The programming environment

For this course, we expect you to write programs on "a real computer," namely your own laptop. Below are instructions for how to set up your computer for programming in C++.

Visual Studio Code (VS Code)

We suggest you (install and) use **VS Code** (https://code.visualstudio.com) for developing your assignment solutions. VS Code is a fairly popular programming environment, developed by Microsoft, that is available without cost for Windows, macOS, and Linux.

Linux (Unix) execution environment

We use a Linux (Unix) execution environment for compiling and running your programs. CodeGrade also uses (virtual) Linux machines for compiling, running, and testing your program submissions. We are using the *g*++ compiler. Please see below how to get such an environment up and running on your laptop, depending on its operating system.

Laptops by operating system

Please read the section that refers to your laptop's OS:

Windows 10

If your laptop runs Windows 10, you can use the **Windows Subsystem for Linux** (https://docs.microsoft.com/en-us/windows/wsl/about) (WSL).

The Windows Subsystem for Linux has two major versions, v1 and v2. The installation guide below works for both versions and has special remarks for enabling the v2 of the WSL. Please note that WSL2 is available with Windows 10, updated to version 2004, Build 19041, or higher. For our purposes, it does not matter whether you use WSL1 or WSL2. If you want to know the details, you can check **here** (https://docs.microsoft.com/en-us/windows/wsl/compare-versions#:~:text=File%20performance%20across%20the%20Windows,faster%20performance%20with%20WSL%201.).

Installation Guide

1. Open PowerShell as Administrator and run:
   ```
   dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart
   ```
   Alternative: Open the "Turn Windows Features on and off" application through the Windows search bar. Then, scroll down and check the Windows Subsystem for Linux box.
2. After the installation is finished, **restart** your computer.

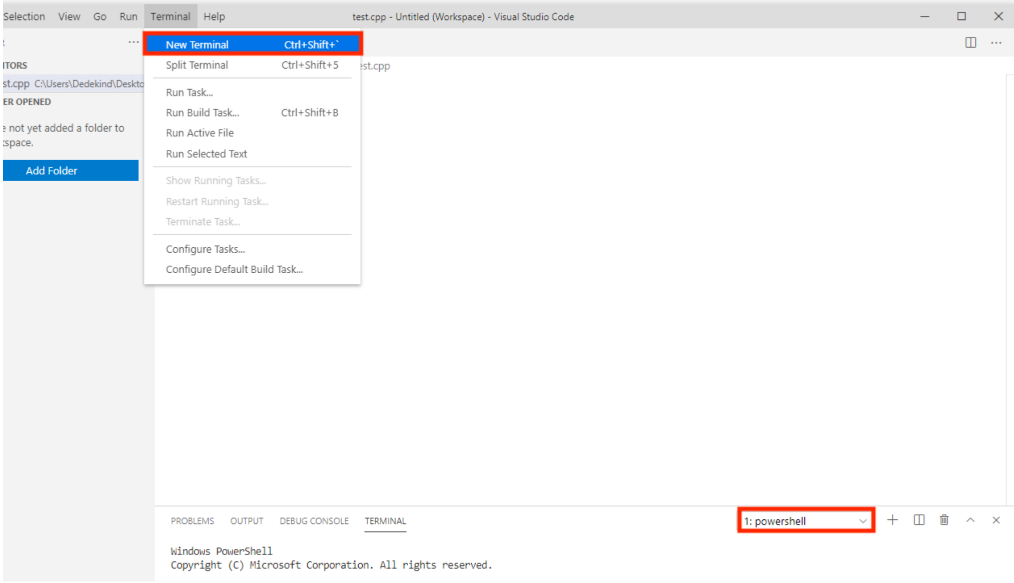
3. Go to the Windows Store and choose your preferred Linux distribution to install. To find the Linux distribution you want, search for "Linux" on the Windows Store. Once you've done that, it should show you several different distributions, we recommend Ubuntu.
4. Launch the Linux App you just installed (either from the Windows Store or by searching the distribution name at the search bar) and proceed in creating a new user.
5. Keep the terminal window open. First update the packages with the command: `sudo apt-get update && sudo apt-get upgrade` (you will be asked to input the password you chose for your subsystem).
   Now, you must install the GNU compiler tools which include the g++ compiler. Run the following command:
   ```
   sudo apt-get install build-essential gdb
   ```
6. (Optional) For WSL2, open PowerShell as Administrator and run:
   ```
   Enable-WindowsOptionalFeature -Online -FeatureName VirtualMachinePlatform
   ```
   Restart your computer.
   Then, open PowerShell as Administrator and run:
   ```
   wsl --set-version DistroName 2
   ```
   (replace "DistroName" with the name of the Linux Distribution you installed (ex. `wsl --set-version Ubuntu 2`)).
   If you receive "WSL 2 requires an update to its kernel component" update the Linux kernel by downloading and installing the latest Linux Kernel from **here** **(https://docs.microsoft.com/en-us/windows/wsl/wsl2-kernel)** and retry the last step.

If you encounter problems during the installation process described above, please also read the official guide from **Microsoft Docs** **(https://docs.microsoft.com/en-us/windows/wsl/install-win10#install-your-linux-distribution-of-choice)** and the dedicated **WSL Troubleshooting Page** **(https://docs.microsoft.com/en-us/windows/wsl/troubleshooting)**.

How to work with WSL

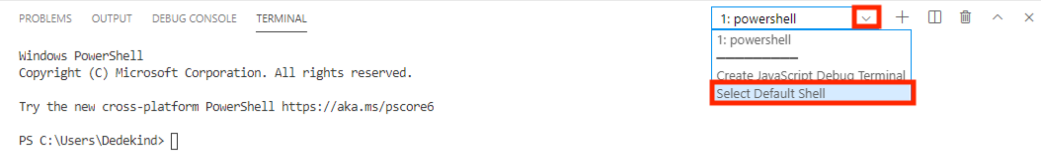
VSCode

1. With VSCode open a c++ file. Click the terminal option in the top bar and select New Terminal. You will notice that a terminal is opened at the bottom of the screen, as a PowerShell terminal.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
```

2. Now you should set your default terminal to the WSL BASH Shell. Click the little arrow next to "1:powershell" and click the "Select Default Shell"

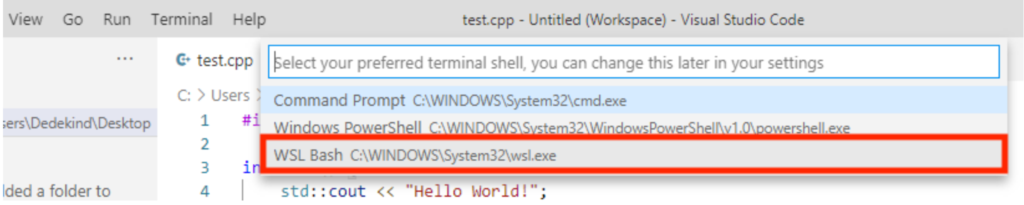


```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Dedekind> 
```

3. On the new dialog choose the "WSL Bash" option.



Select your preferred terminal shell, you can change this later in your settings

Command Prompt C:\WINDOWS\System32\cmd.exe

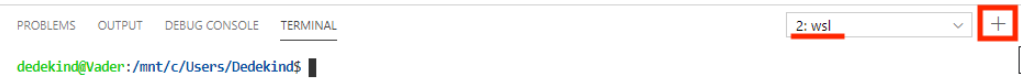
Windows PowerShell C:\WINDOWS\System32\WindowsPowerShell\v1.0\powershell.exe

WSL Bash C:\WINDOWS\System32\wsl.exe

```
#i

in
    std::cout << "Hello World!";
```

4. Finally, click the "+" symbol next to the dropdown menu from before. You will notice that now you have a bash terminal open and you can use the compiler from Linux.



```
dedekind@Vader:/mnt/c/Users/Dedekind$ 
```

Use the BASH terminal without VSCode

Open the "bash" shell application through the search bar. You will notice that you are currently in the `/mnt/c/WINDOWS/system32` directory.

Now you can simply access your Windows files by changing directory with the cd command.

macOS

On a Mac, you can use the *Terminal* app that will give you a *bash* (or *zsh*, doesn't matter) command-line shell in which you can navigate your working directories, invoke the *g++* compiler, and execute your programs.

1. Open the terminal and run the command:
   `g++`
   If your terminal displays the message `clang: error: no input files` then the g++ is already installed.
2. If the g++ is not installed, an alert box will appear to install the g++ compiler. Click the Install button.
3. If the alert box did not appear and you did not have the g++ compiler, you can install it by running the following command:
   `xcode-select --install`

Linux

If your laptop runs Linux, you are basically all set. Just install VS Code.

Any other environment

At the end of the day, your programs must pass the tests on CodeGrade, which is using g++ on Linux. Feel free to develop your programs elsewhere, but do not expect us to provide help with any environment not described above. Also, for deciding whether your program does what it is supposed to do, the tests on CodeGrade define the reference.