☑ Lab 0 — Development Environment Setup & First Program

Quick Checklist

Prep & Install

- ☑ Choose install method (Windows: Cygwin ✅ / WSL, macOS, or Linux)
- ☑ Install required packages for your OS (Cygwin: chere, diffutils, flex, gcc-core, gcc-g++, grep, make)
- ☑ Verify g++, make, flex, and diff run in your terminal

Editor Setup

- ☐ Install Visual Studio Code (recommended)
- ☐ Add C/C++ extension (and WSL extension if using WSL)

First Program

- ☐ Create folder lab0_<NETID> and go inside it
- ☐ Create intro_<NETID>.cpp with your personalized greeting (NetID + current year)
- ☐ Compile and run it successfully

Testing with diff

- ☐ Place expected.out (download from Canvas) in your folder
- ☐ Redirect output to my.out
- ☐ Compare with expected.out using diff — confirm only allowed differences

Bash Script Automation

- ☐ Create lab0_test_<NETID>.sh with compile, run, diff, and whoami steps
- ☐ Make script executable
- ☐ Run script successfully

Submission

- ☐ Screenshot showing: date, compilation, program runs, diff result, and whoami

📌 Goal

• Set up a UNIX-like command-line environment and install the tools we'll use throughout the course.

• Learn basic terminal commands.

• Write, compile, and run a personalized C++ program.

• Compare your program's output against an expected file with diff.

• Automate testing with a bash script.

Help & Support

• You may attend either lab section each week (optional, recommended).

• I'm available during office hours and by appointment.

• Share questions/solutions in Discord or Canvas forums.

• Work together — pair up and compare notes to get set up.

1 Installing the Development Environment

All assignments assume a UNIX-like terminal environment (Linux, macOS Terminal, or Windows with Cygwin/WSL).

Windows — Option 1: Cygwin ☑ (Recommended)

1. Download installer: https://www.cygwin.com/setup-x86_64.exe
2. Run installer with defaults, select any mirror.
3. In "Package Selection," search and install: chere, diffutils, flex, gcc-core, gcc-g++, grep, make
4. After installation, right-click any folder in Explorer → Bash Prompt Here.
5. If you forget a package, re-run the installer and add it.

Windows — Option 2: WSL (Windows Subsystem for Linux)

Follow these official, student-friendly guides:

• Microsoft Learn — Install WSL: https://learn.microsoft.com/windows/wsl/install

• Microsoft Learn — Get started using VS Code with WSL: https://learn.microsoft.com/windows/wsl/tutorials/wsl-vscode

• VS Code Docs — Developing in WSL: https://code.visualstudio.com/docs/remote/wsl

• Windows Central — How to install WSL 2 (walkthrough): https://www.windowscentral.com/how-install-wsl2-windows-10

After completing a guide, open Ubuntu and install course tools:

```
sudo apt update && sudo apt install -y build-essential flex diffutils grep
```

Tip: In VS Code connected to WSL, set EOL to LF to avoid CRLF issues when running scripts.

macOS

```
xcode-select --install
```

Install Homebrew from https://brew.sh

(Optional) Enable "New Terminal at Folder" in System Settings → Keyboard → Shortcuts → Services.

Linux

```
Debian/Ubuntu:  sudo apt update && sudo apt install build-essential flex diffutils
grep
Fedora:         sudo dnf install make gcc gcc-c++ flex diffutils grep
Arch/Manjaro:   sudo pacman -S base-devel flex diffutils grep
```

2 Choosing a Code Editor

Any plain-text editor works. We recommend Visual Studio Code with the C++ extension.

3 Write Your First Program

Step 1: Create your lab folder

```
mkdir lab0_<NETID>
cd lab0_<NETID>
```

Step 2: Create intro_<NETID>.cpp

Use your editor of choice (Notepad, VS Code, Sublime, etc.). Make sure the file name ends with .cpp and not .txt.

Step 3: Type the program code

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

int main() {
   cout << "Hello from <NETID> in <CURRENT_YEAR>!" << endl;
   return 0;
}
```

Step 4: Replace placeholders

- <NETID> → your actual NetID (example: abc123)
- <CURRENT_YEAR> → the actual year (example: 2025)

Step 5: Save the file

(Ctrl+S in most editors).

Step 6: Compile your program

```
g++ intro_<NETID>.cpp -o intro_<NETID>
```

Step 7: Run your program

```
./intro_<NETID>
```

Expected output: Hello from abc123 in 2025!

4 Testing with diff

Download expected.out from Canvas and place it in your folder.

Save your program's output:

```
./intro_<NETID> > my.out
```

Compare:

```
diff -y expected.out my.out
```

Reading diff:

Your output file will be on the right side, expected.out will be on the left side

Lines with '|' in between the files contain some difference(s).

Lines with '<' in between, indicate the file on the left contains a line that the other file does not

Lines with '>' in between, indicate the file on the right contains a line that the other file does not

Lines with nothing in between them are a perfect match.

If diff shows nothing, your output matches exactly.

5 Automating with a Bash Script

Create lab0_test_<NETID>.sh in the same folder with the following:

```
#!/usr/bin/env bash
date
echo "Compiling intro_<NETID>.cpp..."
g++ intro_<NETID>.cpp -o intro_<NETID>
echo "Running program first time..."
./intro_<NETID>
echo "Running program second time, saving output..."
./intro_<NETID> > my.out
echo "Comparing with expected..."
diff -y expected.out my.out
echo "Current user is: $(whoami)"
```

Remember to replace placeholders with your actual NET ID.

Change the permissions for your script:

```
chmod +x lab0_test_<NETID>.sh
```

💡 Tip: If you see 'Permission denied' when running your script, use chmod +x lab0_test_<NETID>.sh to set execute permissions.

Execute the script:

```
./lab0_test_<NETID>.sh
```

```
ls -l intro_<NETID>.cpp intro_<NETID> expected.out
bash -x ./lab0_test_<NETID>.sh
```

6 What to Submit

Submit the following:

• One screenshot with the output of your script

• A brief written report (see Section 7 for details).

```
$ ./lab0_test_dpw9.sh
Mon Aug 18 14:08:38 CDT 2025
Compiling intro_dpw9.cpp...
Running program first time...
Hello from dpw9 in 2025!
This line will not change.
Running program second time, saving output...
Comparing with expected...
Hello from abc123 in 4567!                          | Hello from dpw9 in 2025!
This line will not change.                            This line will not change.
Your program should not have this line.             <
Current user is: willi
```

7 Written Report (Brief Reflection)

Write a short report (about half a page) reflecting on your Lab 0 experience. The report must include:

- Two concrete details from your setup — Describe at least two specific steps you performed while installing or configuring your environment. For example, a command you ran, a package you installed, or an error message you encountered and how you solved it.
- Your Bash script — Copy-paste one or two lines from your lab0_test_<NETID>.sh file and explain what those lines do in your own words.
- Your impression of the tools — In one or two sentences, describe what you noticed about using `diff`, `g++`, or other commands you used during this lab (such as `whoami` or `pwd`). Did anything surprise you? Was there something confusing?
- Personal reflection — End with a one-sentence opinion: Was this lab easy, challenging, or somewhere in between for you?

📌 Tips:

• Keep your writing clear and concise — one to two short paragraphs are enough.

• Use your own experience and wording. Answers that are overly generic or do not clearly reflect your own work may lose credit.

TL;DR — Install & Run

Submission: Upload both your screenshot (with all required outputs) and your brief written report.

```
Windows – Cygwin (✅ Recommended): install chere diffutils flex gcc-core gcc-g++ grep
make
Windows – WSL: follow the Microsoft Learn / VS Code docs above, then install: sudo apt
install build-essential flex diffutils grep
macOS: xcode-select --install → Homebrew from brew.sh
Linux: install build-essential flex diffutils grep (package names vary)

Then:
mkdir lab0_<NETID> → cd lab0_<NETID> → create intro_<NETID>.cpp → compile → run →
save output → diff compare → create script → chmod +x → run script → screenshot
```