CS 2261 Lab 00: Installation

Provided Files

- Installation Files
    1. SublimeTextSetup.txt
    2. vbam-2.0.1.zip
- Sample Project
    1. main.c
    2. Makefile

Files to Edit/Add

- Makefile

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 and fix the problem before continuing.

- **Part One – Cygwin**
    - ➢ Cygwin is what students running Windows will use as a C Compiler. This is the biggest difference between the Windows and Mac installations, since the Mac folk (usually) already have a C Compiler installed.
    1. Go to the project website, https://www.cygwin.com/
    2. If your computer is 64-bit, click on `setup-x86_64.exe`. Otherwise, click on `setup-x86.exe`.

The Cygwin DLL currently works with all recent, commercially released x86 32 bit and 64 bit versions of Windows, starting with Wind

NOTE: As previously announced, Cygwin version 2.5.2 was the last version supporting Windows XP and Server 2003. (Instructions for

Current Cygwin DLL version

The most recent version of the Cygwin DLL is 2.9.0. Install it by running setup-x86_64.exe (64-bit installation) or setup-x86.exe (32-b

Use the setup program to perform a fresh install or to update an existing installation.

Note that individual packages in the distribution are updated separately from the DLL so the Cygwin DLL version is not useful as a gene

Support for Cygwin

3. Open the folder where it downloaded and run that file.
4. On the first popup, click next. Then, select "Install from the Internet" and click next.
5. Choose the directory where you want Cygwin to install. This must be somewhere that you will not move or delete until this class is over. <u>Make note of this location, because you will have to find it in a later step.</u> Then click next.
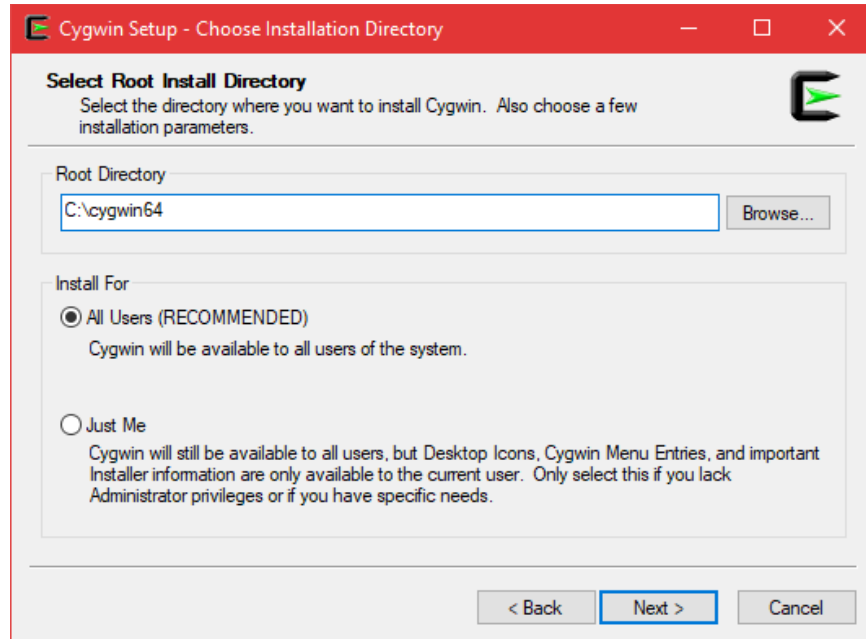

6. This next part doesn't matter. You can just accept the default and click next.
7. Select "Use System Proxy Settings" and click next.
8. Select the first mirror site in the list, the click next.
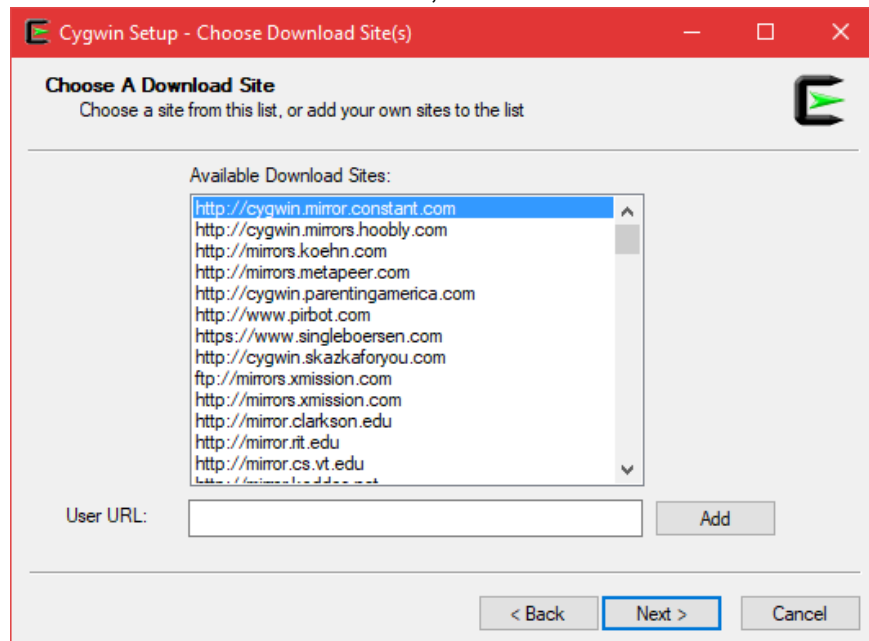
9. This part is complicated, so pay close attention. In the search bar, type "gcc-core". Click the plus next to the category "All" and the category "Devel" and find the package titled exactly "gcc-core". Find the arrow to the right of the "Skip" for that package, and click it. Select the latest version that does not say "(test)" by it.
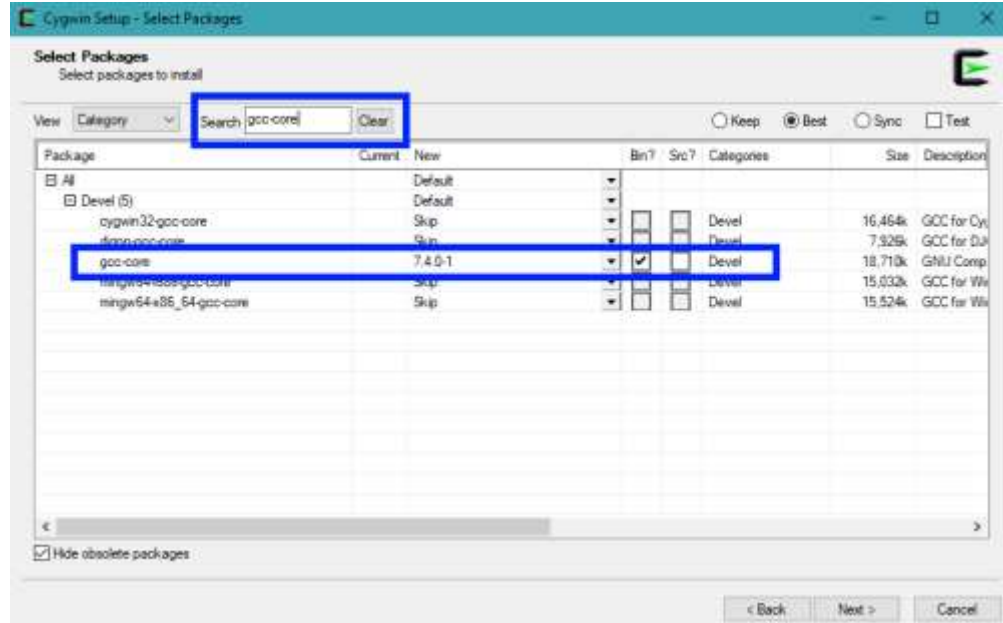
10. Back in the search bar, type "gcc-g++". Find the package titled exactly "gcc-g++". Find the arrow to the right of the "Skip" for that package, and click it. Select the latest version that does not say "(test)" by it.
11. Again, in the search bar, type "gdb". Find the package titled exactly "gdb". Find the arrow to the right of the "Skip" for that package, and click it. Select the latest version that does not say "(test)" by it.
12. One last time, in the search bar, type "make". Find the package titled exactly "make". Find the arrow to the right of the "Skip" for that package, and click it. Select the latest version that does not say "(test)" by it.
13. Click next, then next again, then wait quite a while for everything to finish.
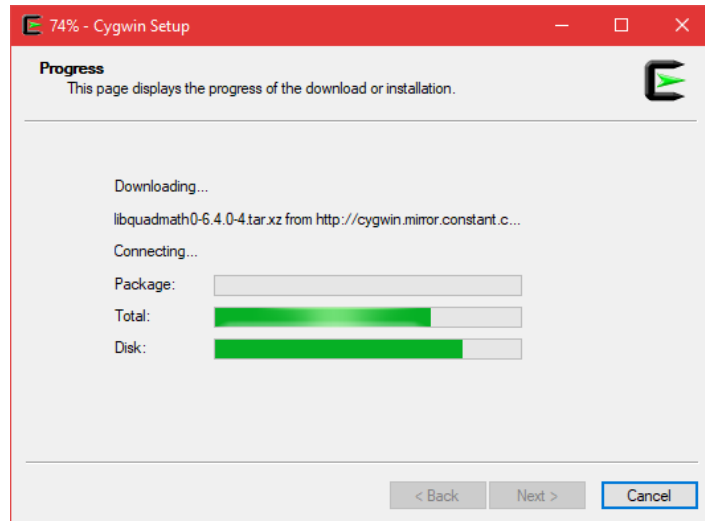
14. Uncheck the two boxes (or don't; it's your clutter, not mine) and then click finish.
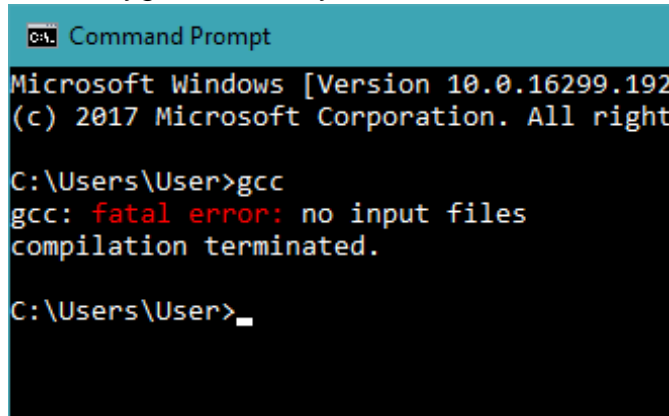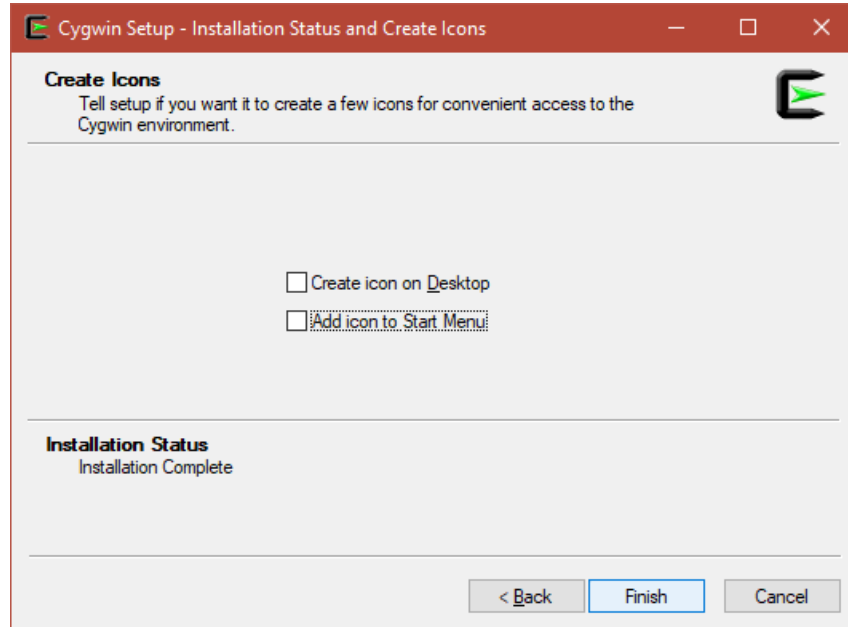

15. The last thing we need to do is add Cygwin's bin folder to our System path. Open Control Panel, then go to System and Security, then System, then Advanced System Settings.
16. In the dialog, click "Environmental Variables".
17. In the new dialog, under System Variables (not User Variables), select "Path" and click "Edit".

18. This new screen will depend on which version of Windows you are running.

- If it is an organized list of items (like in the picture below), click "New", and in the blank that opens up, copy the exact path to where you installed Cygwin (the location I told you to remember earlier) and add "\bin" to the end. Then press Enter.

- If it is a singular text box, move the cursor all the way to the end, type a semicolon, and then copy the exact path to where you installed Cygwin (the location I told you to remember earlier) and

add "\bin" to the end. If it has any spaces in it, surround the whole thing in quotes (not including the semicolon you added before).

19. Press OK on all of the Control Panel dialogs you have opened thus far.
20. Open Command Prompt.
21. Type "gcc" and press enter. If what you see is something like the picture below, you installed Cygwin correctly. If not, alert a TA.


➢ Congrats! You have completed the hard part. You are free to delete all of the things associated with this part in your Downloads folder (unless of course you were crazy enough to install Cygwin there).

- **Part Two – devkitARM**
  ➢ DevkitARM is your development kit for the Game Boy Advance, created by a group called devkitPro. Without it, we can compile C code, but not actually format the result for the Gameboy. So let's fix that.
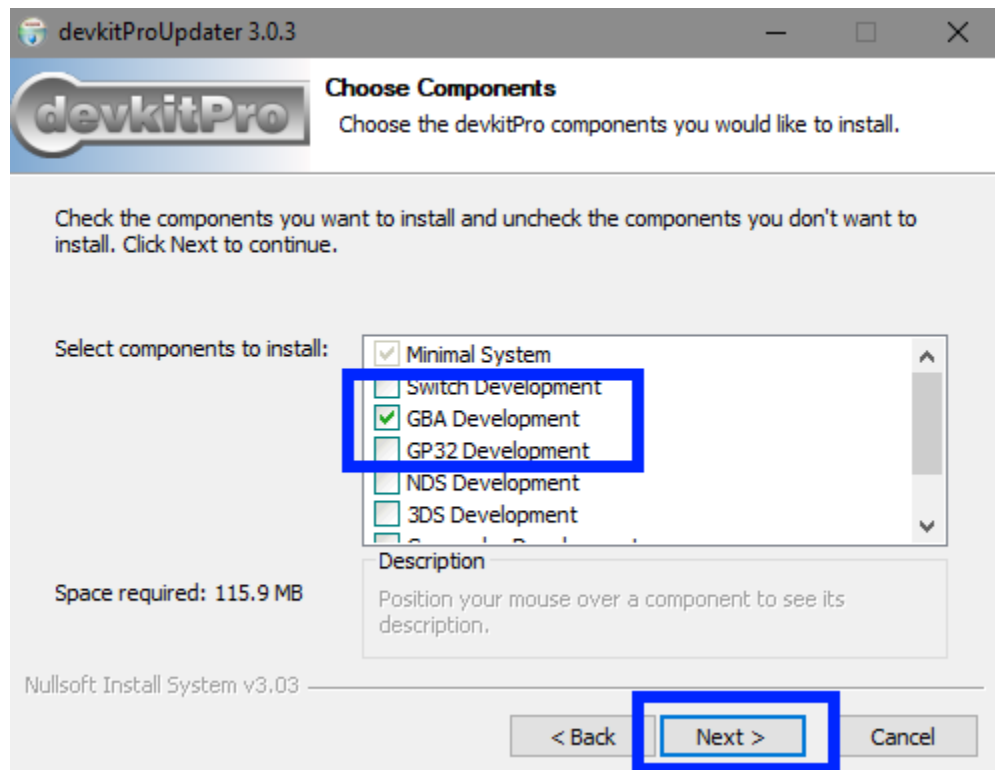 1. Go to the project website, https://github.com/devkitPro/installer/releases
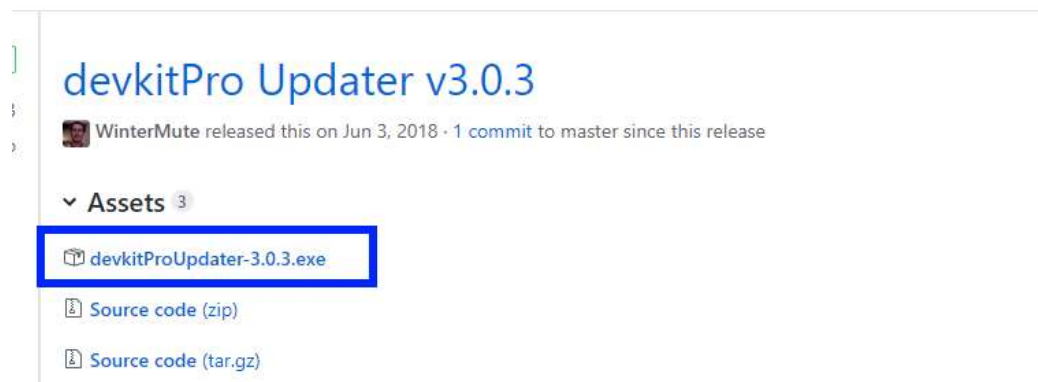
  2. Click on "devkitProUpdater-3.0.3.exe" to download it.
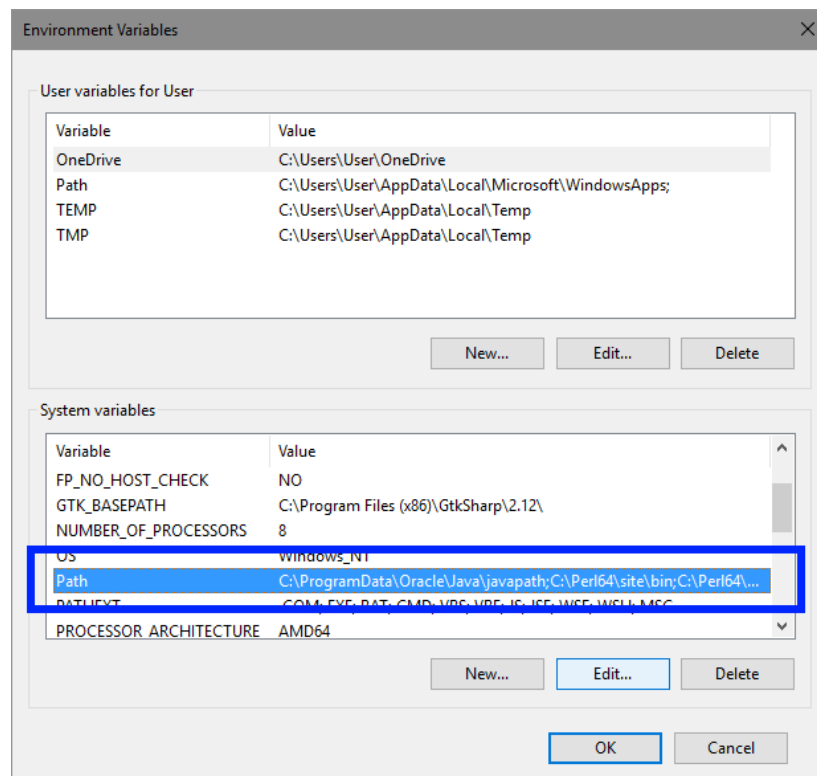 3. When the download finishes, open it, then hit Next.
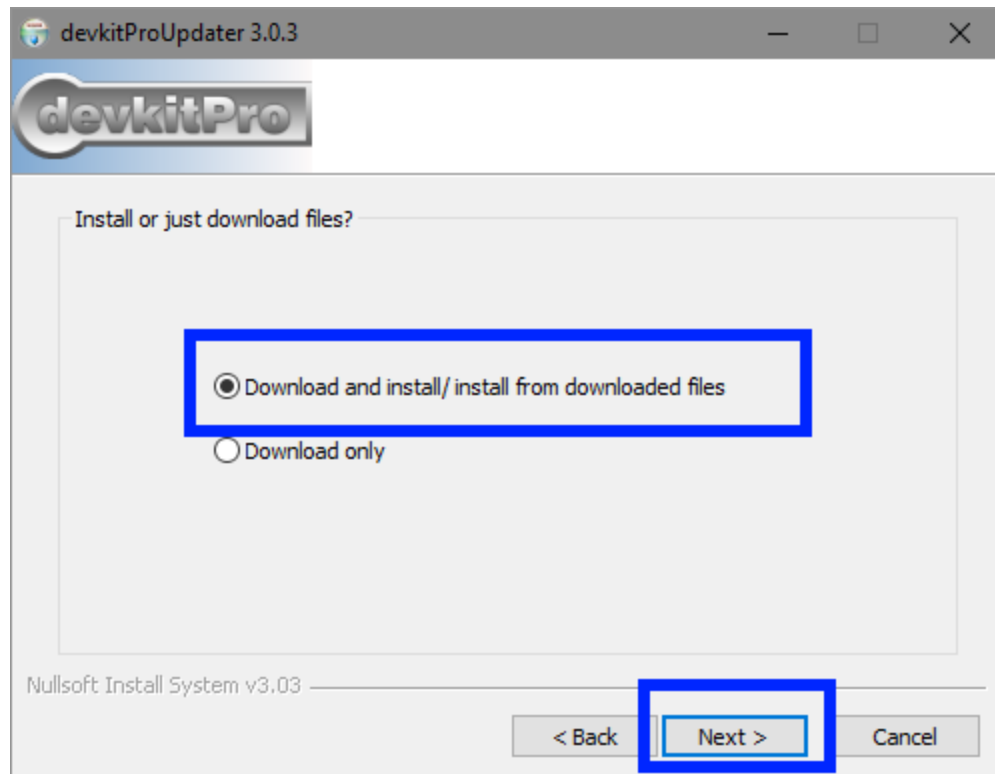
4. Click "Download and install/ install from downloaded files" and hit Next.
5. Select "Remove downloaded files" and hit Next.

6. During this step, deselect everything in the list **except** GBA, and hit Next.

7. Select a destination folder somewhere that you will not move or delete until this class is over. The same folder where you put Cygwin will usually suffice. <u>Make note of this location, because you will have to find it in a later step.</u>

8. When asked to choose the Start Menu Folder, put whatever you want. The default is fine. Hit Next, wait for the install to finish (it will take a long time, and open a bunch of strange terminal windows, and look like it's going to fail, but it won't).

9. After the install finishes, the last thing we need to do is add devkitARM's bin folder to our System path, just like we did with Cygwin. Open Control Panel, then go to System and Security, then System, then Advanced System Settings.

10. In the dialog, click "Environmental Variables".

11. In the new dialog, under System Variables (not User Variables), select "Path" and click "Edit".

12. This new screen will depend on which version of Windows you are running.
   - If it is an organized list of items (like in the picture below), click "New", and in the blank that opens up, copy the exact path to where you installed devkitPro (the location I told you to remember earlier) and add "\devkitARM\bin" to the end. Then press Enter.
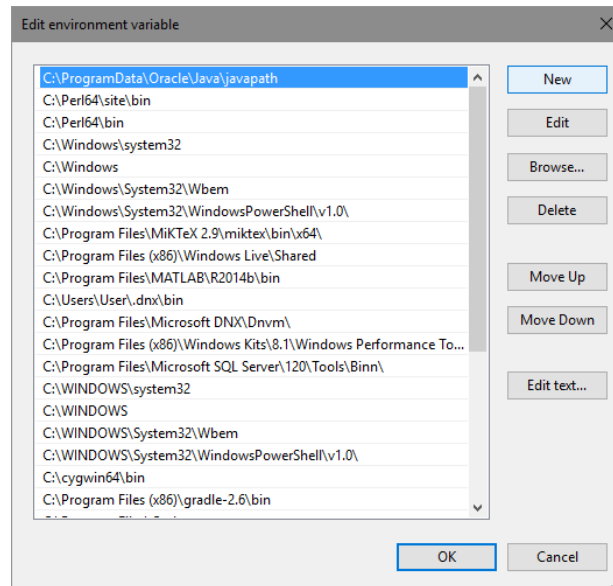
- If it is a singular text box, move the cursor all the way to the end, type a semicolon, and then copy the exact path to where you installed devkitPro (the location I told you to remember earlier) and add "\devkitARM\bin" to the end. If it has any spaces in it, surround the whole thing in quotes (not including the semicolon you added before).

13. Press OK on all of the Control Panel dialogs you have opened thus far.
14. If you have any terminal windows (or text editors) open right now, close and reopen them.

➢ That's all you have to do for devkitARM. Easy, right?

- **Part Three – VisualBoyAdvance-M**
    ➢ VisualBoyAdvance-M is your emulator. Since we can now compile and format our code, it would be useless if we couldn't actually run it. So let's fix that.
        ▪ Note: 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.
    1. Downloading this from its website requires having 7-Zip installed, so I provided it to you in a `.zip` file. Extract this file.
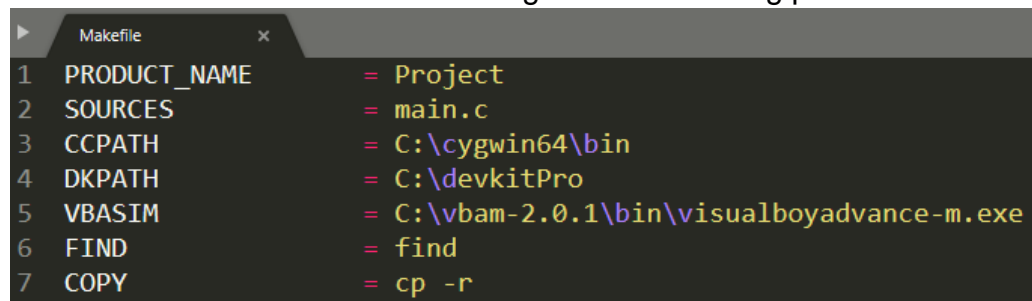   2. Find the folder it creates once unzipped, and move it somewhere that you will not move or delete until this class is over. The same folder where you put Cygwin and devkitARM will usually suffice. Make note of this location, because you will have to find it in a later step.
    ➢ That's all have to do for VisualBoyAdvance-M. Even easier.

- **Part Four – Makefile**

➢ Now that we have installed everything, we need to set up the compilation process so that we don't have to run 1,000 commands every time we want to compile our code. C uses Makefiles for this. I have provided one for you, but it needs to know where to find the things you just installed.

1. In the provided Sample Project, find `Makefile` and open with a text editor (like Notepad or Sublime Text). Near the beginning, you will see three lines that end in "=".

2. After the "=" next to "CCPATH", type the exact path to the `cygwin` (or `cygwin64`) folder, and add "\bin" to the end. If there are any spaces in the path, surround it in quotes.

3. After the "=" next to "DKPATH", type the exact path to the `devkitPro` folder (you don't need to add anything to the end for this one). If there are any spaces in the path, surround it in quotes.

4. After the "=" next to "VBASIM", type the exact path to the `vbam-2.0.1` folder, add "\bin" to the end, then add "\visualboyadvance-m.exe". If there are any spaces in the path, surround it in quotes.

5. Make sure the result looks something like the following picture.

```
    Makefile                    ×
1   PRODUCT_NAME        = Project
2   SOURCES             = main.c
3   CCPATH              = C:\cygwin64\bin
4   DKPATH              = C:\devkitPro
5   VBASIM              = C:\vbam-2.0.1\bin\visualboyadvance-m.exe
6   FIND                = find
7   COPY                = cp -r
```

6. Save and close the file.

➢ Congrats. We now officially have everything it takes to simply compile, format, and run GBA games. Now let's make it simpler.

- **Part Five – Sublime Text**

  ➢ We can make the process even more streamlined by incorporating the compilation process into our text editor. For this class, we highly encourage Sublime Text. Even if you have Sublime Text installed, do not skip these steps.

  1. If you don't have Sublime Text installed, download it from the project website, https://www.sublimetext.com/3. I don't need to walk you through this installation process; it's user-friendly enough.

  2. Once you have Sublime Text installed, open it, then go to Tools, then Build System, then click "New Build System…"

  3. It will open a new file. Delete everything in it, then paste in everything from the provided `SublimeTextSetup.txt`.

```
1  {
2      "cmd": ["make", "build"],
3      "variants":
4      [
5          {
6              "name": "Build",
7              "cmd": ["make", "build"]
8          },
9          {
10             "name": "Run",
11             "cmd": ["make", "run"]
12         },
13         {
14             "name": "Clean",
15             "cmd": ["make", "clean"]
16         },
17         {
18             "name": "All",
19             "cmd": ["make", "all"]
20         }
21     ]
22 }
```

4. Save it in the folder it suggests when you hit save, and title it "GBA.sublime-build".
5. Close everything in Sublime Text, and then close Sublime Text.
6. Reopen Sublime Text, then use it to open `Makefile` in the Sample Project folder.
7. Go to Tools, then Build System, and make sure GBA is selected.
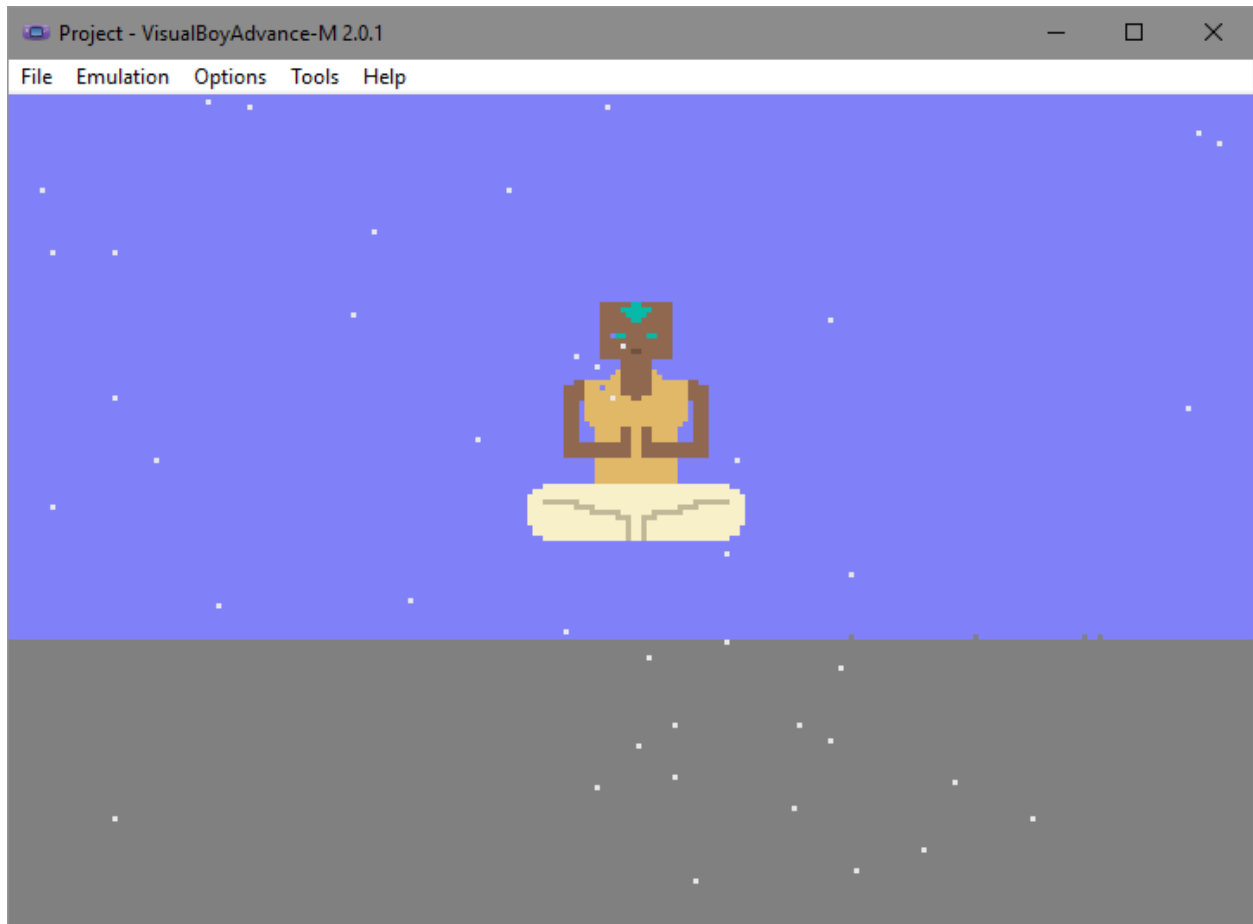8. Go to Tools, then Build With, then select Run.

After completing all of these steps, you should see output in the console at the bottom of Sublime Text that looks something like the following picture.

You should also see a GBA game running that looks something like the following picture.

If so, you are done with the lab. If not, there is an issue somewhere.

Submission Instructions

Zip up the entire sample project folder, including all source files, the Makefile, and everything produced during compilation (including the .gba file). Submit this zip on Canvas. Name your submission Lab00_FirstameLastname, for example: "Lab00_ChianneConnelly.zip".