello. If you leave this out, it will be named a.out on Linux/Mac or a.exe on Windows.

4. Run the Program

After the compilation command finishes, you will see a new file named hello (or hello.exe on Windows). To run it, type the following in your command prompt:

* On macOS/Linux: ./hello
* On Windows: hello

You should see Hello, World! printed to your screen.

**Congratulations!** You have completed the core C programmer workflow: **Write -> Compile -> Run.** You will repeat this cycle thousands of times.

**Day 2 Practice:**

1. Modify the string inside printf to say something else.
2. Compile and run the program again to see your change.
3. Add a second printf statement to print another line of text. Compile and run it. This will help make the workflow feel natural.