Lab 00:
Windows Installation

Provided Files, Folders, and Executables

- .vscode (folder)
  - tasks.json
- Makefile
- main.c
- Dockerfile
- mGBA.exe

Files to Edit

- tasks.json

Instructions

In this lab, you will be installing the software to write, compile, and run GBA games for this class. It is broken up into various parts. If any part does not produce the expected outcome, alert a TA via Piazza and fix the problem before continuing. **It is incredibly important that you read each and every instruction**. Many of your questions will be answered if you read carefully. Do not skim this document! :)

Part 1: Docker Desktop

Part 1.1: Verify your Windows operating system (OS). There are different Docker installation procedures for Windows 10 Home versus Windows 10 Pro. If your machine is not running Windows 10 (e.g. Windows 7 or 8), email Julia (juliareuter@gatech.edu) and Instructor Hansen (ah252@gatech.edu) before continuing.

a. **To determine your OS, press the Windows key + R, type *winver*, and enter.**
b. **Take note of the version and build, too, you'll need this information for Step 2.**

Part 1.2: Follow the appropriate procedure for your OS.

a. Go to **Part 1.5** if your OS is **Windows 10 Pro**. Skip 1.3 and 1.4.
b. Go to **Part 1.4** if your OS is **Windows 10 Home** AND your **version** is **2004+** and your **build** is **above 19041**. Skip 1.3, but DO NOT skip 1.5.
c. Go to **Part 1.3** if your OS is **Windows 10 Home** AND your **version** is **NOT 2004** and/or your **build** is **NOT above 19041**, respectively. DO NOT skip 1.4 and 1.5.

Part 1.3: Update your machine to Windows 10 Home version 2004, build 19041+
In order to run Docker Desktop on your machine, you'll need to update your Windows 10 Home version and build. Follow the detailed instructions on how to do so here: https://support.microsoft.com/en-us/help/4028685/windows-10-get-the-update. It is to note that this update may take one to two hours to complete.

Once you've completed the update, ensure again that you've got the correct version and build by pressing the **Windows key + R, type winver, and enter**. If you do not have version 2004 and a build at or above 19041, then check again for updates.

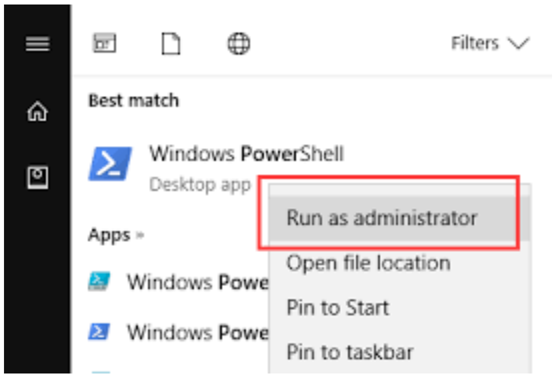
Move on to **Part 1.4** when complete.

Part 1.4: Install Subsystem for Linux on Windows with WSL2
Now, this is (likely) the trickiest part of the lab, but we'll guide you along the process way :)

Follow the detailed instructions here (excluding the "*Set up a new distribution*" section): https://docs.microsoft.com/en-us/windows/wsl/install-win10. Note that important bits of information on how to follow these instructions properly are delineated below.

Thus, you'll need to complete the following sections:

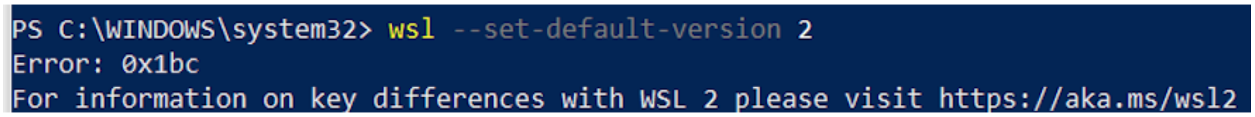
1. *Install the Windows Subsystem for Linux*
    a. Enter the command provided in the instructions in the Powershell, ensuring you're running Powershell as an **administrator**.



2. *Update to WSL 2*
    a. You should already have the correct Windows version and build, so scroll down to "*Enable the 'Virtual Machine Platform' optional component*" subsection, and run the command provided in the instructions, ensuring you're running Powershell as an **administrator**. Once the command has executed, restart your computer.
3. *Set WSL 2 as your default version*
    a. Run the command provided in Powershell as an **administrator**. Two errors may (but hopefully won't) occur:
        i. **Error 1**: wsl --set-default-version results as an invalid command. Enter wsl --help. If the --set-default-version is not listed, it means that your OS

doesn't support it and you need to update to version 2004, Build 19041 or higher. Revisit **Part 1.3** to update your OS. Then return back to this step and run the command provided again.

ii. **Error 2**: If you get an error as pictured below, you must download the latest WSL2 Linux kernel. To download the latest WSL2 Linux kernel, follow the detailed instructions here: https://docs.microsoft.com/en-us/windows/wsl/wsl2-kernel. Then, rerun the command provided again.



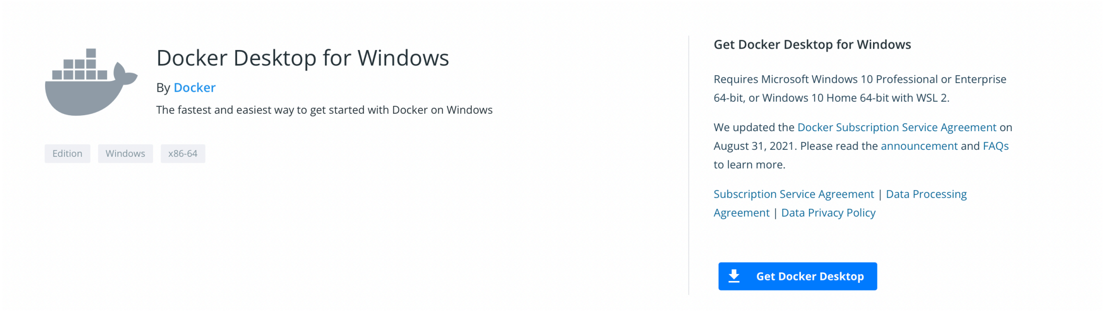
```
PS C:\WINDOWS\system32> wsl --set-default-version 2
Error: 0x1bc
For information on key differences with WSL 2 please visit https://aka.ms/wsl2
```

4. *Install your Linux distribution of choice*
   a. Install a Linux distribution. If you do not have a preference, we suggest installing **Ubuntu 16.04 LTS** from the Microsoft store. After you've installed a Linux distribution, move on to **Part 1.5**.

Part 1.5: Install Docker Desktop

First, you'll need to download Docker Desktop. In order to do this, you'll need to sign up for Docker Hub. Sign up for Docker Hub here: https://hub.docker.com/signup. You can use your personal or GT email; either is fine.

Once you've signed up, download the stable Windows Docker Desktop here: https://hub.docker.com/editions/community/docker-ce-desktop-windows/. You'll simply press the "Get Docker Desktop" button on the right hand side of the screen. Open the executable once it's downloaded and follow the installation instructions carefully.



Docker Desktop for Windows

By Docker

The fastest and easiest way to get started with Docker on Windows

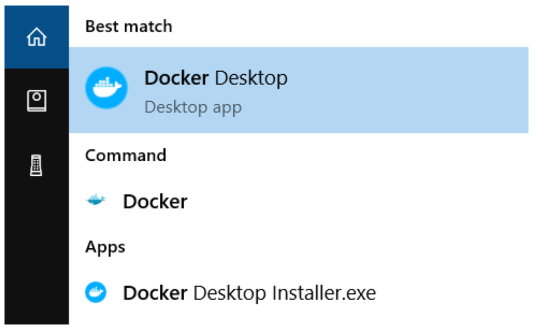
Get Docker Desktop for Windows

Requires Microsoft Windows 10 Professional or Enterprise 64-bit, or Windows 10 Home 64-bit with WSL 2.

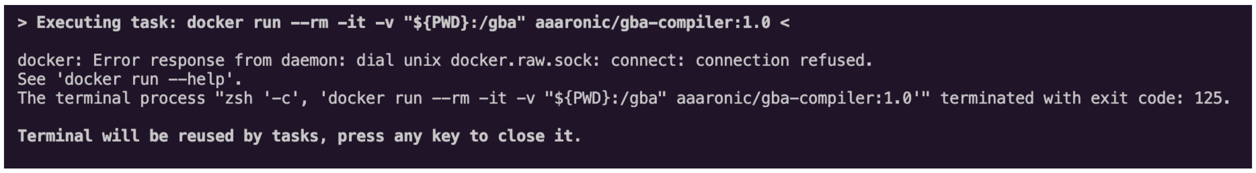
We updated the Docker Subscription Service Agreement on August 31, 2021. Please read the announcement and FAQs to learn more.

Subscription Service Agreement | Data Processing Agreement | Data Privacy Policy

Once you've installed Docker Desktop, you should be able to launch the application. You should sign in using the credentials you used to create your Docker account. Then, Docker Desktop should be running as long as you've launched the application!



YOU MUST BE RUNNING DOCKER IN ORDER TO BUILD YOUR CODE. Otherwise, when you attempt to build your code, you will see the following error:



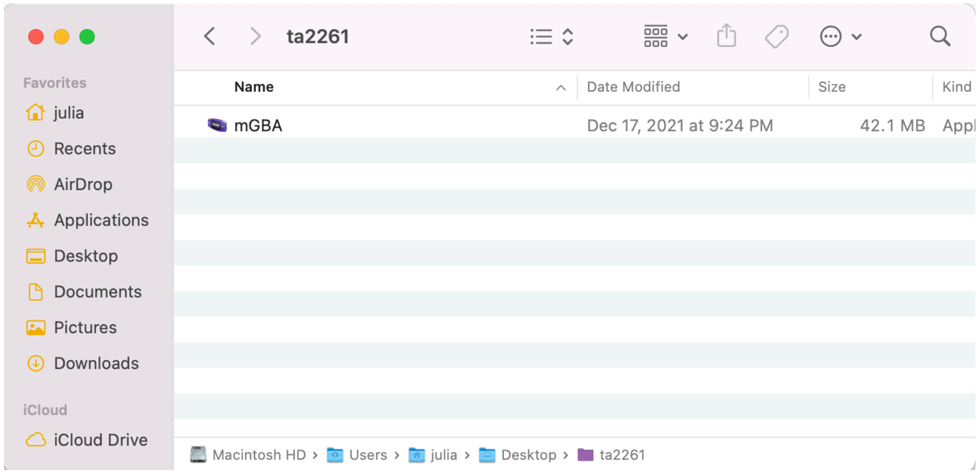
```
> Executing task: docker run --rm -it -v "${PWD}:/gba" aaaronic/gba-compiler:1.0 <

docker: Error response from daemon: dial unix docker.raw.sock: connect: connection refused.
See 'docker run --help'.
The terminal process "zsh '-c', 'docker run --rm -it -v "${PWD}:/gba" aaaronic/gba-compiler:1.0'" terminated with exit code: 125.

Terminal will be reused by tasks, press any key to close it.
```

If you see this when you attempt to build your project (which we will get to in a moment), you'll know you forget to open the Docker Desktop application. Launch it, then build again.

Part 2: mGBA

Let's get the GBA emulator setup! mGBA is the emulator for this class. If you already have a GBA emulator that you are comfortable with, I still highly recommend you use this one for this class. It has some special features that will come in handy for debugging.

Find the "*mGBA.exe*" executable in the Lab00 folder and put the executable in a parent folder folder. I recommend creating a parent CS2261 folder that will contain all of your homework, labs, etc. and keeping the "*mGBA.exe*" in this folder. Lab00 should be in this CS2261 folder, too (as a subfolder). Keep a note of the exact path of the "*mGBA.exe*". In the image below, you'll see a recommended folder setup (of course, this is for a Mac, but a similar file structure is recommended for Windows users). For executables, Windows uses the .exe extension. For executables, Mac uses the .app extension. the. The exact path to "*mGBA.app*", in my case and on my Mac, is /Users/julia/Desktop/ta2261/mGBA.app

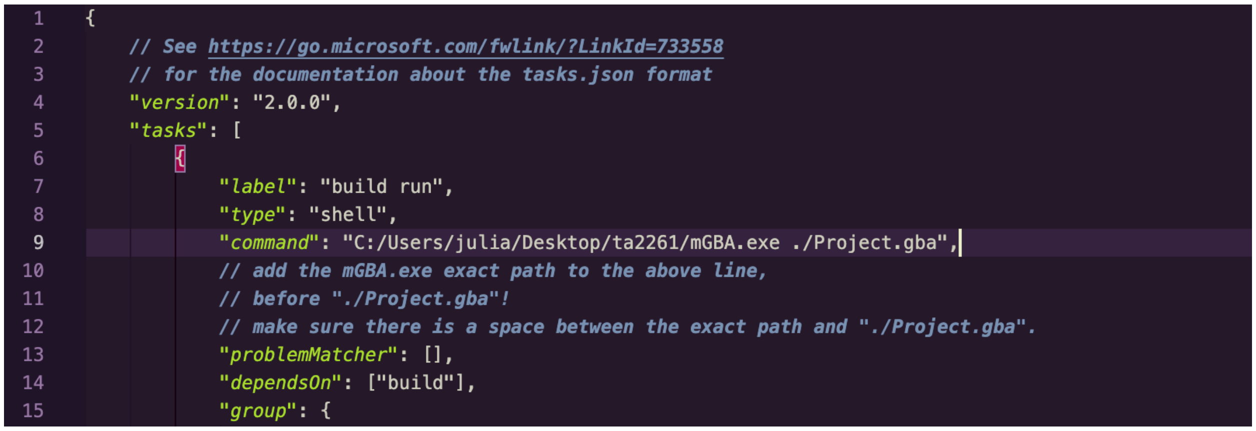


Name	Date Modified	Size	Kind
mGBA	Dec 17, 2021 at 9:24 PM	42.1 MB	Appl

You can verify the directory location of the "*mGBA.exe*" on your machine by opening the CS2261 folder where you placed the "*mGBA.exe*", and dragging the "*mGBA.exe*" icon into the command prompt. The output in the command prompt is the exact path to the "*visualboyadvance-m.exe*". Copy this path.

In my case, as stated before, the exact path is /Users/julia/Desktop/ta2261/visualboyadvance-m.app. Copy this exact path and paste it into your task.json, on line 9 in before "./Project.gba". Make sure there is a space between your exact path and "./Project.gba".

IMPORTANT: FOR WINDOWS USERS, REPLACE ALL BACKSLASHES (i.e. "\") IN THE PATH WITH FORWARD SLASHES (i.e. "/"). Your code will NOT build otherwise. For example, if the exact path of the emulator executable was C:\Users\julia\Desktop\ta2261\mGBA.exe then on line 9 before "./Project.gba", you should have the following: C:/Users/julia/Desktop/ta2261/mGBA.exe



```
1   {
2       // See https://go.microsoft.com/fwlink/?LinkId=733558
3       // for the documentation about the tasks.json format
4       "version": "2.0.0",
5       "tasks": [
6           {
7               "label": "build run",
8               "type": "shell",
9               "command": "C:/Users/julia/Desktop/ta2261/mGBA.exe ./Project.gba",
10              // add the mGBA.exe exact path to the above line,
11              // before "./Project.gba"!
12              // make sure there is a space between the exact path and "./Project.gba".
13              "problemMatcher": [],
14              "dependsOn": ["build"],
15              "group": {
```

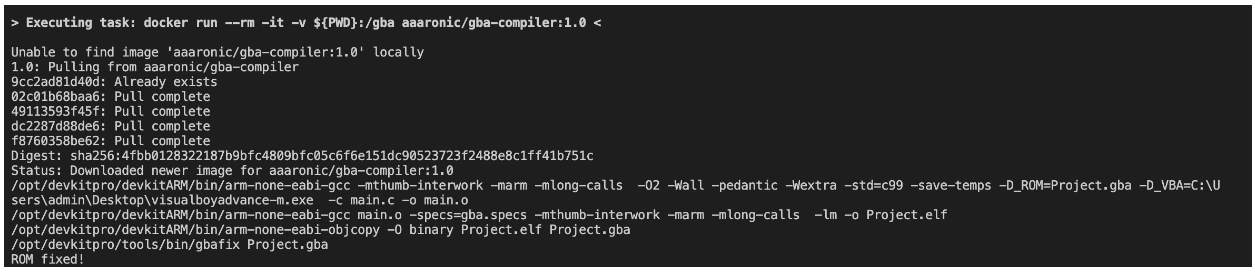
Save your changes. The image above outlines what my (truncated) tasks.json looks like. Your exact path on line 9 will be different!

Part 3: Visual Studio Code

For this class, we are now encouraging you to use VSCode for writing and building your projects. Even if you have VSCode installed, do not skip these steps.

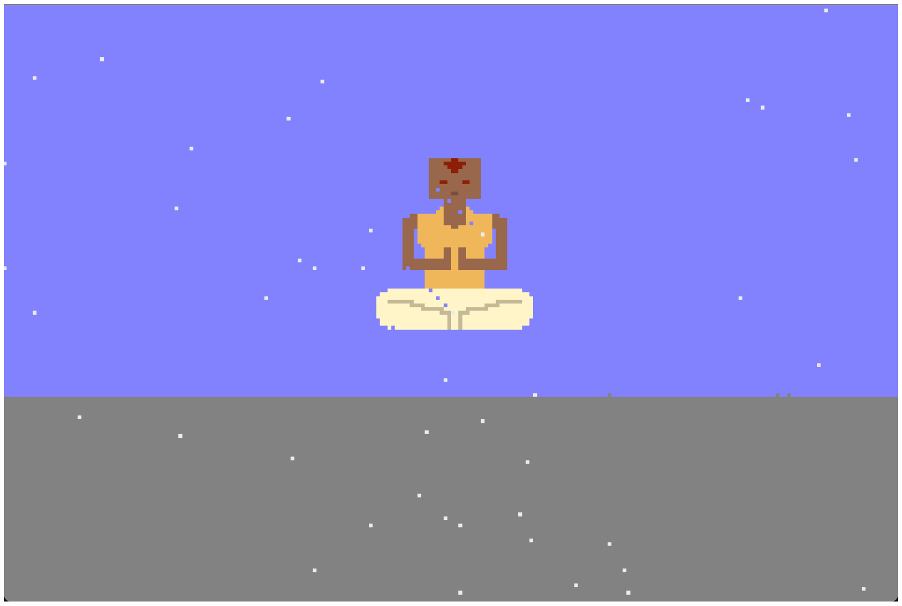
1. If you don't have VSCode installed, download it from the project website, https://code.visualstudio.com/download.
    a. Pick the "User Installer" for Windows 7/8/10 for your system (32 vs 64-bit)
2. Once you have installed VSCode, open the Lab00 folder from VS code (File > Open > Lab00).
3. Open main.c and then hit **ctrl+shift+b** to build your project (remember, Docker Desktop needs to be running!). This first compilation process will take some time because a Docker image is being downloaded from Dockerhub (don't worry about the nitty gritty

details of this). **All subsequent builds will not take this long** (whew). In the terminal output in VSCode, you should see something like this for the very first time you build:



```
> Executing task: docker run --rm -it -v ${PWD}:/gba aaaronic/gba-compiler:1.0 <

Unable to find image 'aaaronic/gba-compiler:1.0' locally
1.0: Pulling from aaaronic/gba-compiler
9cc2ad81d40d: Already exists
02c01b68baa6: Pull complete
49113593f45f: Pull complete
dc2287d88de6: Pull complete
f8760358be62: Pull complete
Digest: sha256:4fbb0128322187b9bfc4809bfc05c6f6e151dc90523723f2488e8c1ff41b751c
Status: Downloaded newer image for aaaronic/gba-compiler:1.0
/opt/devkitpro/devkitARM/bin/arm-none-eabi-gcc -mthumb-interwork -marm -mlong-calls  -O2 -Wall -pedantic -Wextra -std=c99 -save-temps -D_ROM=Project.gba -D_VBA=C:\U
sers\admin\Desktop\visualboyadvance-m.exe  -c main.c -o main.o
/opt/devkitpro/devkitARM/bin/arm-none-eabi-gcc main.o -specs=gba.specs -mthumb-interwork -marm -mlong-calls  -lm -o Project.elf
/opt/devkitpro/devkitARM/bin/arm-none-eabi-objcopy -O binary Project.elf Project.gba
/opt/devkitpro/tools/bin/gbafix Project.gba
ROM fixed!
```

Then, you should also see a GBA game running that looks something like the following picture. If so, you are done with the lab and may follow the submission instructions! :D If not, there is an issue somewhere, so notify a TA via Piazza!



Submission Instructions

Zip up the entire sample project folder, including all source files, the Makefile, and everything produced during compilation **(including the .gba file and your tasks.json in the .vscode folder)**. Submit this zip on Canvas.
It is your responsibility to ensure that the file you submitted on Canvas includes all the files required.

Name your submission Lab00_FirstnameLastname, for example:"Lab00_JuliaReuter.zip".