# Mac Insructions

# installing cmake

## is `cmake` installed?

Let's check to see if `cmake` is installed on your system: type `cmake --version` at the commandline.

If you do not get a response similar to this, then you do not have `cmake` on your system and you have to install it.

```
CS-xx-projects  $> cmake --version
cmake version 3.16.0-rc4

CMake suite maintained and supported by Kitware (kitware.com/cmake).
CS-xx-projects  $>
```

## install `cmake`

We use `homebrew` to install `cmake`:

```
brew install cmake
```

This will go on and on...

```
CS-xx-projects  $> brew install cmake
Updating Homebrew...
==> Auto-updated Homebrew!
Updated 4 taps (homebrew/core, homebrew/cask, homebrew/services and mongodb/br
ew).
==> New Formulae
aerc                                    libxcomposite
aida-header                             libxcursor
aliddns                                 libxdamage
ansible@2.9                             libxdmcp
arturo                                  libxext
asroute                                 libxfixes
atkmm@2.28                              libxfont
attr                                    libxft
aws-console                             libxi
aws-rotate-key                          libxinerama
bandit                                  libxkbfile
```

... and on...

```
==> Updated Formulae
Updated 4905 formulae.
==> Renamed Formulae
glibmm@2.64 -> glibmm@2.66              pangomm@2.42 -> pangomm@2.46
gtk+4 -> gtk4                           prest -> prestd
now-cli -> vercel-cli
==> Deleted Formulae
boost@1.55                godep                      scw@1
boost@1.59                llvm@6                     stlviewer
confluent-platform        meson-internal             unp64
curl-openssl              mysql-connector-c++@1.1    unrar
dtrx                      ori                        woboq_codebrowser
fmsx                      pgplot                     xspin
gobby                     rmtrash                    xu4
==> New Casks
7777
abyssoft-teleport
accordance
aio-creator-neo
aldente
aleo-studio
amazon-workdocs-drive
anka-build-cloud-controller-and-registry
anka-build-cloud-registry
anka-virtualization
atemosc
aural
```

... and on...

```
Warning: Treating cmake as a formula. For the cask, use homebrew/cask/cmake
==> Downloading https://homebrew.bintray.com/bottles/cmake-3.19.4.mojave.bottl
==> Downloading from https://d29vzk4ow07wi7.cloudfront.net/c4eac1fa4580a117a33
                                                                        0.7
#                                                                       2.3
##                                                                      3.8
###                                                                     5.5
####                                                                    6.9
######                                                                  8.6
#######                                                                10.2
########                                                               11.8
#########                                                              13.2
##########                                                             14.9
###########                                                            16.6
#############                                                          18.2
##############                                                         19.9
###############                                                        21.4
################                                                       23.0
#################                                                      24.6
##################                                                     26.1
###################                                                    27.7
####################                                                   29.4
#####################                                                  30.9
######################                                                 32.6
#######################                                                34.1
########################                                               35.7
```

... and on...

Finally, cmake is installed!

```
==> Summary
🍺  /usr/local/Cellar/cmake/3.19.4: 6,376 files, 63.9MB
==> `brew cleanup` has not been run in 30 days, running now...
Removing: /Users/sassanbarkeshli/Library/Caches/Homebrew/gdbm--1.18.1_1.mojave
.bottle.tar.gz... (203.3KB)
Removing: /Users/sassanbarkeshli/Library/Caches/Homebrew/openssl@1.1--1.1.1g.m
ojave.bottle.tar.gz... (5.3MB)
Removing: /Users/sassanbarkeshli/Library/Caches/Homebrew/python@3.8--3.8.5.moj
ave.bottle.tar.gz... (16.6MB)
Removing: /Users/sassanbarkeshli/Library/Caches/Homebrew/readline--8.0.4.mojav
e.bottle.tar.gz... (517.6KB)
Removing: /Users/sassanbarkeshli/Library/Caches/Homebrew/sqlite--3.33.0.mojave
.bottle.tar.gz... (2.0MB)
Removing: /Users/sassanbarkeshli/Library/Caches/Homebrew/xz--5.2.5.mojave.bott
le.tar.gz... (386.8KB)
Removing: /Users/sassanbarkeshli/Library/Caches/Homebrew/portable-ruby-2.6.3_2
.yosemite.bottle.tar.gz... (9.1MB)
Removing: /Users/sassanbarkeshli/Library/Logs/Homebrew/gdbm... (64B)
Removing: /Users/sassanbarkeshli/Library/Logs/Homebrew/readline... (64B)
Removing: /Users/sassanbarkeshli/Library/Logs/Homebrew/sqlite... (64B)
Removing: /Users/sassanbarkeshli/Library/Logs/Homebrew/xz... (64B)
Removing: /Users/sassanbarkeshli/Library/Logs/Homebrew/openssl@1.1... (64B)
Removing: /Users/sassanbarkeshli/Library/Logs/Homebrew/python@3.8... (3 files,
 172.3KB)
Pruned 1 symbolic links and 2 directories from /usr/local
```

## check the version of the cmake again:

To make sure `cmake` is intalled correctly, run `cmake --version` again:

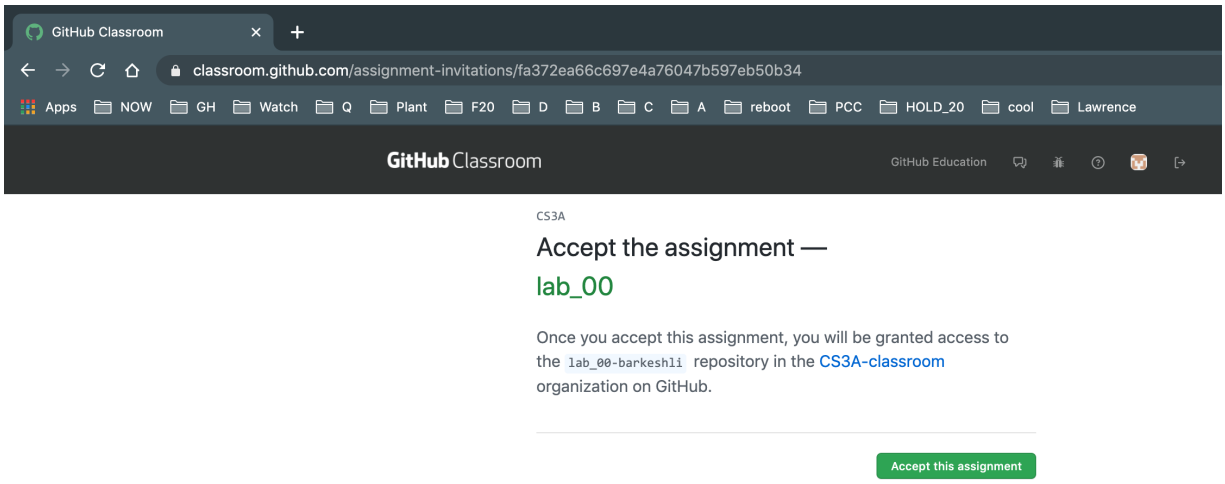
```
CS-xx-projects  🤬$> cmake --version
cmake version 3.16.0-rc4

CMake suite maintained and supported by Kitware (kitware.com/cmake).
CS-xx-projects  🤬$>
```

# Accept the assignment

## Here is the assignment link for CS3A and here is the link for CS8

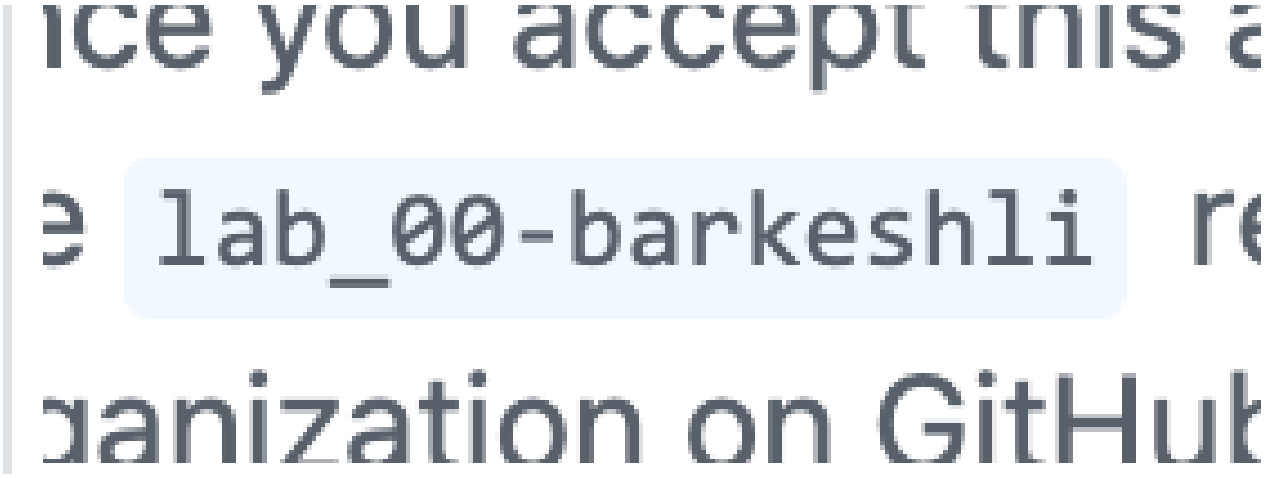
## Accept assignment page:

Once you click on the assignment link, we need you to *accept* the assignment. This will create a repo under your github username. But before you click and accept the assignment, let's look at a couple of things a bit more closely.



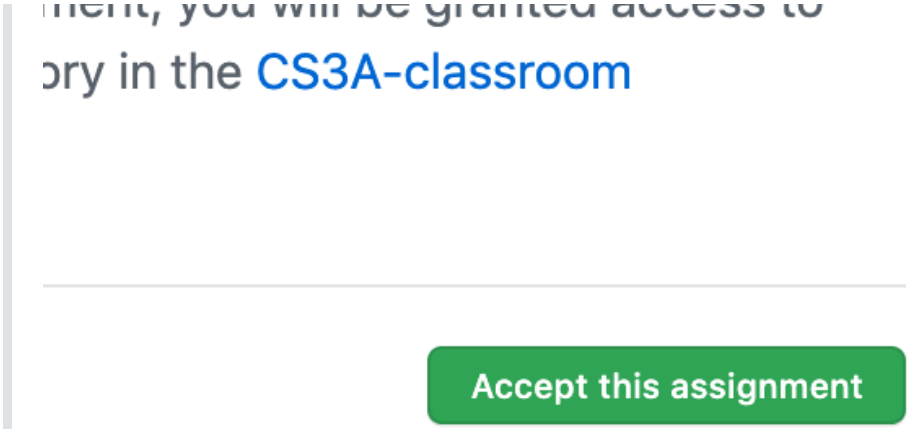

## Your repo name:

This is your repo name. The name of the assignment followed by your github name.



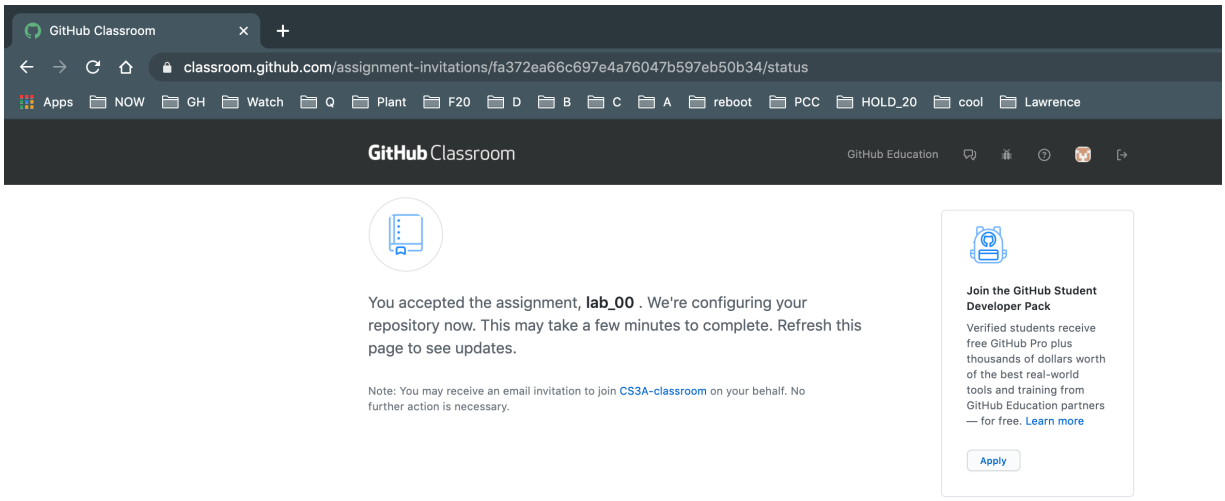

## Accept the assignment

You will *accept* the assignment by clicking the green button.



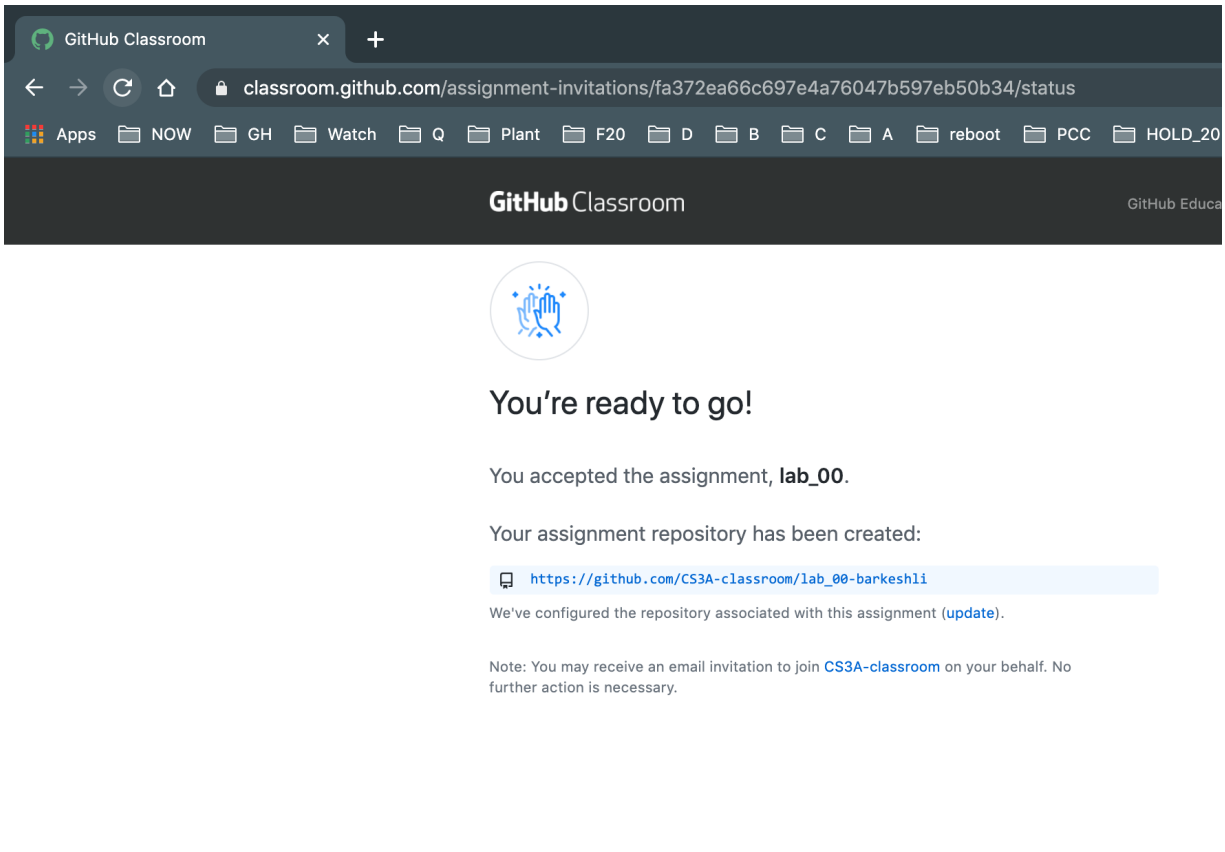

## Your assignment repo being created.

Once you accept the assignment, github will begin to creat your assignment repo. You will see this page:



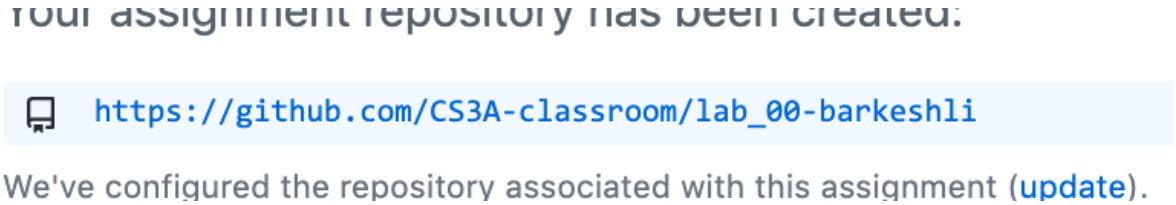

## Your assignment repo:

Give it a few seconds, and reload the page and you should see this:



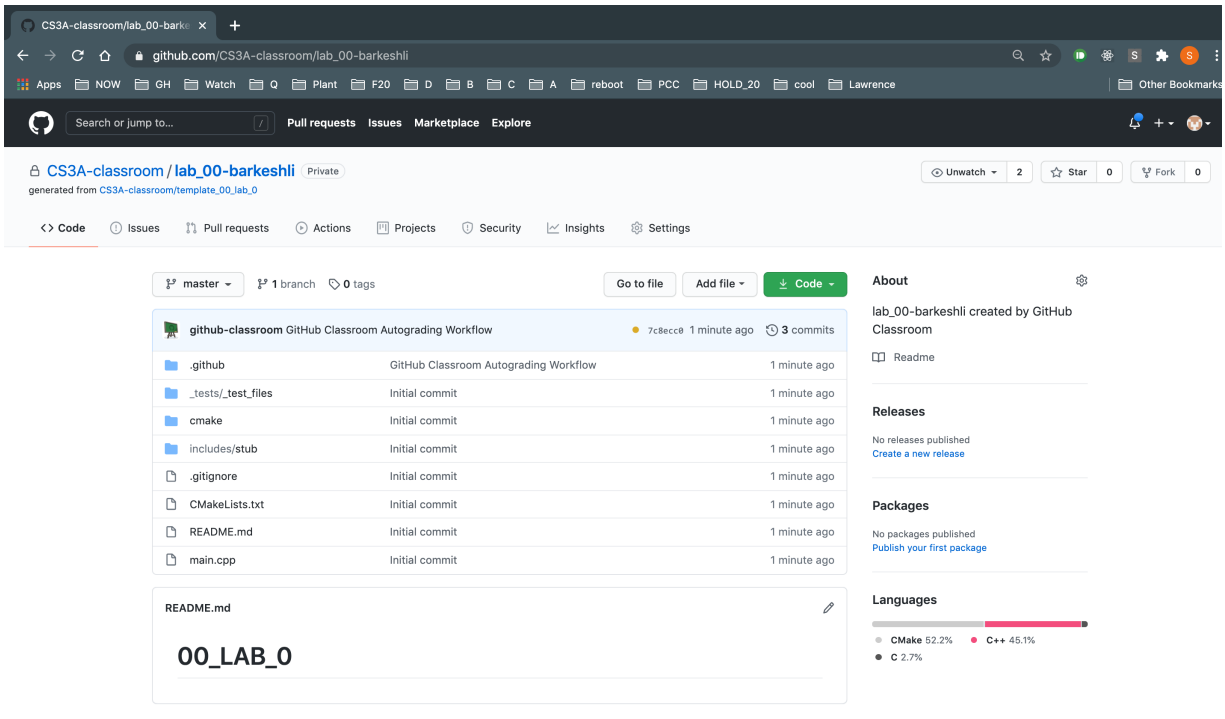

## look closer:

Take a closer look and youwill see the link to your repo. Click it and you will find your assignment repo:




## Your assignment repo:

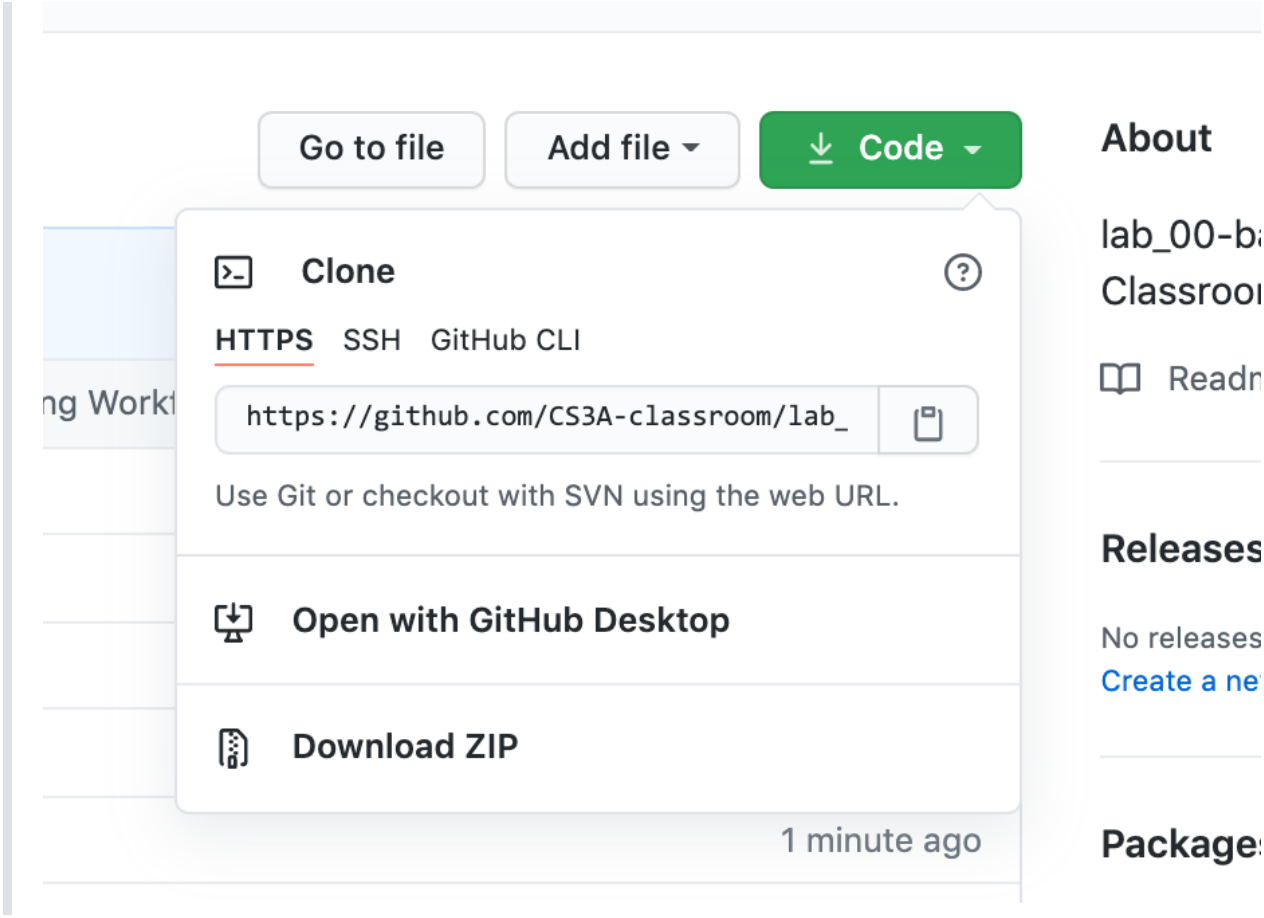
Bookmark this page or know how to get here. We'll need to check in here soon.




# The Code button:

The green button on the mid-right side that says **Code**, click it and that opens this box:

Click the little clipboard and that will copy the link into your clipboard so you can paste it in the next step. You will use this url to clone your repo:




## clone the assignment repo:

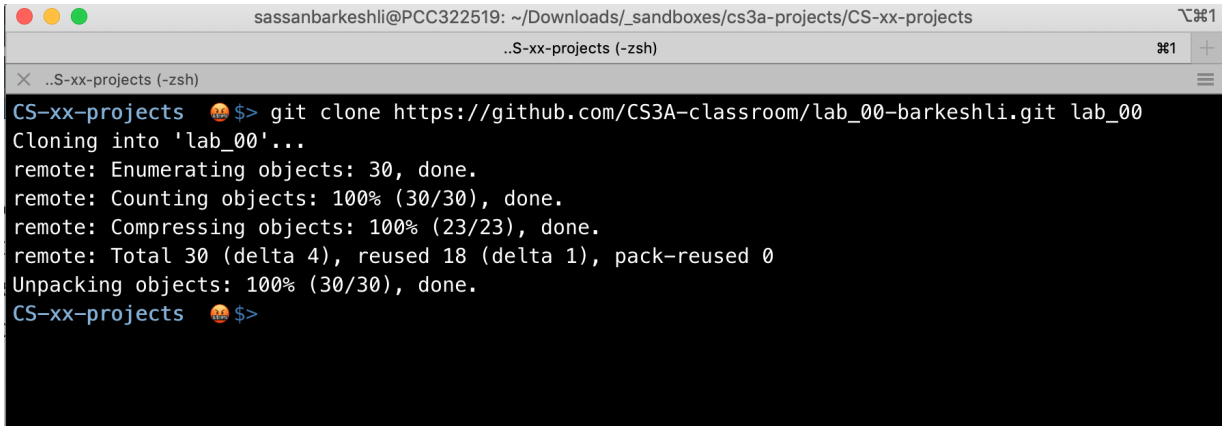
Before you can work on your project, you will need a local copy of the assignment. This is called **cloning** the repository.

First step is to `cd` into the fodler where you will be storing all your projects.

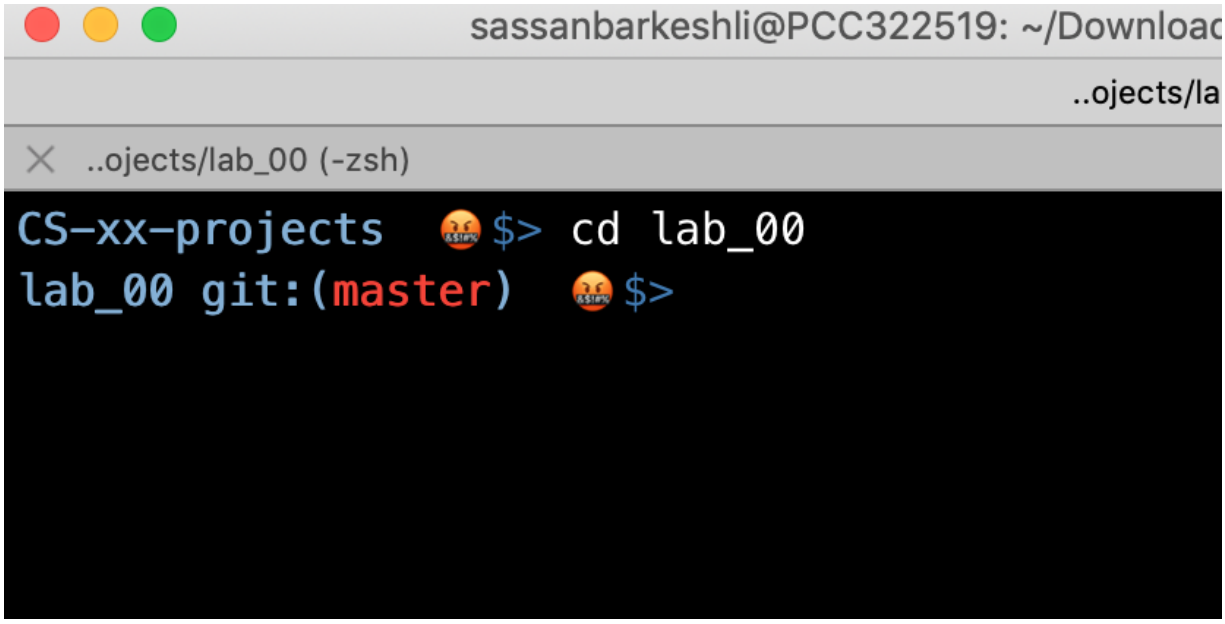
`cd` into your projects folder and clone your project there.

and then: `git clone [clone link] [destination_folder]`

That's what's happening here. I don't like my folder name to be `lab_00_barkeshli` . I like `lab_00` , so, I give it the new name and that clones the project into a folder named `lab_00`

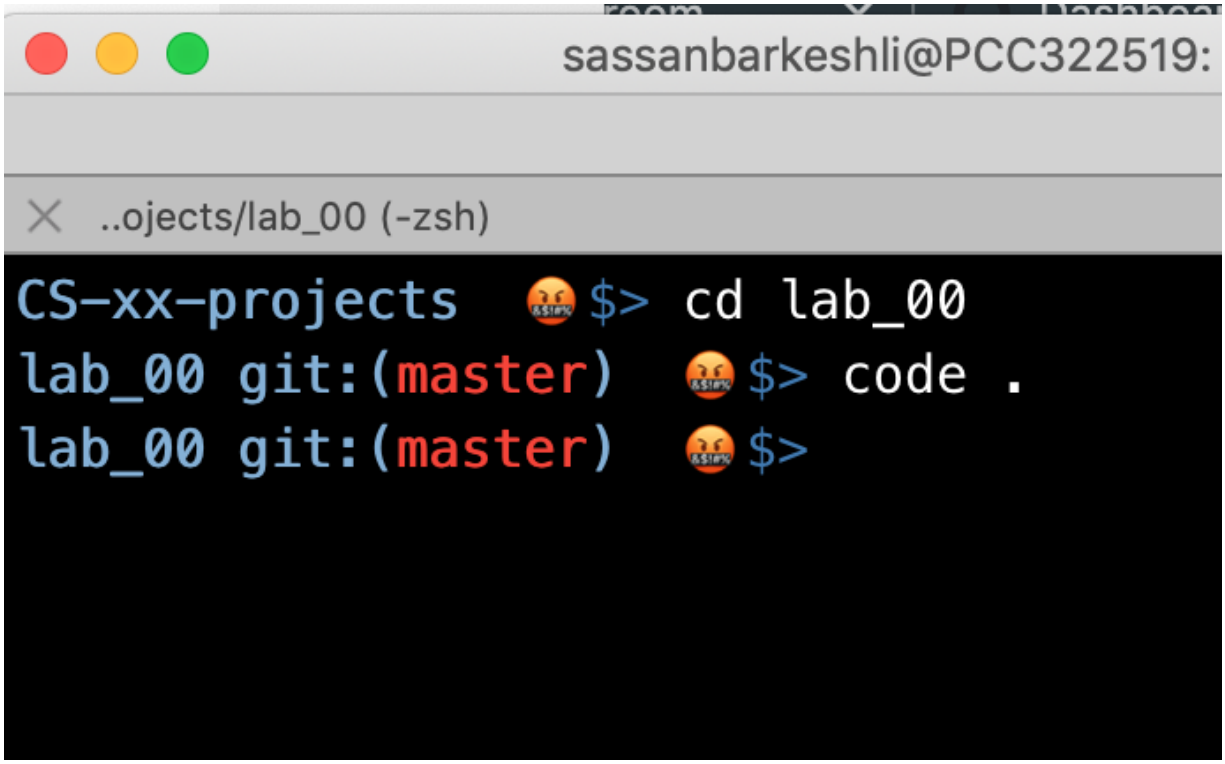



## `cd` into this project folder:



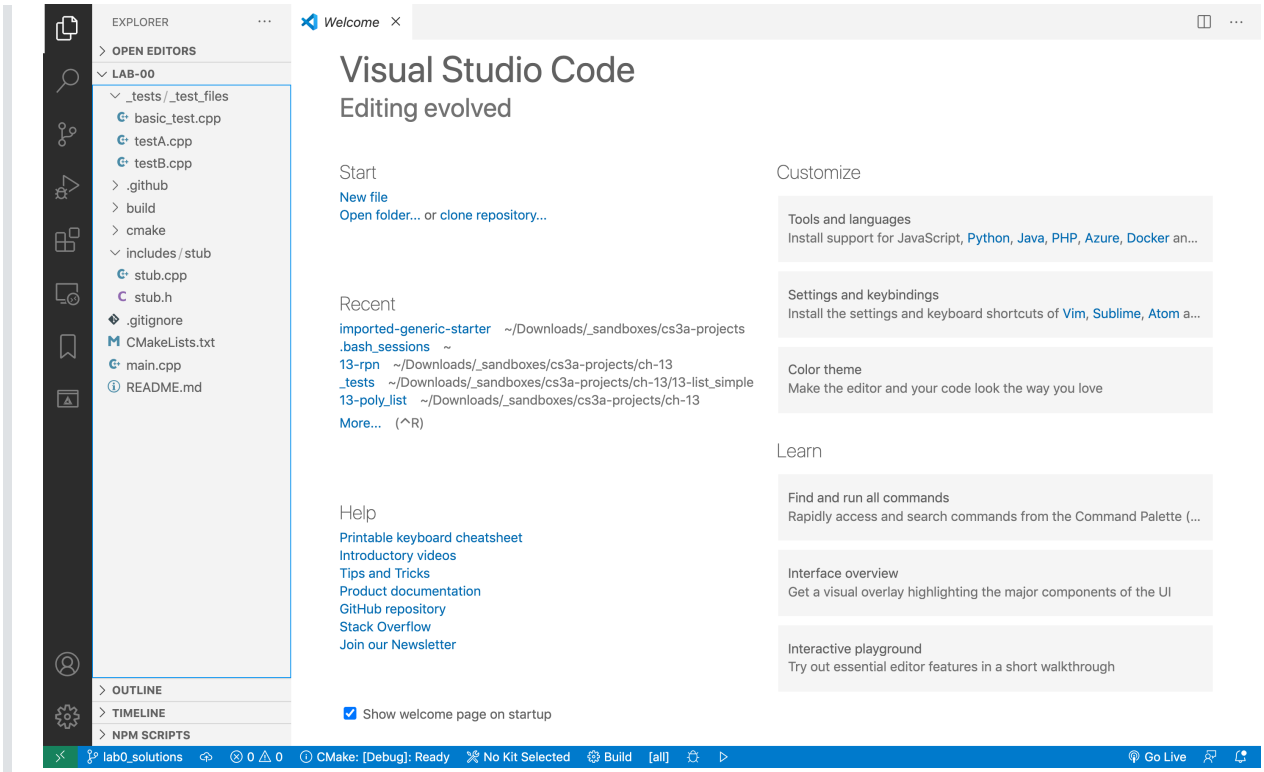

## `code .`:

this will open VSCode and loads the current folder into it.




## Project Organization:

Once you have cloned the project and you open VSCode, this is what you will see:




## File system:

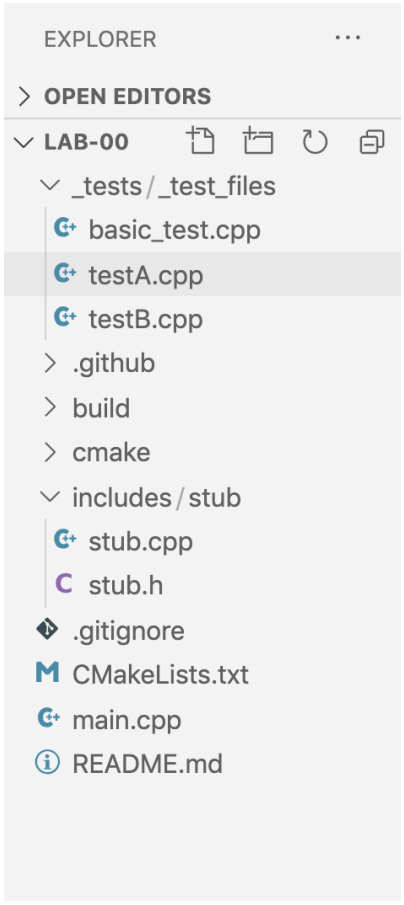
All the projects in this class will follow the same file organization.

On the left panel (Explorer, ) you will find the `main.cpp` on the root folder, and the three most important folders in this directory:

`_tests` : which holds your google test files. The grader will run these files to obtain your score. You will ignore `testA.cpp` for the most part. The bulk of your work will be done in `testB.cpp` .

`includes` : contains a folder for each of the libraries / classes your project depends on. In this starter code, you only have a `stub` folder that contains `stub.h` and `stub.cpp` . These files are `#include` d in `testB.cpp`

`build` : is where you go to build and run your project. This is where all your compiled and executable files will end up.



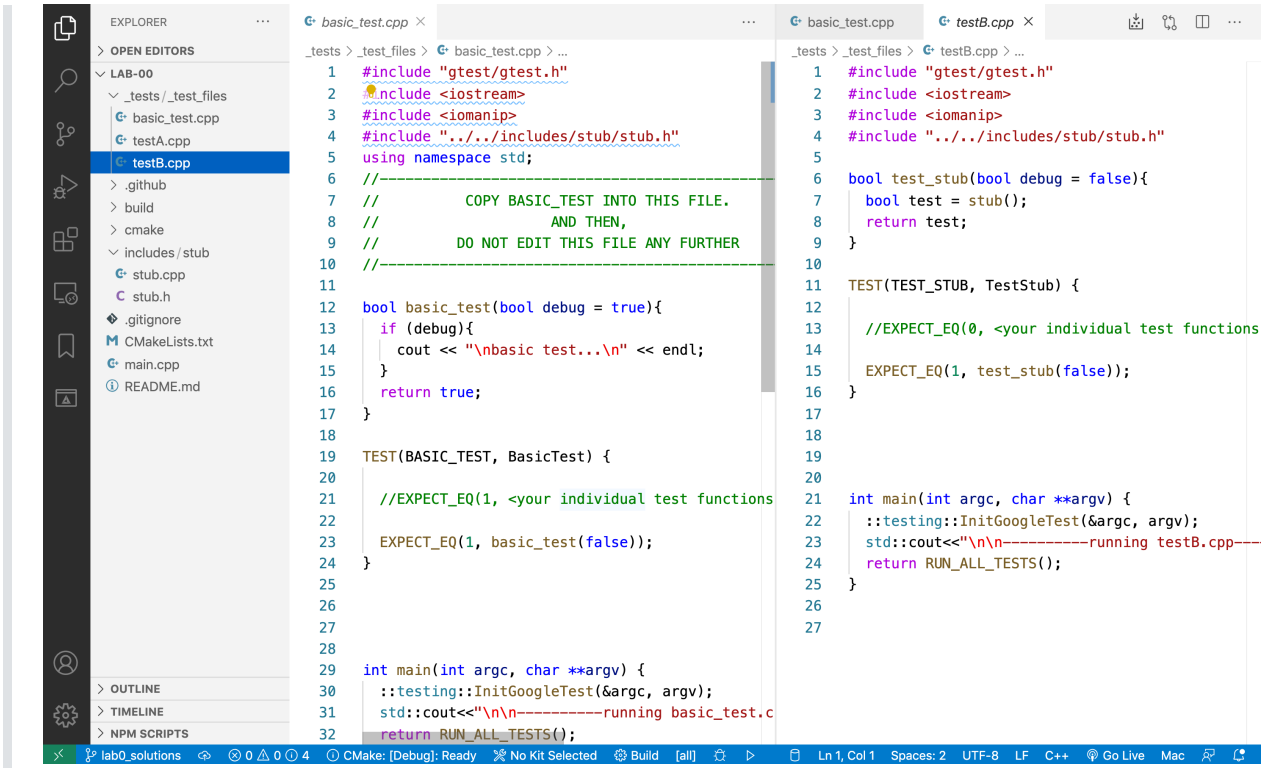

## A look at the test files:

### basic_test.cpp: My sample test goes here.

This is the placeholder file for a sample test file you wilkl be given for each and every project. The purpose of this file is to demonstrate the functionality of the project and for you to make sure that your function signatures and class declarations match the grader's expectations. (otherwise, your projects will not earn a score.)

### testB.cpp: Your tests go here

This is the file that will contain your tests of your own functions and classes. All your test fils that will demonstrate the correctness of your poject are housed here. Part of your grade relies on the quality and success of the tests in this file.




# CMakeLists.txt

The github grader as well as your local Mac or linux systems will use the CMakeLists.txt file to build your project.

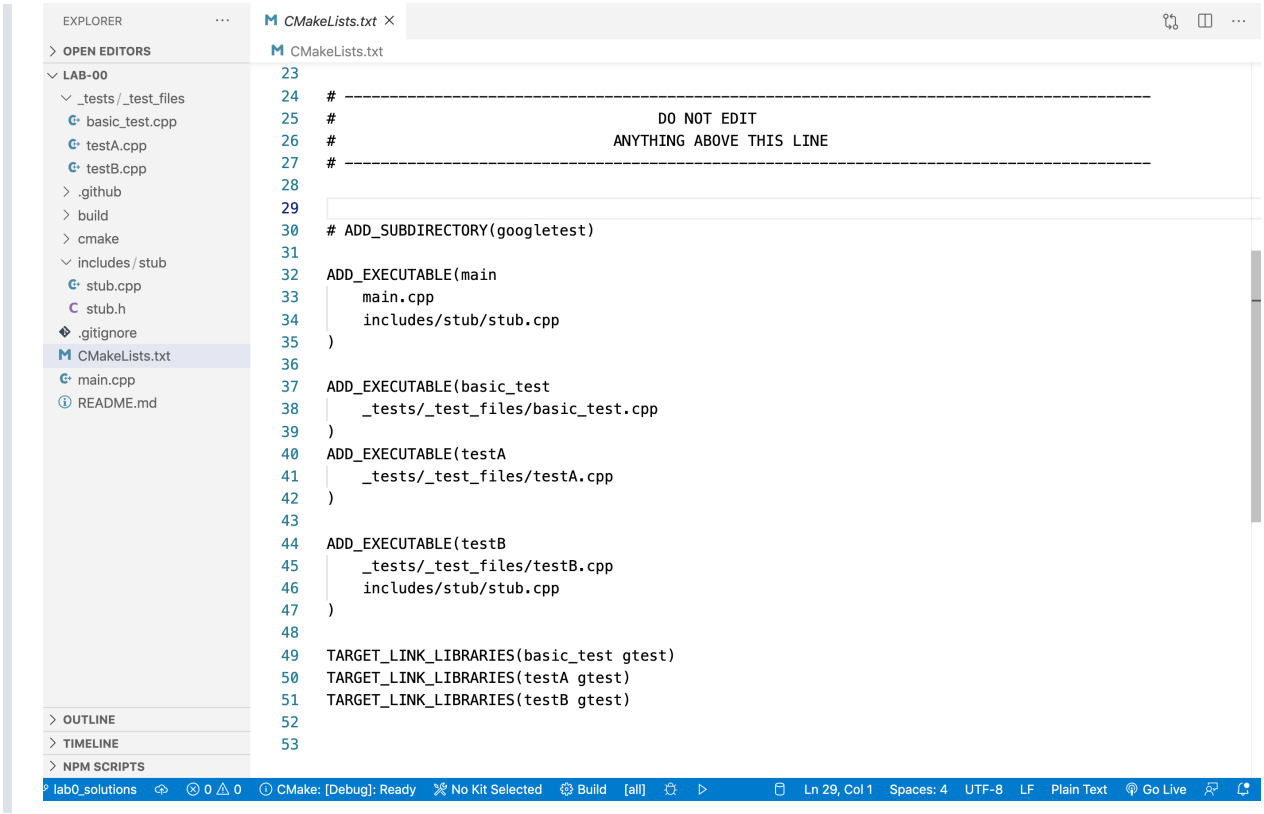
## List your cpp files here

the `CMakeLists.txt` file is what the cmake program looks at to know how to build your project. How the pieces fit together.

When you submit your code, you need to tell it what files are needed to build the test executables.

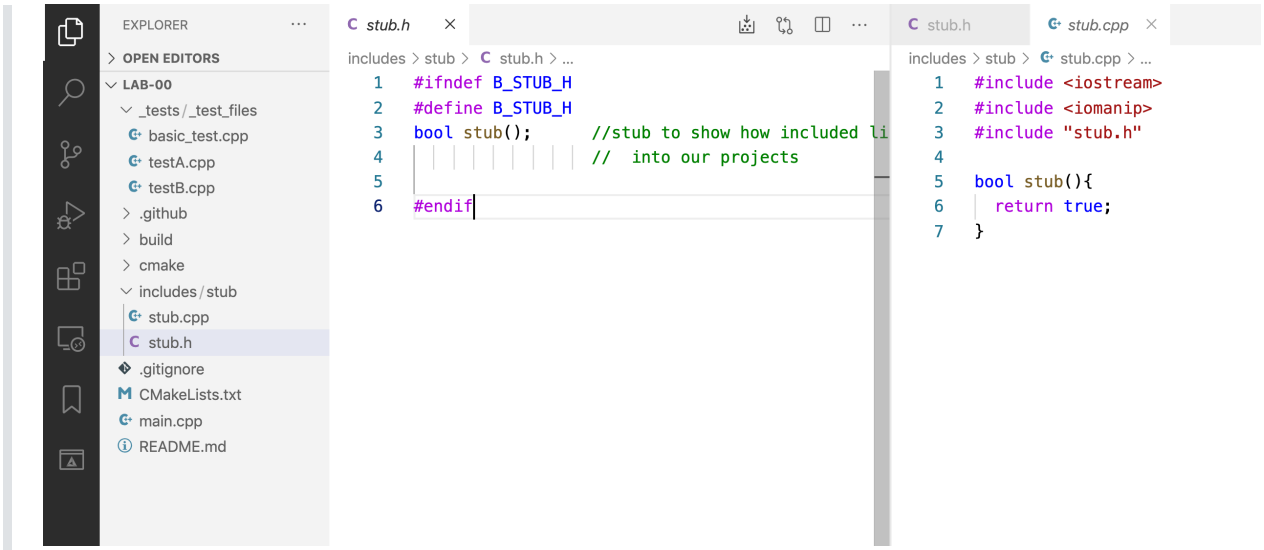
The grader will also use this file to build your project on the server side (once you submit - push your projects to github)

Please note that you will **only** make changes to the bottom half of this file. It's worth mentioning that every **.cpp** file that is used in any of your test files (main, basic_test, testA, testB) will have to be listed here. Notice how the stub.cpp is listed under ADD_EXECUTABLE(testB...)




## stub.h, stub.cpp

Not too much to see here. The stub is used in testB to demonstrate how a function will be tested by the googletest framework in `testB.cpp`. All your functions and classes will be housed under their own folder (`stub/`, `array_functions/`, `vector/`, etc.) which will, in turn, go under the `includes/` folder.



# Quick edit, `status`, `add`, `commit` & `push`

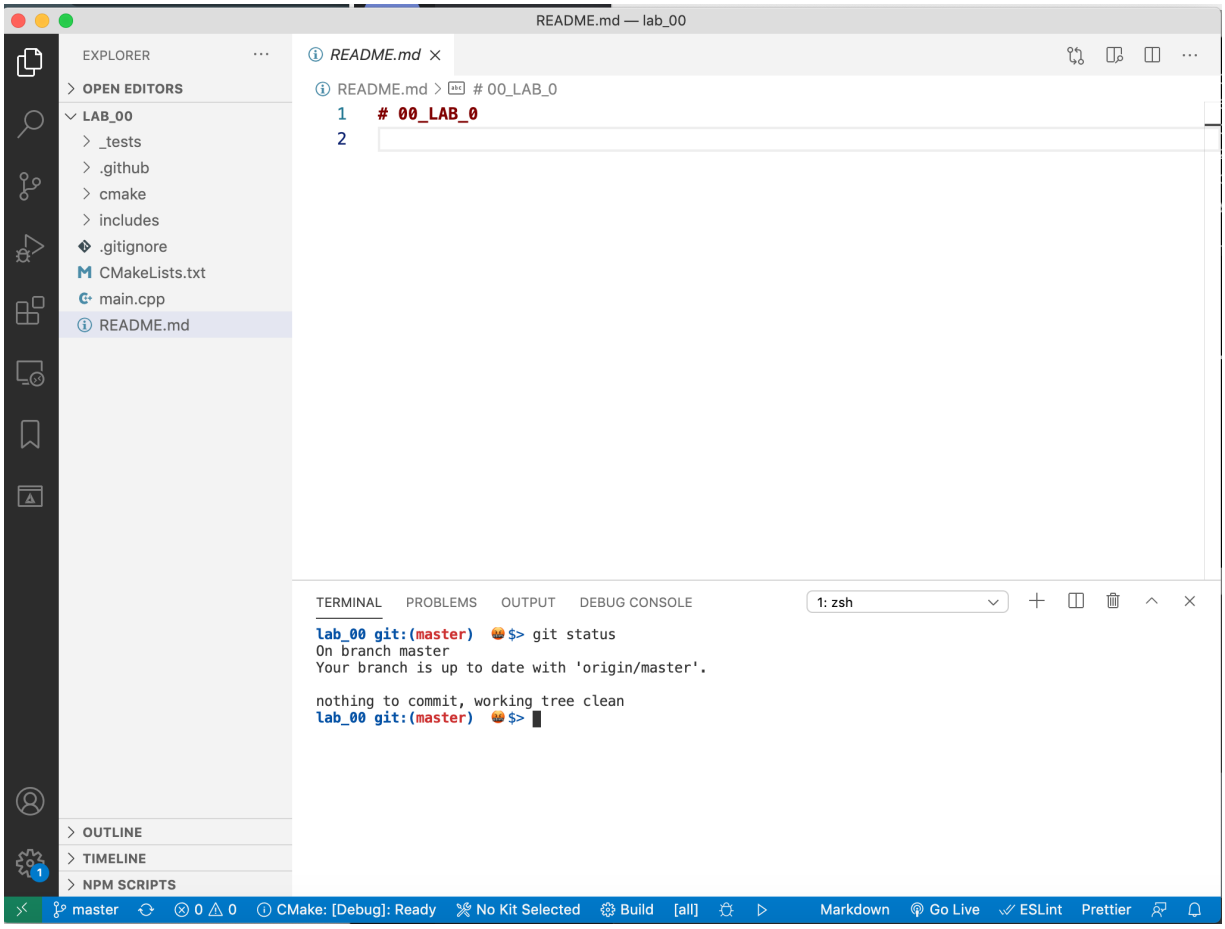
One of the main tasks in this class is tracking changes made to the project. We need to know what happened when and what changed. This is both for your peace of mind (helps you not lose your project accidentally,) and for me to track your progress throught he course.

We use **git** to track changes.

**Here**, we will make some small change to one of our files and walk through the entire process of tracking that change with `git`, just so you can get used to it.

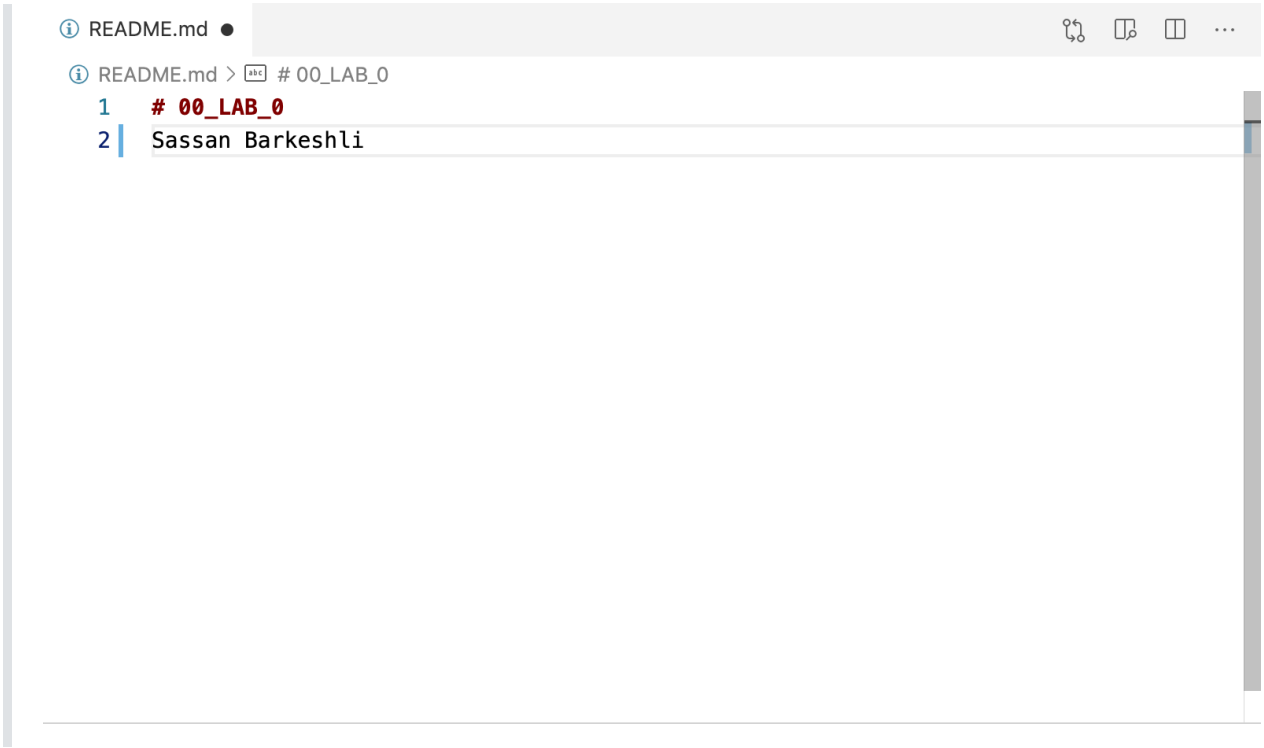
## `git status`

let's run `git status`. look at the response: It says there are not changes as of yet. That's correct, isn't it. We have not made any changes.




# Edit the README.md

Enter your name in the README.md and save it.




## `git status` again:

if you run `git status` again, you will see that git has kept track of our changes. this time, it says that the file `README.md` has changed. But it says that the file is not *staged* for commit.

Think of it this way... There are four zones:

[changes not staged]

`---- git add --->`

[staged changes, but not committed]

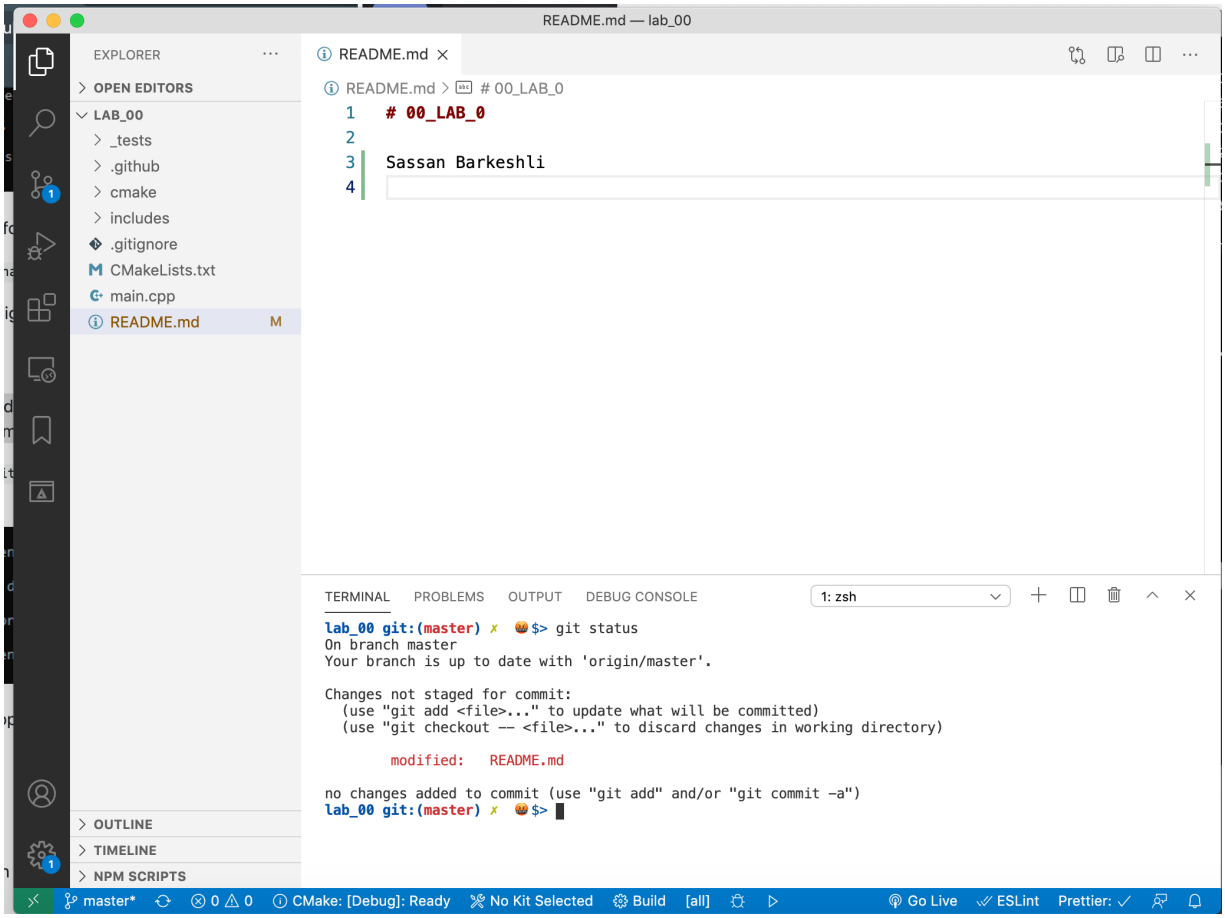
`---- git commit --->`

[committed changes]

`---- git push --->`

[pushed, or synched with remote site]

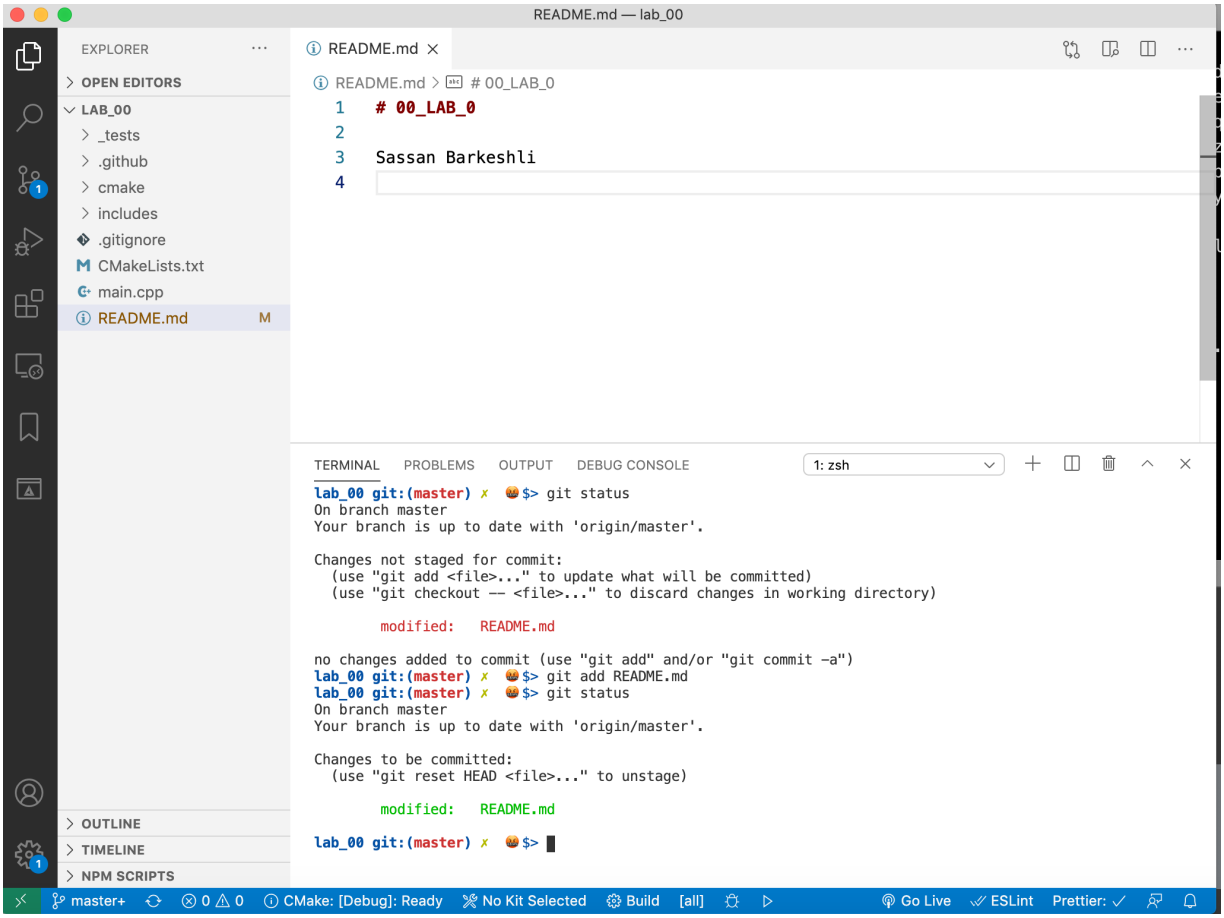
You go from zone to zone by the commands listed above.




## `git add` and then, `git status` again:

In order to *stage* the changes for `commit`, Enter `git add README.md`. This will stage the file.

do `get status` again. Now, you see that `README.md` has been *staged*




## `commit` your changes:

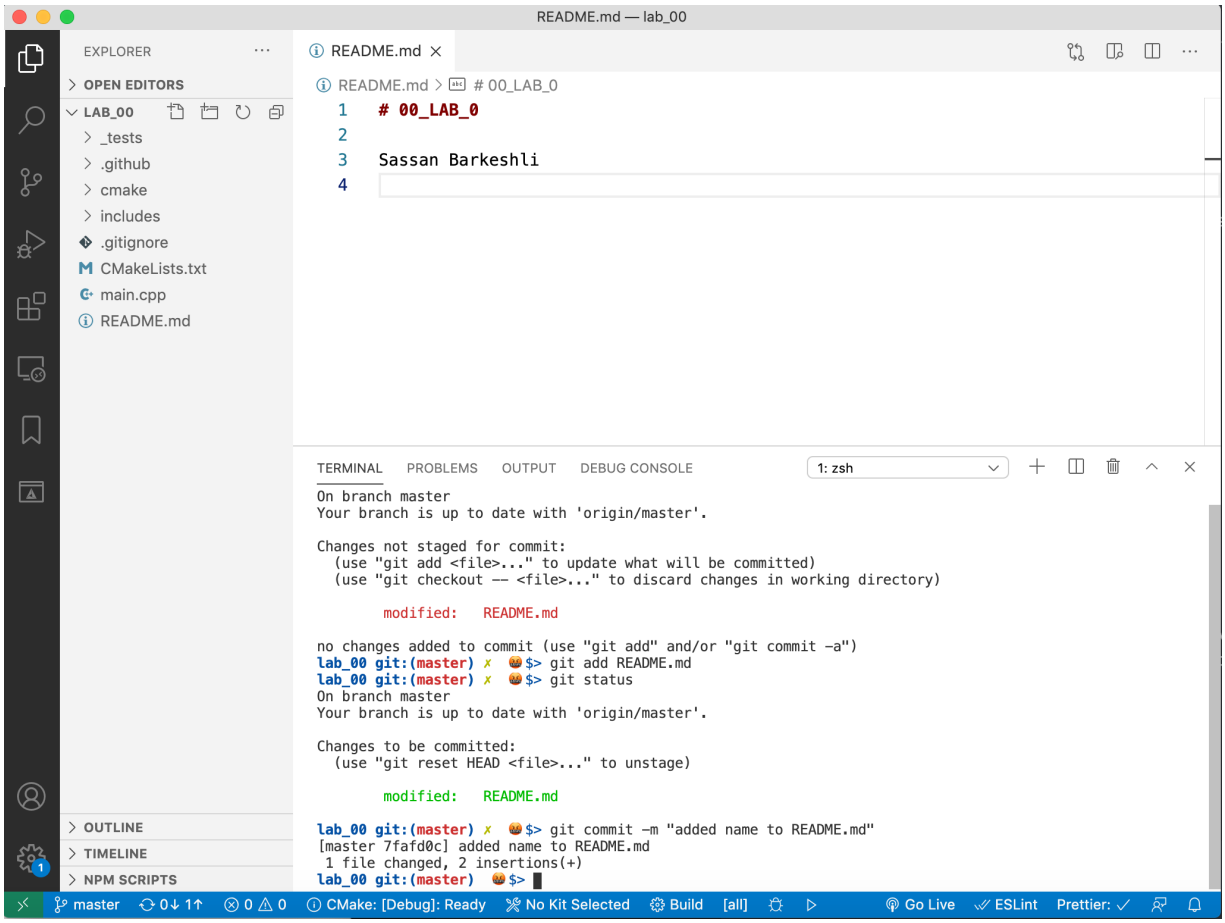
You `commit` your changes to permanently record these changes to `git`. If somehow you ruin your project (as we all have done from time to time,) you can *revert* to this state of your project.

`commit`ting is like *saving* your changes to `git`

Enter `git commit -m "[explain what you just did, in your own words]"`

Here, `-m` is a switch that tells `git` that you will be typing a message to document what this `commit` was for. We want this message to be concise and descriptive of the work that is being recorded.

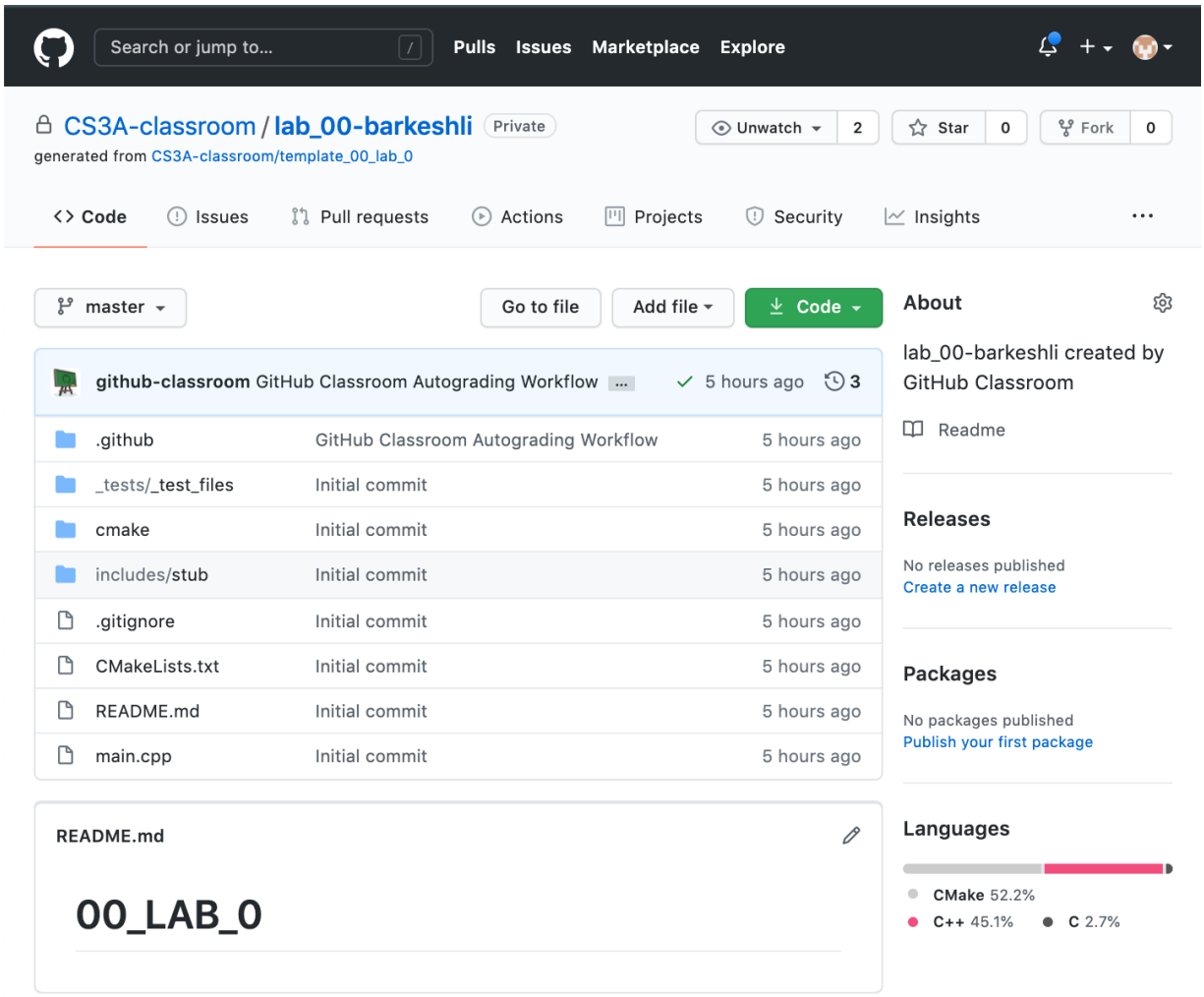
Try typing in your own message in your own words.




## Has Github assignment repo changed?

Take a look at your assignment repo on your browser. You will see that eventhough you have committed your changes, Github does not know about them!

`commit` ting only records your changes on your local machine.




## What is my local `branch` name?

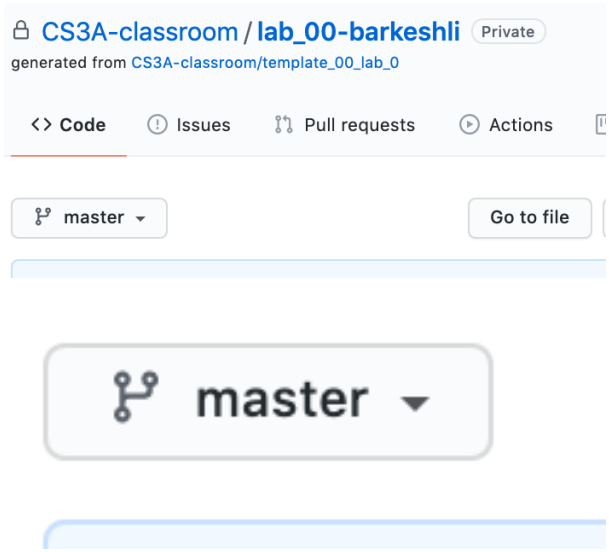
If we want Github to know about our changes, we must `push` them to Github. But before you do, let's find out what our current *branch* is.

For me, the branch name is `master` as evidenced by these images.

Your bash prompt on your terminal:

```
lab_00 git:(master)
```

the branch name on your Github repo page:




>

## `git push` :

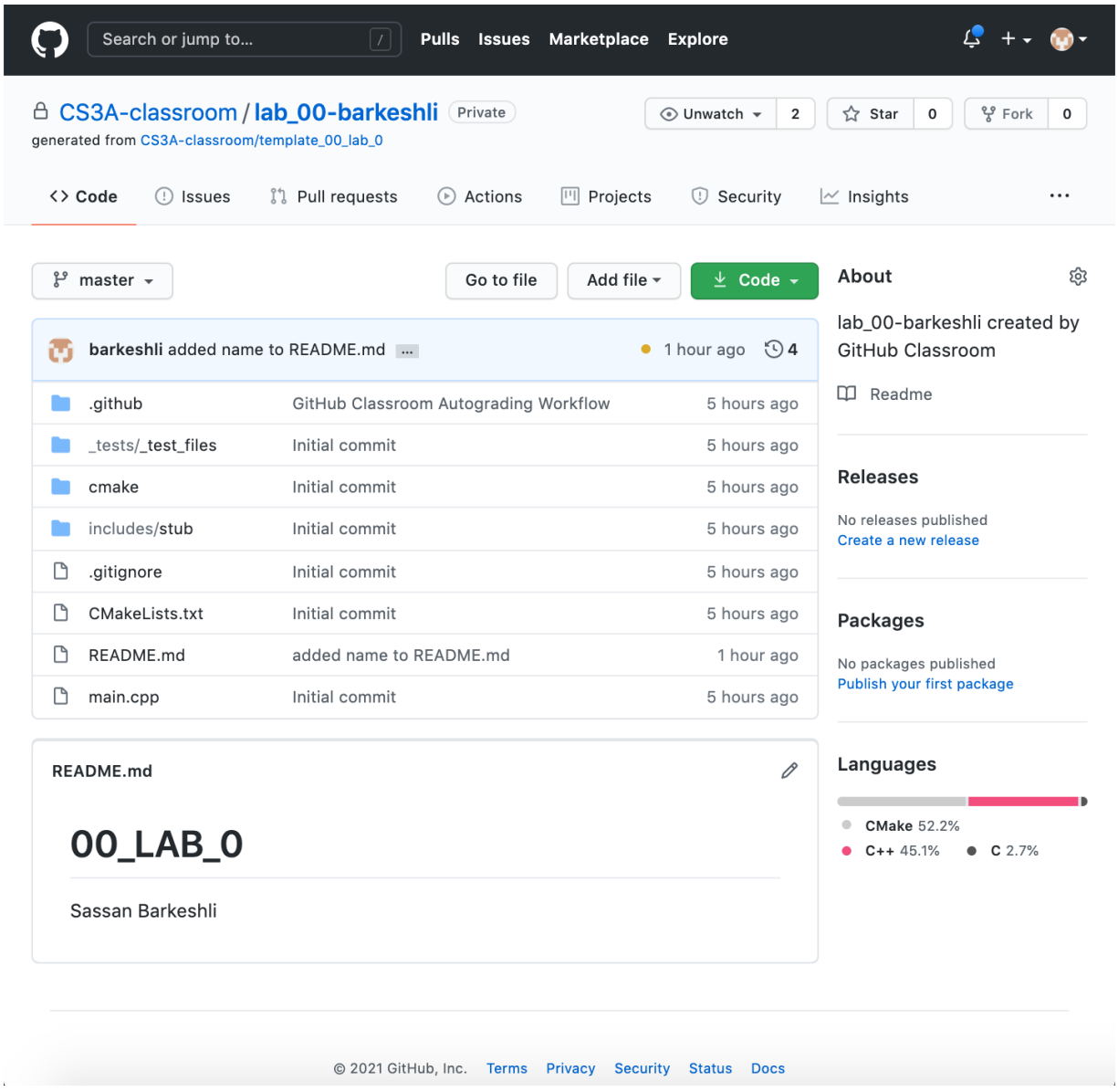
Since my branch name is `master` , I will issue the command `git push origin master` which means push to `origin` , which is my github remote name, from `master` which is my local branch name. For you, your branch namge might be `main` , so, your commmand will be `git push origin main` :

If your command is successful, you will get a response similar to this:

```
lab_00 git:(master)  🤬$> git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 298 bytes | 298.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/CS3A-classroom/lab_00-barkeshli.git
   7c8ecc0..7fafd0c  master -> master
lab_00 git:(master)  🤬$>
```

## Now, Github knows about our changes:

Take a look at your project repo on Github. See how the changes you made to README.md shows on your repo:




## commit and push often:

Make sure you `commit` your changes after completion of **every** task. Remeber, `commit`s are the record of your progress. You will be graded on your commits. And push your changes frequently.

### Bonus: `git log`

Enter `git log` and see what the response is. What does this command do?

## Getting started with the project

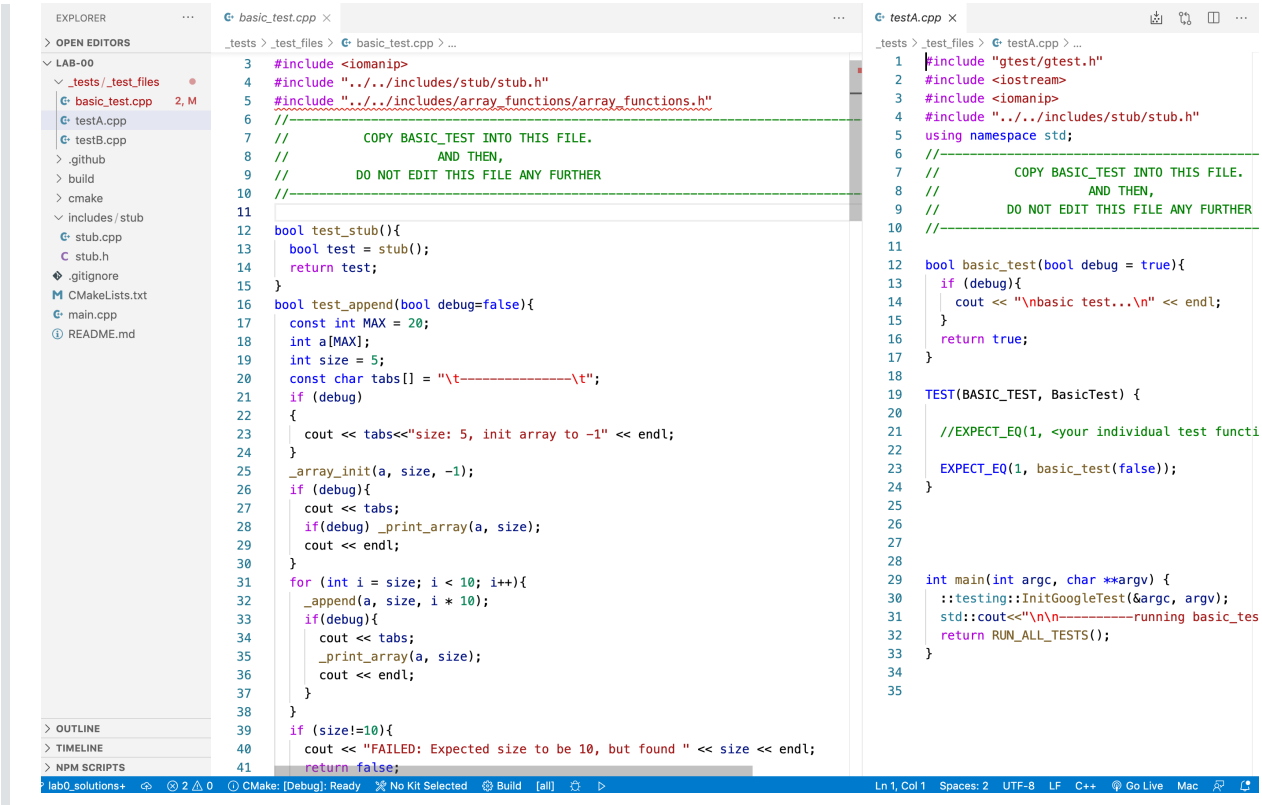
### Find `basic_test.cpp`

You will be supplied with a `basic_test.cpp` file. You will copy this file and overwrite the existing *generic* `basic_test.cpp` in your project folder. After this, you will **never** edit the `basic_test.cpp` file.

`basic_test.cpp` demonstrtes the functionality of the project and gives you an opportunity to make sure your function signatures and class declarations match those of the grader. You should be able to compile and run the `basic_test.cpp` with your functions.

Pay special attention to the `#include` path at the top. Your file structure has to be **exactly** the same as the one depicted here.

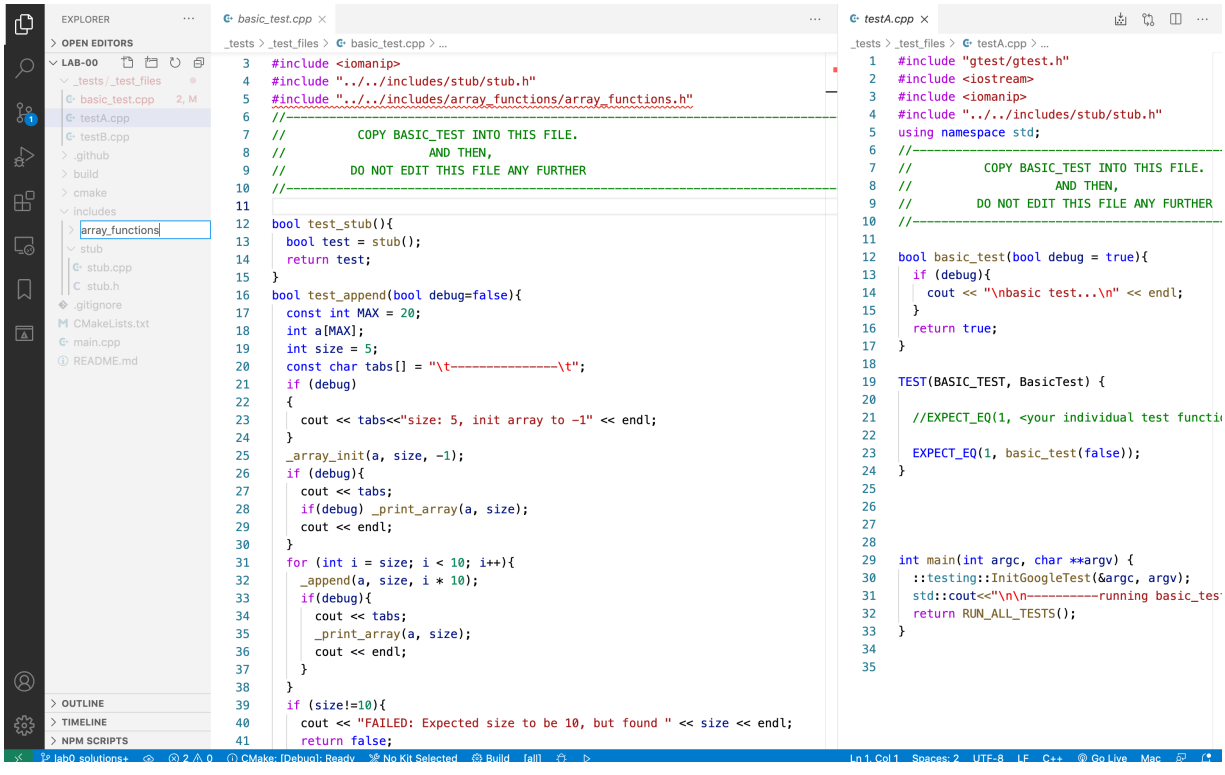
**click here to download basic_test.cpp if you have not already.**




# Add a new folder to the `includes/` folder.

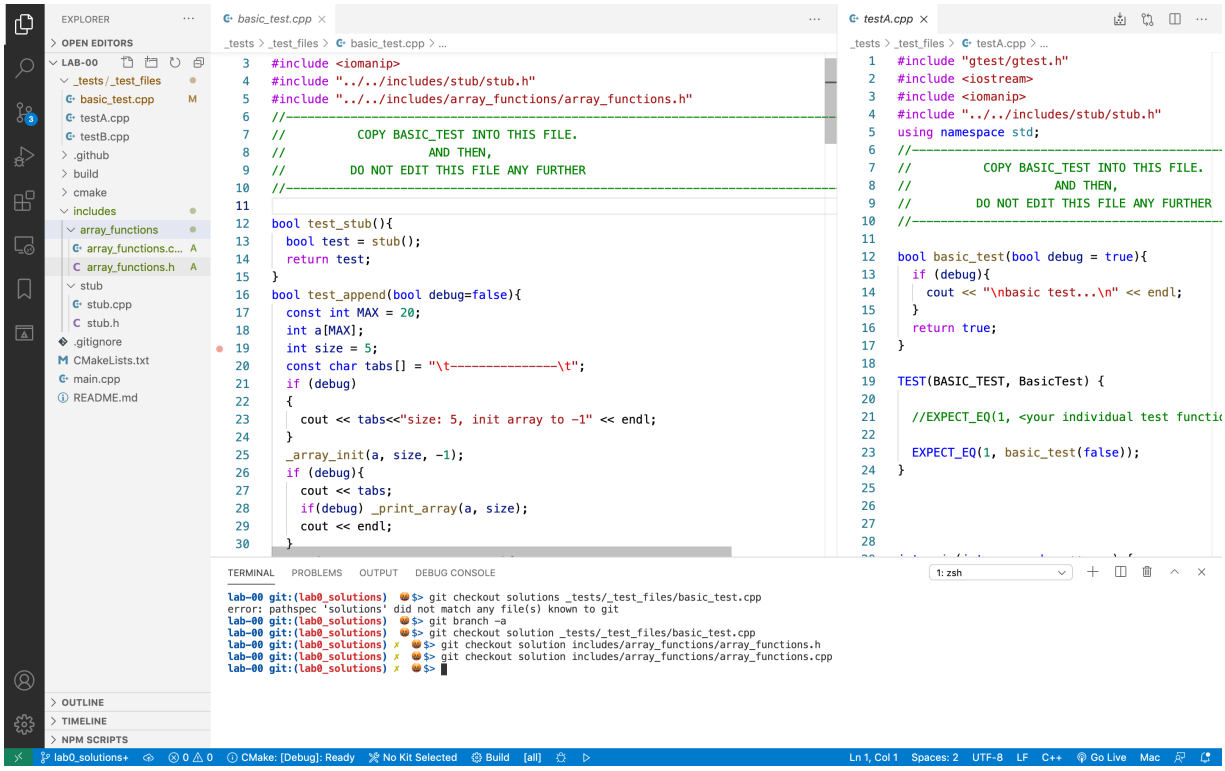
name this folder `array_functions` .

This is where you will add your `.h` and `.cpp` files




# Add two files to this folder.

Name these two files `array_functions.h` and `array_functions.cpp`

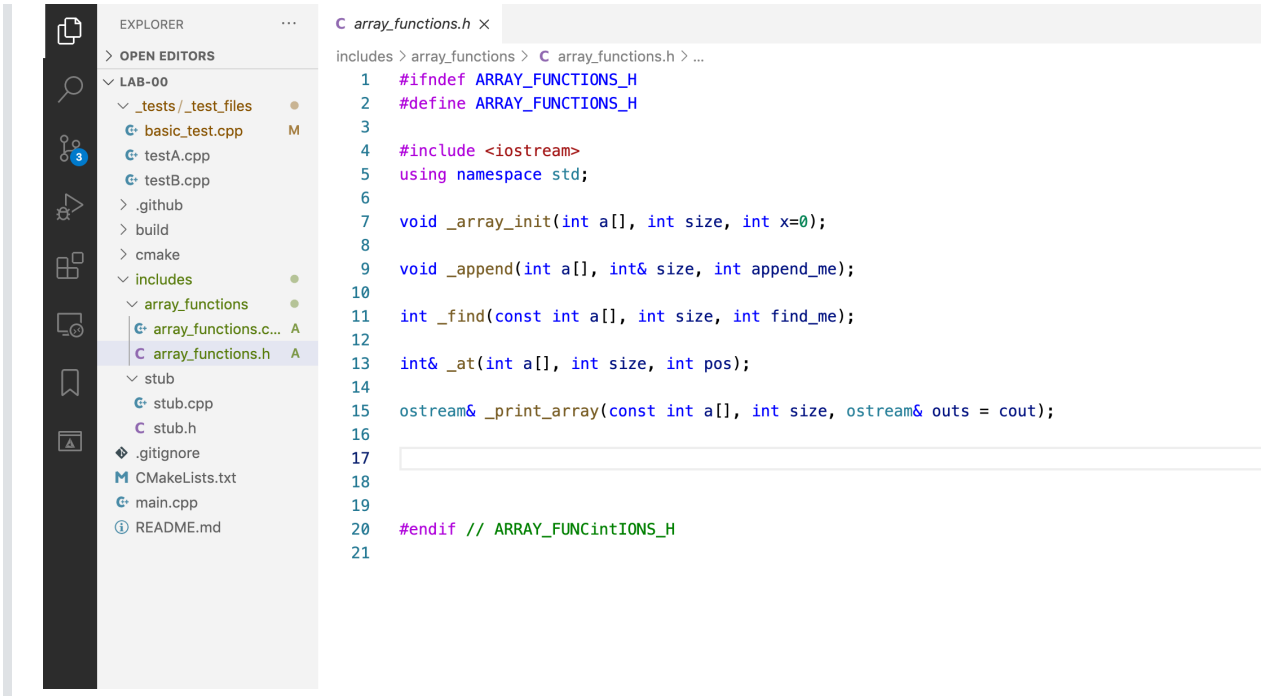


## Add the function signatures.

add these function signatures to the `array_functions.h` file:

```
void _array_init(int a[], int size, int x=0);
void _append(int a[], int& size, int append_me);
int _find(const int a[], int size, int find_me);
int& _at(int a[], int size, int pos);
ostream& _print_array(const int a[], int size, ostream& outs = cout);
```

Normally, you will either be given these function signatures or you will *deduce* them from the code in `basic_test.cpp`



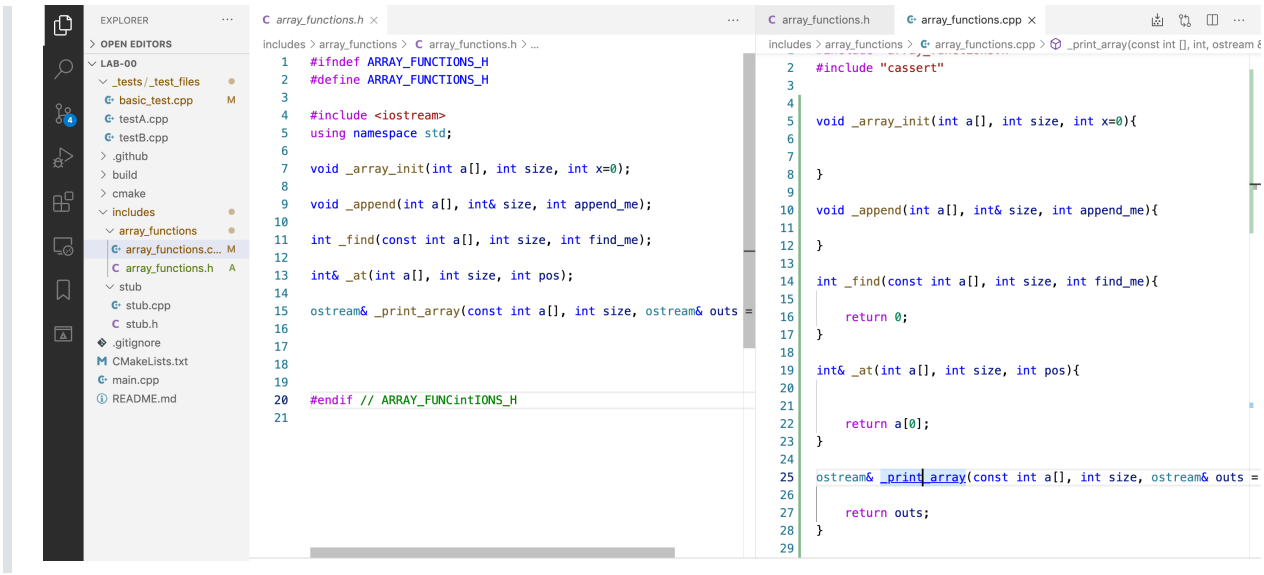

## Write function *stubs*

Function stubs are just function signatures with a return statement if needed.

Function stubs are a quick way to get the project up and running. I find the students who adopt this method in their workflow have an easier time completing projects.

### TIP:

I normally copy the function signatures and paste them into the `.cpp` file. Then, I replace the `;` at the end of the line with braces (`{}`). Then, I add the returns whenever necessary.

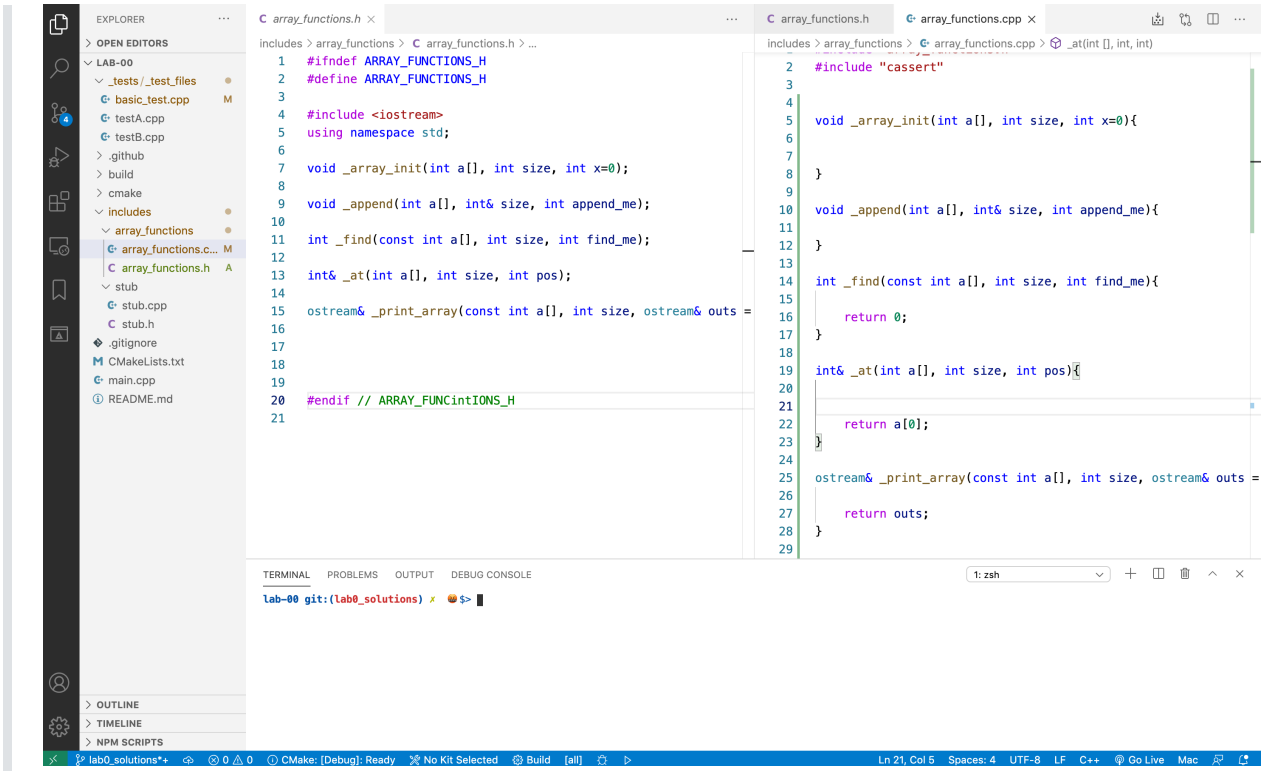



# Open terminal:

If you are using VSCode, you can open the terminal by pressing [ctrl][`]

[`] is the key in the top left of the keyboard under [~]

Using the terminal in this way is very convenient.



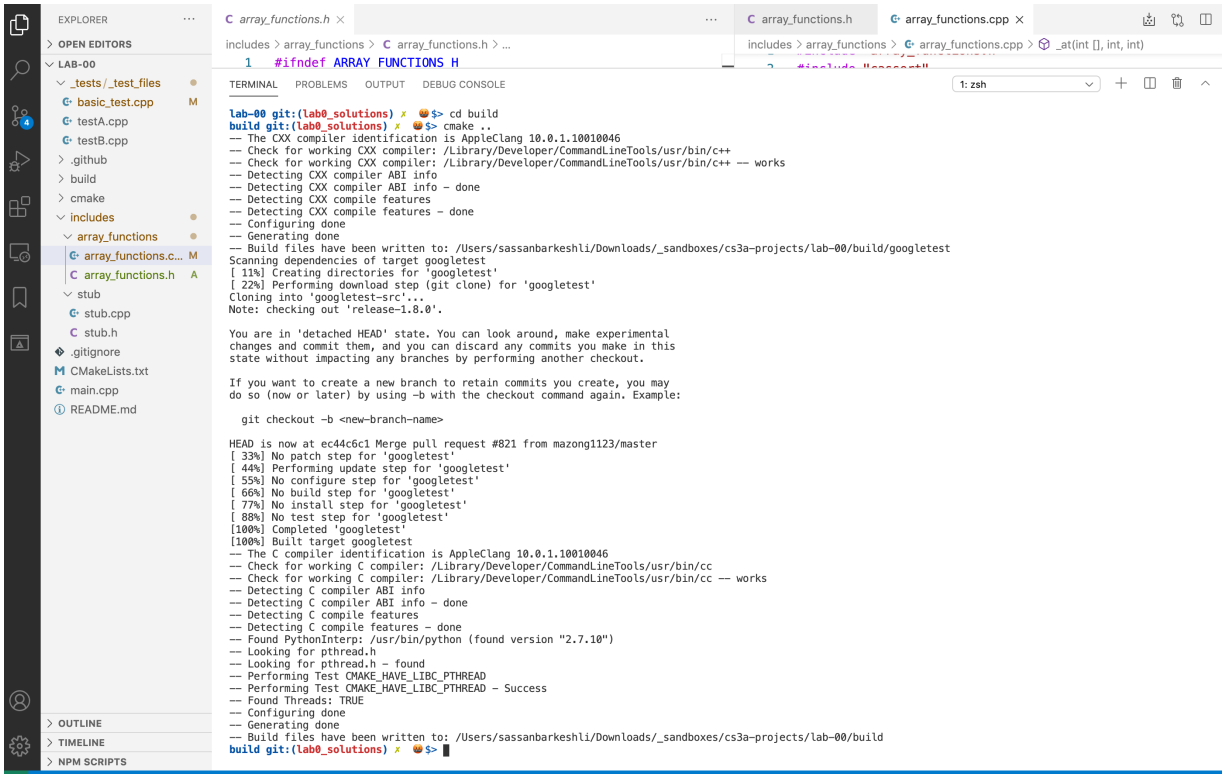

## Go to `build/` and run `cmake`:

`cd` into the `build/` folder, and from there, run `cmake ..`

This will run `cmake` on your *parent* folder. (that's what `..` means. `cmake ..` means run `cmake` on my parent folder.) `cmake` creates a bunch of files and you do not want these files in the root folde of your project. Running `cmake` from `build/` will make sure all your auxilary files are created inside the `build/` folder.

Hopefully, your `cmake` will run without any problems and it will tell you that "Build files are written to `. . . build/`"

Now, we are ready to compile our project using `make`

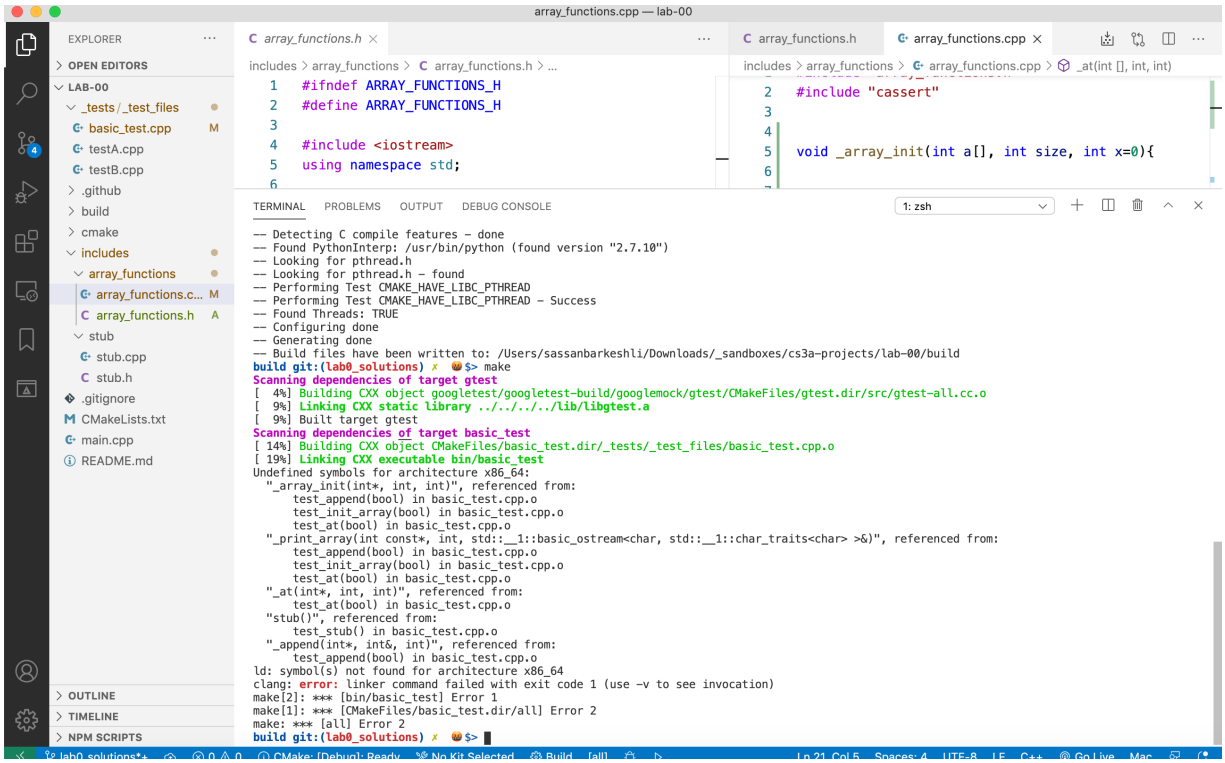



## make

type `make` to compile your project:

If you followed these steps faithfully, you will have the same syntax errors that I had, namely that all those functions we defined in `array_functions.h` and `.cpp` are **undefined!**

We will spare you the suspense. The reason for this error is that we never added `array_functions.cpp` to our `CMakeLists.txt`. Remember that **all .cpp** files must be listed in the CMakeLists.txt under `ADD_EXECUTABLE`




## `make` errors, zoomed in:

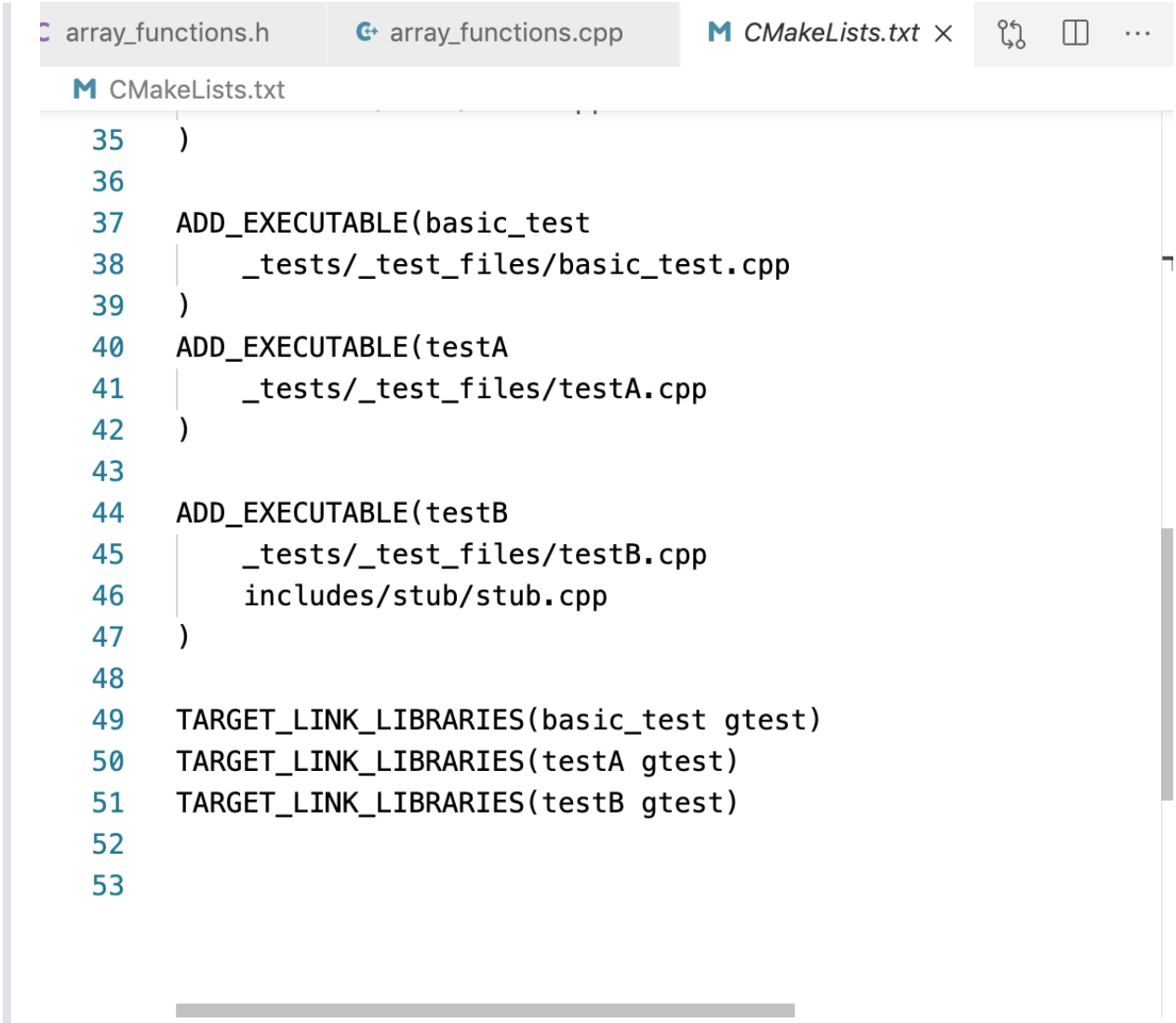
Here is a closer, more readable look at the errors reported by `make`

```
build git:(lab0_solutions) ✗ $> make
Scanning dependencies of target gtest
[ 4%] Building CXX object googletest/googletest-build/googlemock/gtest/CMakeFiles/gtest.dir/src/gtest-all.cc.o
[ 9%] Linking CXX static library ../../../../lib/libgtest.a
[ 9%] Built target gtest
Scanning dependencies of target basic_test
[ 14%] Building CXX object CMakeFiles/basic_test.dir/_tests/_test_files/basic_test.cpp.o
[ 19%] Linking CXX executable bin/basic_test
Undefined symbols for architecture x86_64:
  "_array_init(int*, int, int)", referenced from:
      test_append(bool) in basic_test.cpp.o
      test_init_array(bool) in basic_test.cpp.o
      test_at(bool) in basic_test.cpp.o
  "_print_array(int const*, int, std::__1::basic_ostream<char, std::__1::char_traits<char> >&)", referenced from:
      test_append(bool) in basic_test.cpp.o
      test_init_array(bool) in basic_test.cpp.o
      test_at(bool) in basic_test.cpp.o
  "_at(int*, int, int)", referenced from:
      test_at(bool) in basic_test.cpp.o
  "stub()", referenced from:
      test_stub() in basic_test.cpp.o
  "_append(int*, int&, int)", referenced from:
      test_append(bool) in basic_test.cpp.o
ld: symbol(s) not found for architecture x86_64
clang: error: linker command failed with exit code 1 (use -v to see invocation)
make[2]: *** [bin/basic_test] Error 1
make[1]: *** [CMakeFiles/basic_test.dir/all] Error 2
make: *** [all] Error 2
build git:(lab0_solutions) ✗ $>
```

## back at the `CMakeLists.txt` :

Notice that we are missing the `array_functions.cpp` from the `basic_test` `ADD_EXECUTABLE` statement:

So, let's add it...

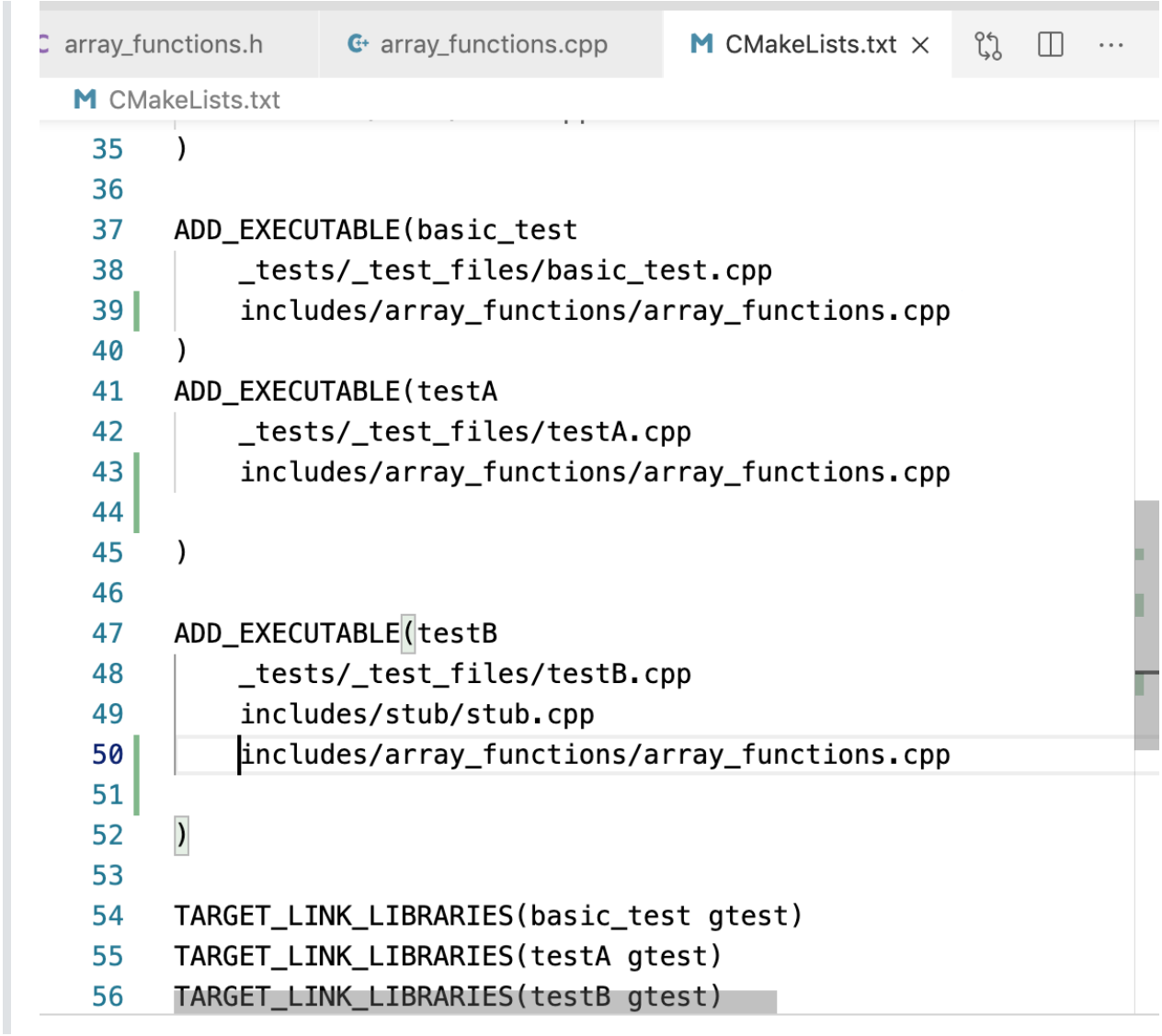



## Add `array_functions.cpp` to the `ADD_EXECUTABLE(basic_test... )`

**Do not** use commas to separate the files.

**Do NOT** include `.h` files.

Normally, **all** three executables will need all the `.cpp` files




## `make` again:

Let's run `make` again and pray that...

... and, we have more syntax errors. Default arguments can only be specified in the declration of the function and **not** in the definition.

So, we must remove all those default values for the defalut arguments on every function.

```
build git:(lab0_solutions) ✗  $> make
[  8%] Built target gtest
Scanning dependencies of target basic_test
[ 12%] Building CXX object CMakeFiles/basic_test.dir/includes/array_functions/array_functions.cpp.o
/Users/sassanbarkeshli/Downloads/_sandboxes/cs3a-projects/lab-00/includes/array_functions/array_functions.cpp:5:41: error: redefinition of default argument
void _array_init(int a[], int size, int x=0){
                                        ^ ~
/Users/sassanbarkeshli/Downloads/_sandboxes/cs3a-projects/lab-00/includes/array_functions/array_functions.h:7:41: note: previous definition is here
void _array_init(int a[], int size, int x=0);
                                        ^ ~
/Users/sassanbarkeshli/Downloads/_sandboxes/cs3a-projects/lab-00/includes/array_functions/array_functions.cpp:25:57: error: redefinition of default argument
ostream& _print_array(const int a[], int size, ostream& outs = cout){
                                                        ^      ~~~~
/Users/sassanbarkeshli/Downloads/_sandboxes/cs3a-projects/lab-00/includes/array_functions/array_functions.h:15:57: note: previous definition is here
ostream& _print_array(const int a[], int size, ostream& outs = cout);
                                                        ^      ~~~~
2 errors generated.
make[2]: *** [CMakeFiles/basic_test.dir/includes/array_functions/array_functions.cpp.o] Error 1
make[1]: *** [CMakeFiles/basic_test.dir/all] Error 2
make: *** [all] Error 2
build git:(lab0 solutions) ✗  $>
```

# Fix the `_print_array` function...

```
24
25  ostream& _print_array(const int a[], int size, ostream& outs = cout){
26
27      return outs;
28  }
29
```

# by removing the default value `= cout`

array_functions.h | array_functions.cpp × | CMakeLists.txt

includes > array_functions > array_functions.cpp > _print_array(const int [], int, ostream &)

```
17  }
18
19  int& _at(int a[], int size, int pos){
20
21
22      return a[0];
23  }
24
25  ostream& _print_array(const int a[], int size, ostream& outs){
26
27      return outs;
28  }
29
```

# Same with `_array_init` :

```
4
5  void _array_init(int a[], int size, int x=0){
6
7
8  }
```

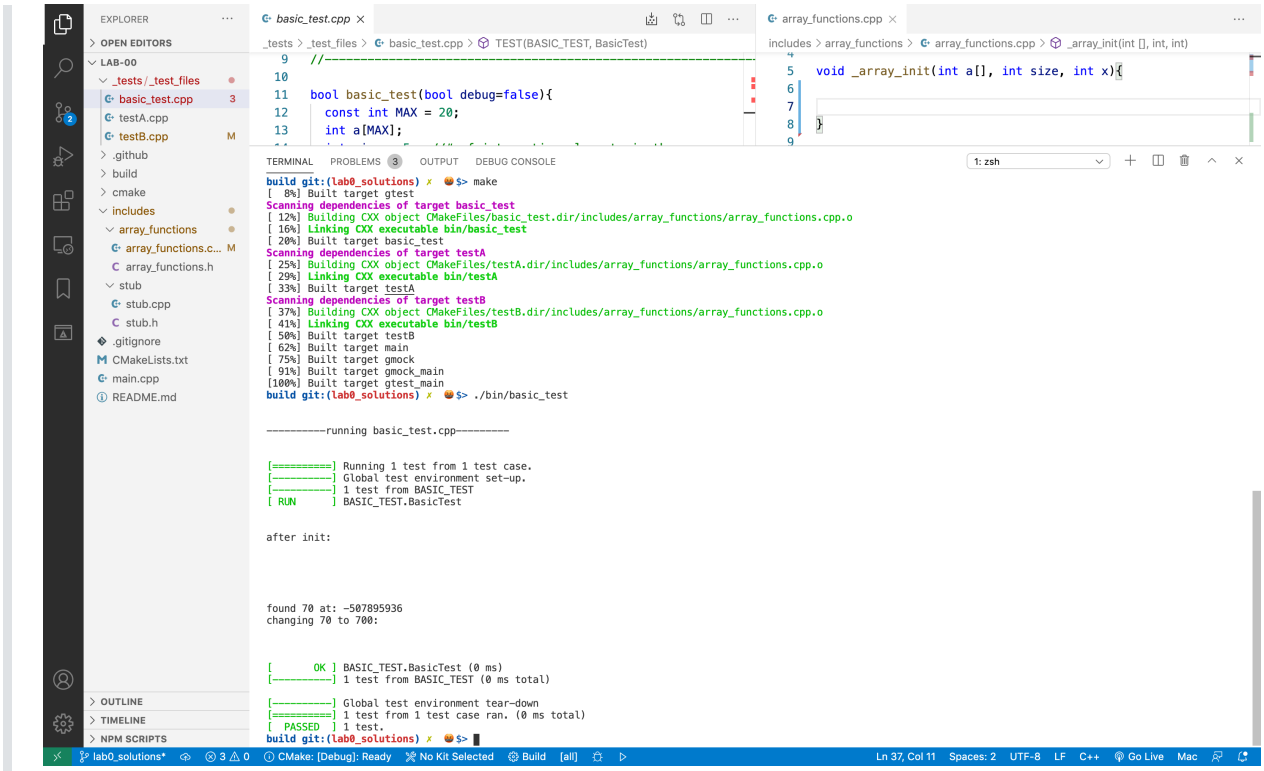
```
4
5  void _array_init(int a[], int size, int x){
6
7
8  }
9
```

## `make` one more time:

and this time it will run successfully.

This is a huge step. We now have a working project eventhough our functions are basically empty.

You can even run the `basic_test` from the `bin/` directory. Of course this will not run satisfactorily. You will get mostly garbage. -afterall, we are running on stubs!- but it **does** run!!



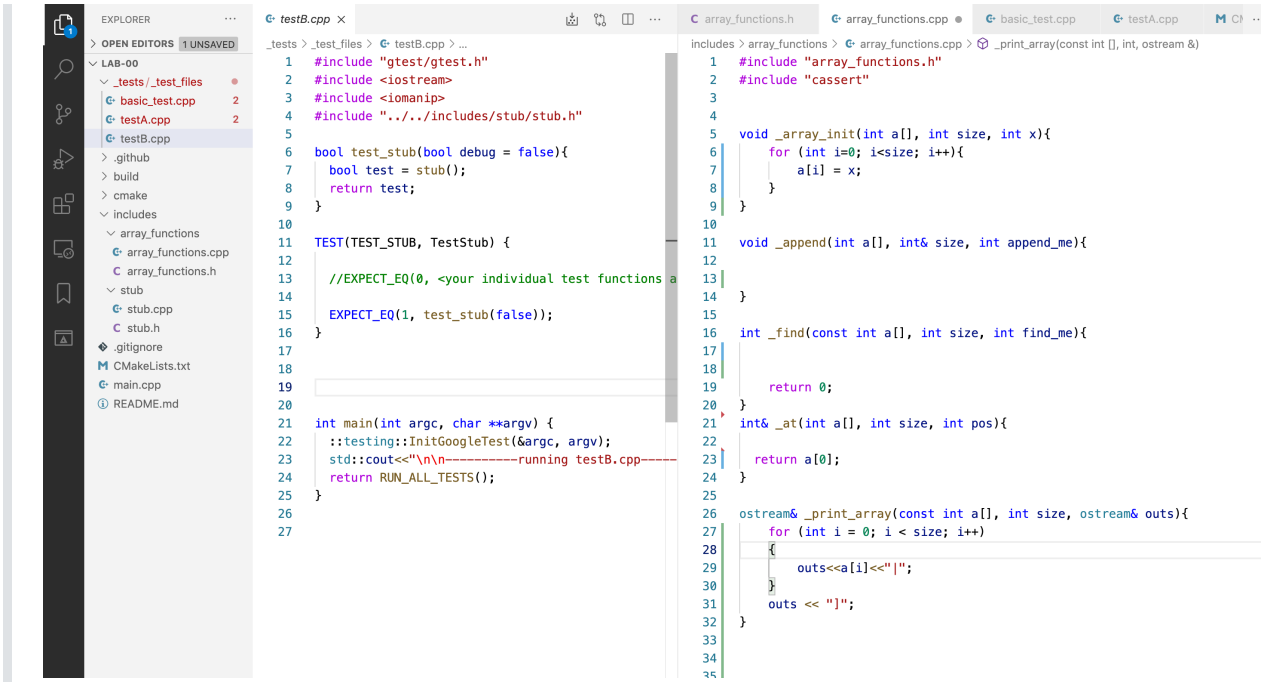

## run `git status`, `add`, and `commit` with the message *success on make with stubs*

The importance of having regular `commit`s in your project cannot be overstated. This is a large part of the evaluation of your project by me.

```
[100%] Built target gtest_main
build git:(lab0_solutions) ✗  $> cd ..
lab-00 git:(lab0_solutions) ✗  $> git status
On branch lab0_solutions
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   _tests/_test_files/basic_test.cpp
        new file:   includes/array_functions/array_functions.cpp
        new file:   includes/array_functions/array_functions.h

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   CMakeLists.txt
        modified:   _tests/_test_files/basic_test.cpp
        modified:   _tests/_test_files/testA.cpp
        modified:   includes/array_functions/array_functions.cpp

lab-00 git:(lab0_solutions) ✗  $> git add .
lab-00 git:(lab0_solutions) ✗  $> git commit -m "success on make with stubs"
[lab0_solutions 4c5b80c] success on make with stubs
 5 files changed, 286 insertions(+), 12 deletions(-)
 create mode 100644 includes/array_functions/array_functions.cpp
 create mode 100644 includes/array_functions/array_functions.h
lab-00 git:(lab0_solutions)  $>
```

## Implement `_array_init` and `_print_array`. `testB.cpp` can be seen waiting to host the test functions.

Now, we can go in and implement the functions one by one and write tests for them. These tests will be written in the `testB.cpp` file.



# Writing Tests:

## `testB` : our first test:

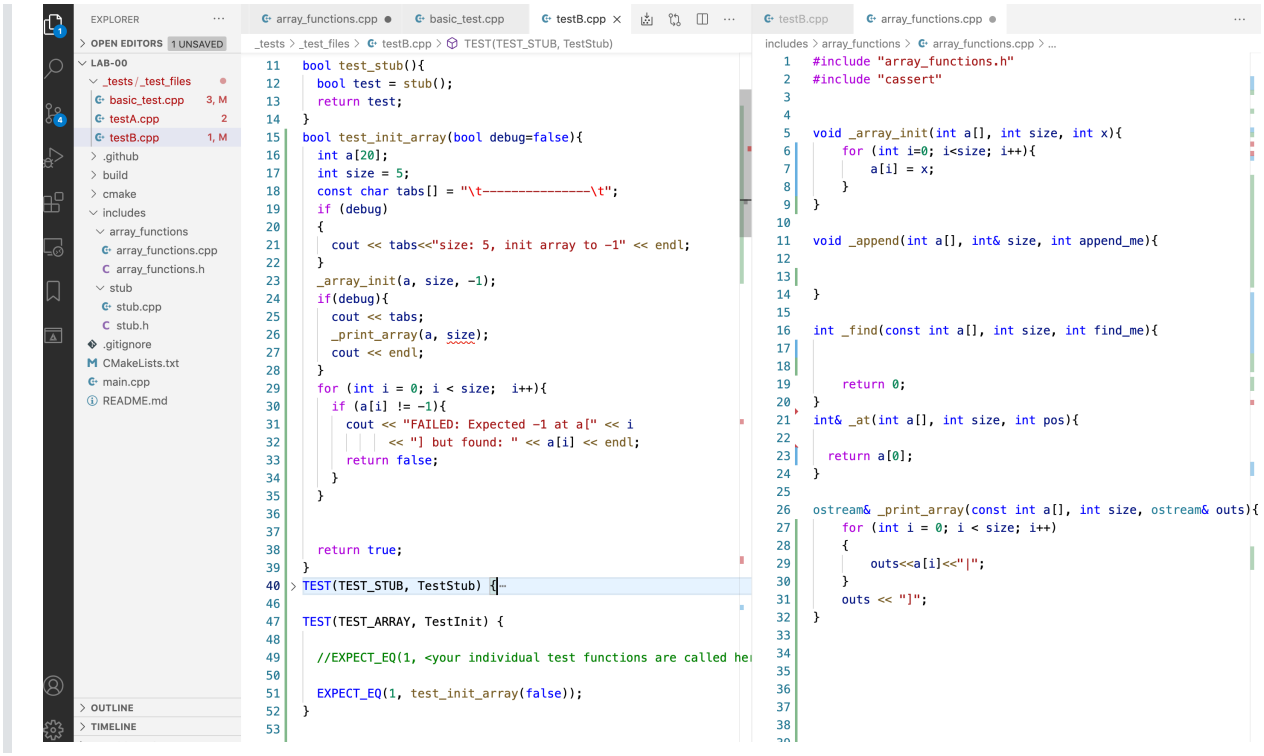
After implementing the _array_init and _print_array functions, we will write a simple test that will verify that the _init function works as it should.

The test function is boolean. It returns `true` if the init function works properly and false otgherwise.

Call the `_array_init` function and then go through each and every cell and verify that each element is -1.

If you find one cell that is not -1, return false.

Note also that we return `true` at the end of the test function. I do this in every test function I write.




#The `TEST` function:

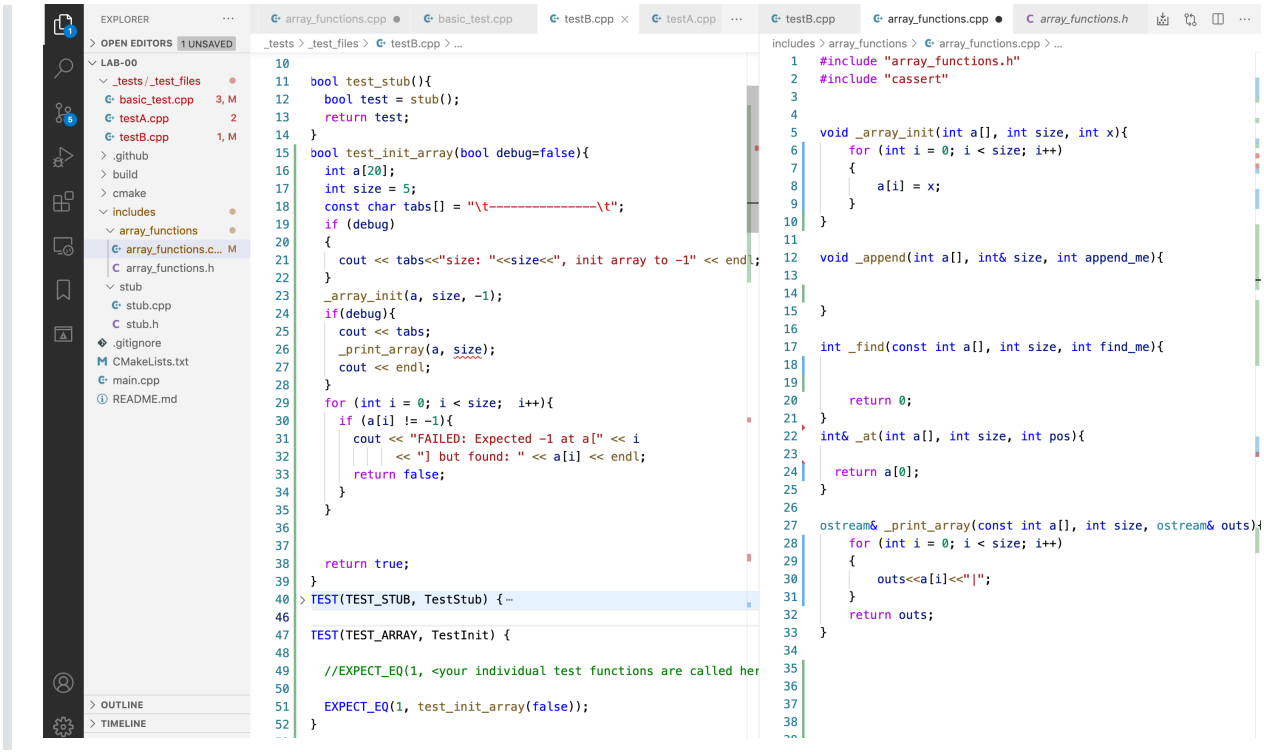
The `TEST` function is part of the googletest testing framework. To simplify our work, we always use the same format for the `TEST` function: Declare a `bool success` and assign it to the return value of the test function.

Then, compare `success` with `1` or `true`

A quick word about the two arguments of the `TEST` function:

The first is the name of the *test suit* and the second is the name of this very test. Each test suit may contain multiple tests. Later, we will write another test for the `_append` function with the same first argument as this test: `TEST_ARRAY` . By the time we are done, the `TEST_ARRAY` test suite will have three individual tests.

Pay attention to the **naming conventions** for this course: The test suite will be in ALL CAPS with underscores between the words. The test names will be camel case and regular function names are all lower case with underscores.




## `make` and RUN!!

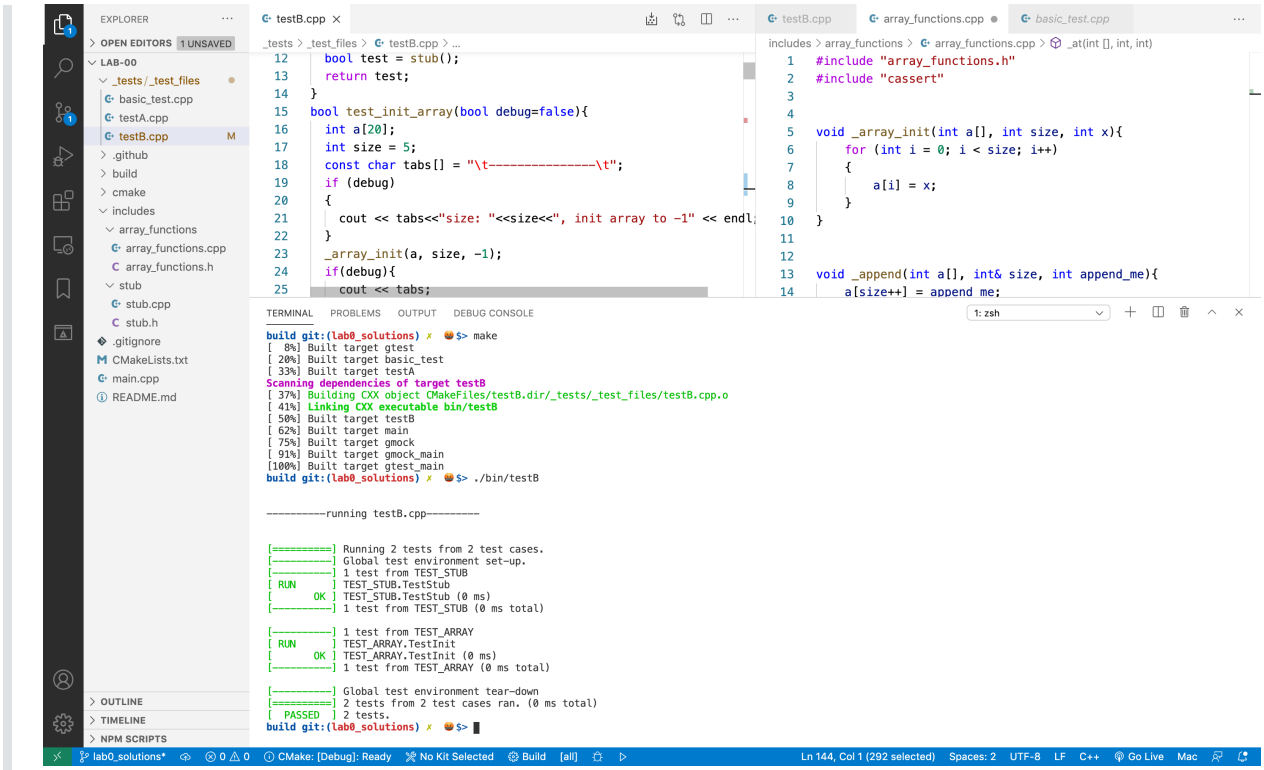
This time, we will run `make` successfully and then, we run the `testB` executable by typing `./bin/testB`

This means execute the file named `testB` that is located in the `bin` folder which is under the `current folder`. The bin folder is created by `make`

remember that `.` means current folder and is not optional. You **must** include the dot in the call to execute `testB`

This will display two successful test runs: one for the `stub` test that was already part of the project, and one for the `TestInit` that we just wrote.

This means that our test function returned `true`.

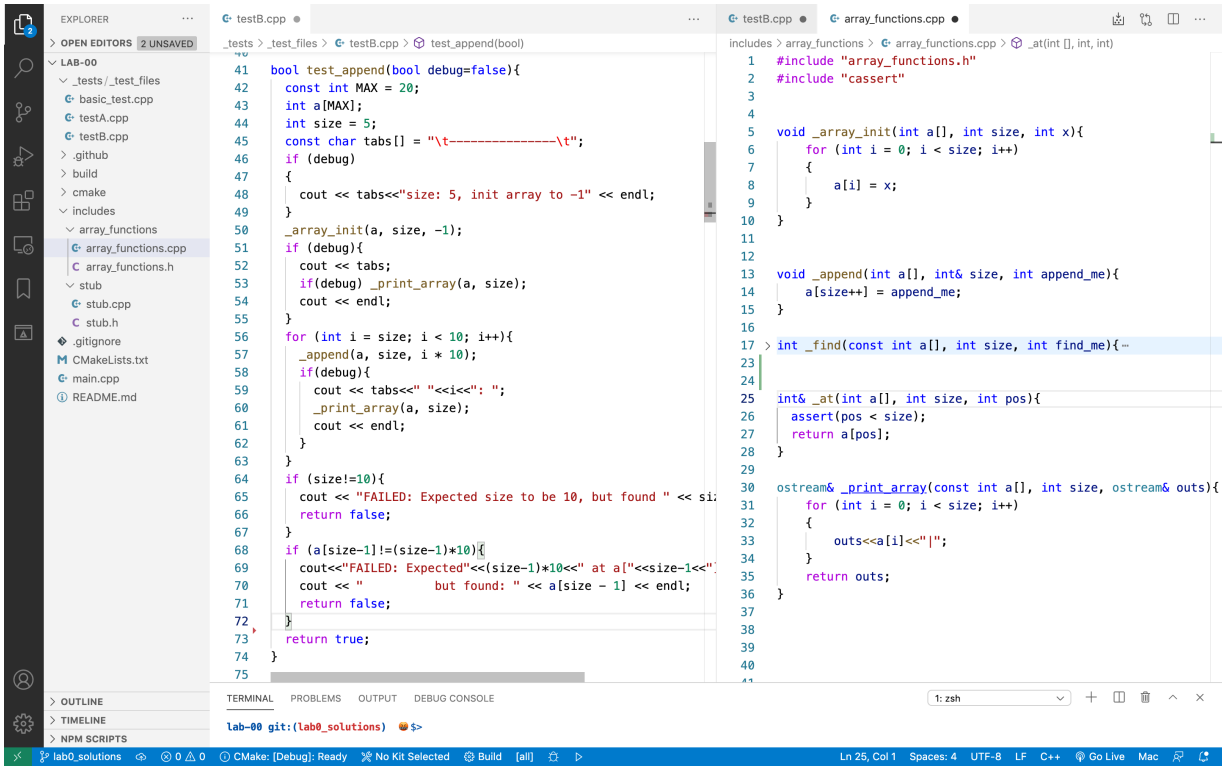



# Implement `_append` and `_at` :

We have thus far implemented `_array_init` and `print_array` . Let's implement `_append` and `_at` as well.

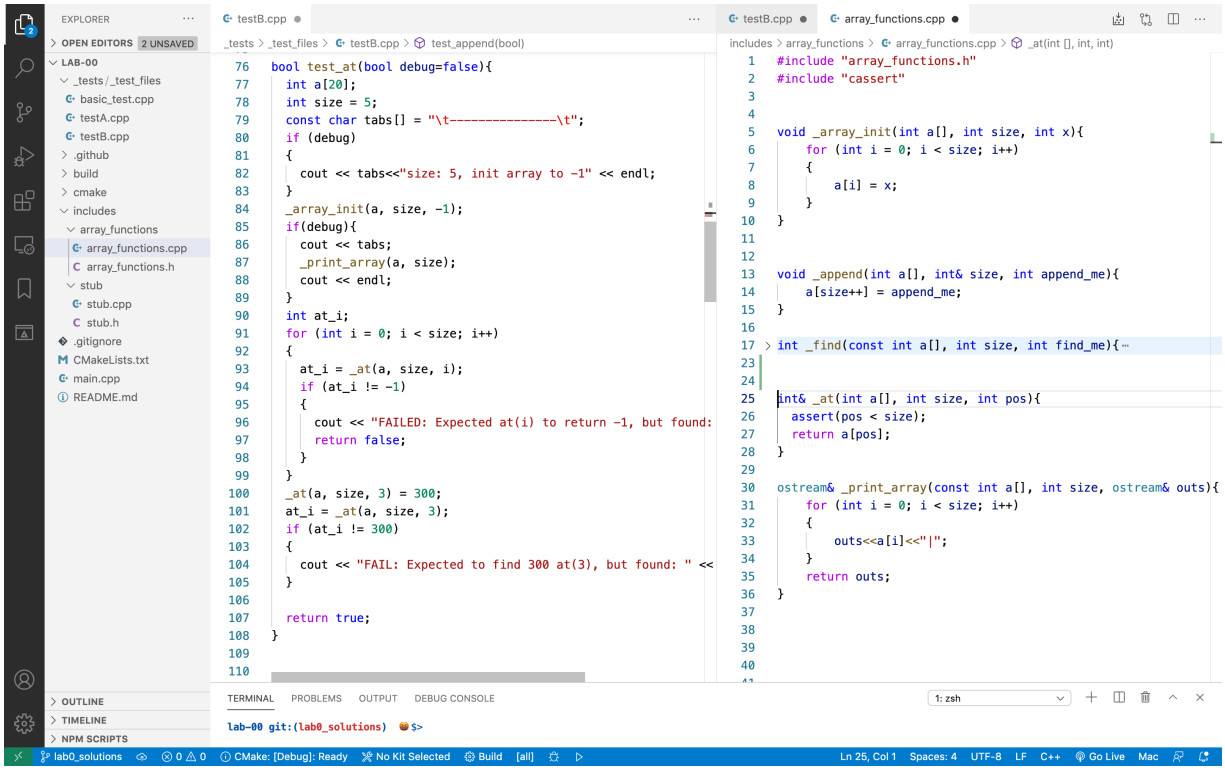
You will *borrow* my code for this particular lab, but make sure you comment the code very well.

Once we have implemented `_append` , write the test for it in `testB.cpp` . Don't forget to add a `TEST( )` for the `test_append()` function.



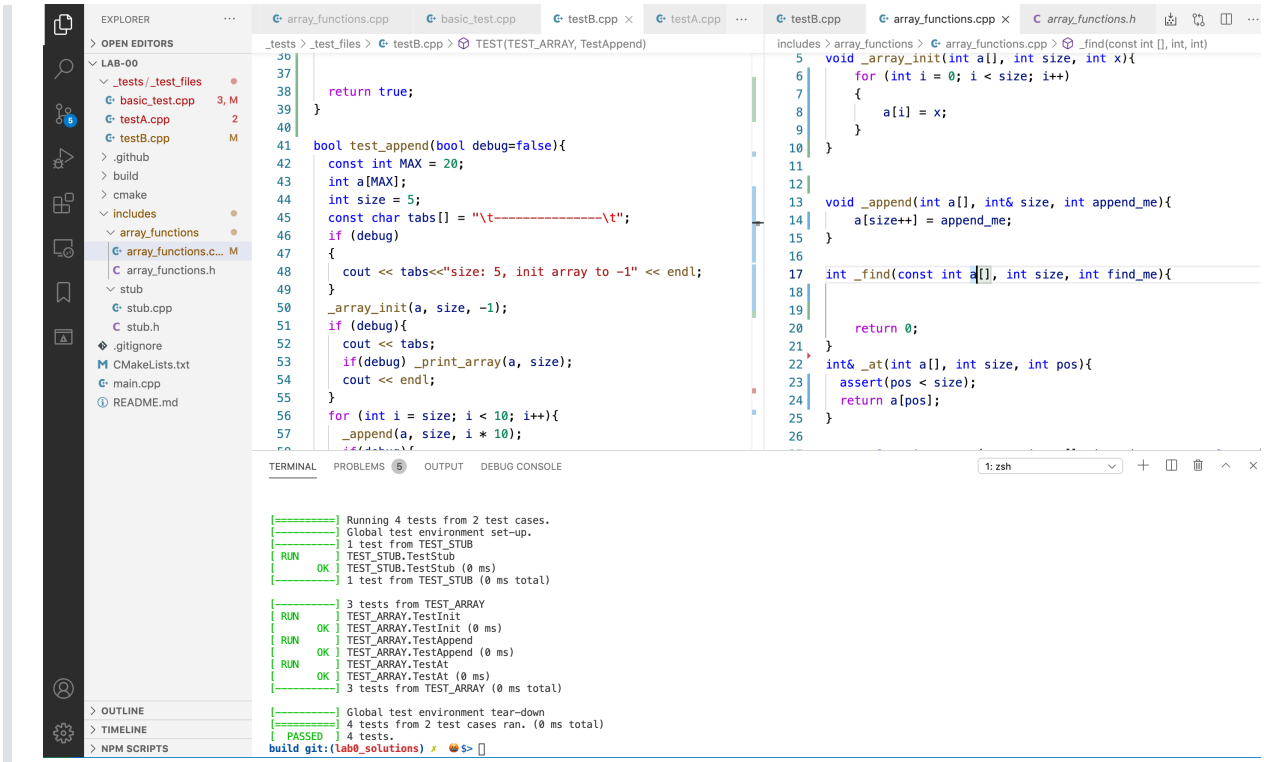

Obviously, this is done in the `testB.cpp` file. Again, do not forget to add the `TEST( )` function for `test_at()`

Once again, you will *borrow* my code for this particular lab, but make sure you comment the code very well.



## `make` and run `testB.cpp` again:

Let's `make` and run `testB` to make sure our `test_append` and `test_at` pass:




# Completing the project

## Implement the `_find()` function on your own

You will also write a `test_find()` function. Once you have implemented `_find()` and written the test function for it (don't forget to comment) you are ready to `make` and run `testB` once again.

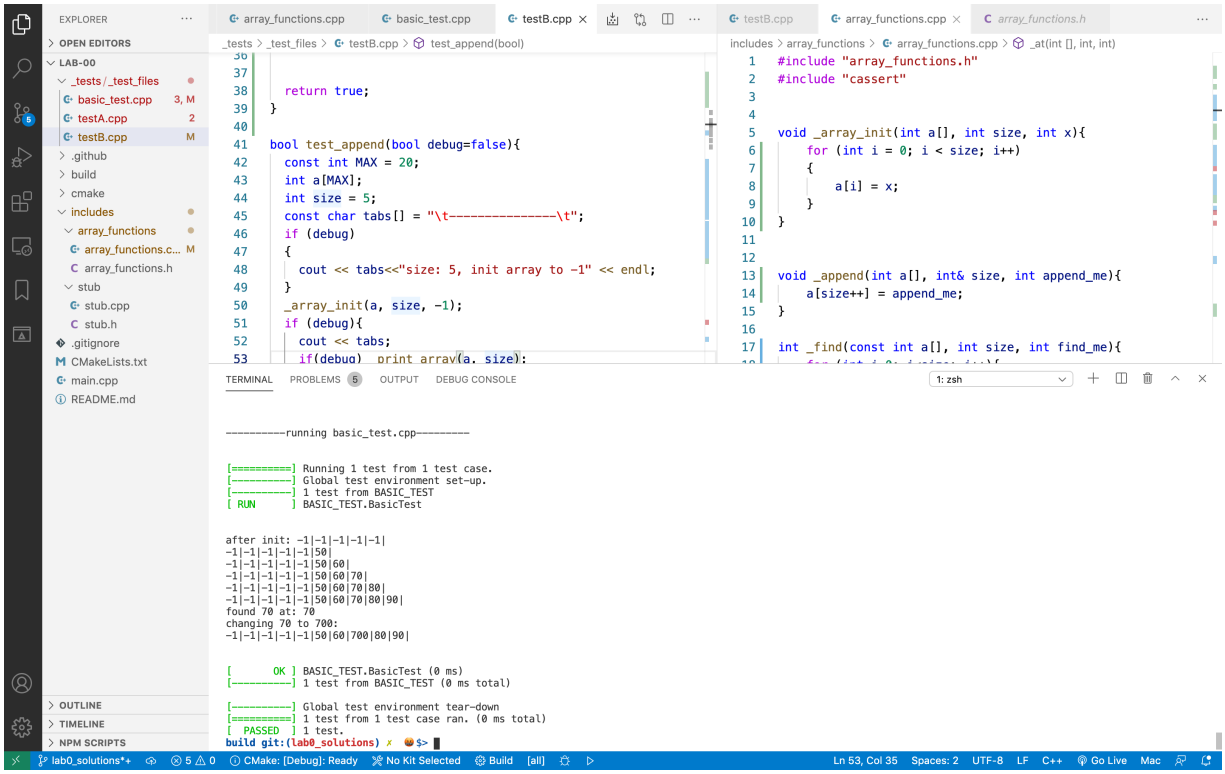
Once you have successfully run `testB` with `_find`, you `git add` and `commit` your changes:

```
build git:(lab0_solutions) ✗  🤬$> git status
On branch lab0_solutions
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   ../_tests/_test_files/basic_test.cpp
        modified:   ../_tests/_test_files/testB.cpp

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   ../_tests/_test_files/basic_test.cpp
        modified:   ../_tests/_test_files/testB.cpp
        modified:   ../includes/array_functions/array_functions.cpp

build git:(lab0_solutions) ✗  🤬$> git add .
build git:(lab0_solutions) ✗  🤬$> git commit -m "PASSED: testB passed all tests."
[lab0_solutions 11c5bfd] PASSED: testB passed all tests.
 2 files changed, 73 insertions(+), 130 deletions(-)
 rewrite _tests/_test_files/basic_test.cpp (69%)
 copy _tests/_test_files/{basic_test.cpp => testB.cpp} (89%)
```

## Finally, we can run `basic_test.cpp`:

Now that we have implemented all the functions that are used in `basic_test.cpp`, we can `make` and run this file.

I cannot overemphasize how important it is for this test to be able to compile and run **without** your editing it in any way. If your project cannot compile and run `basic_test`, the grader will not be able to run your project.




## `git add` and the final `git commit`

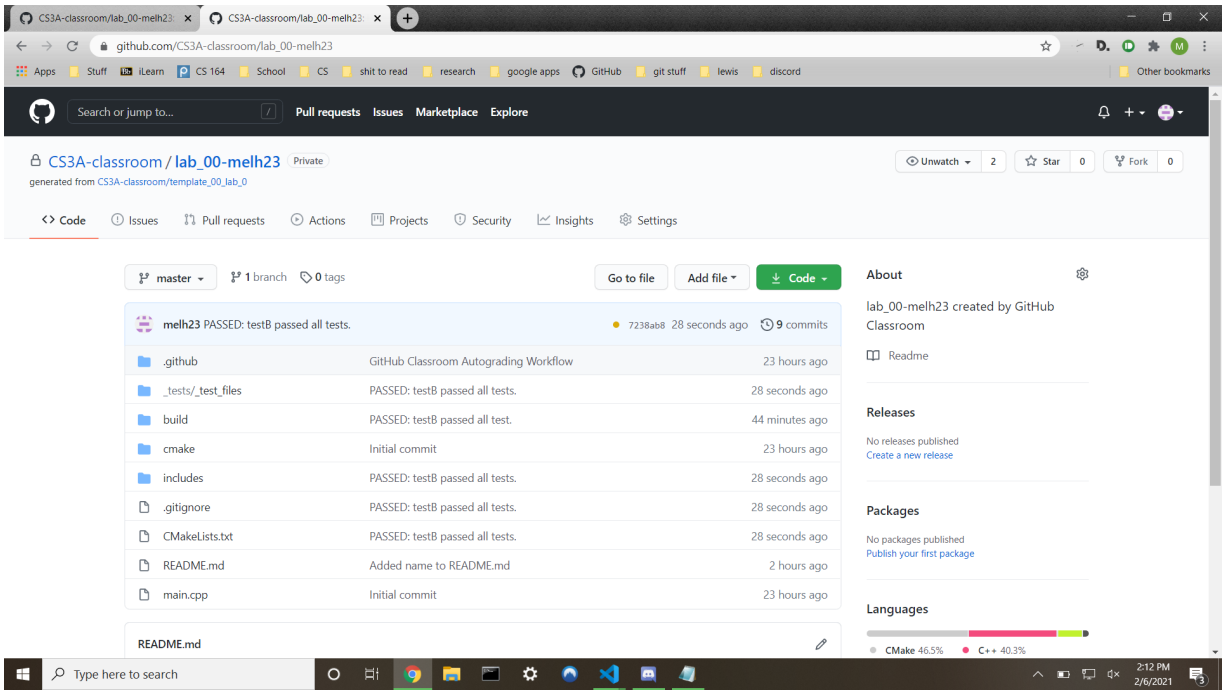
Let's go back to the root directory by typing `cd ..` - remember that `..` means parent directory. `cd ..` means change directory to the parent.

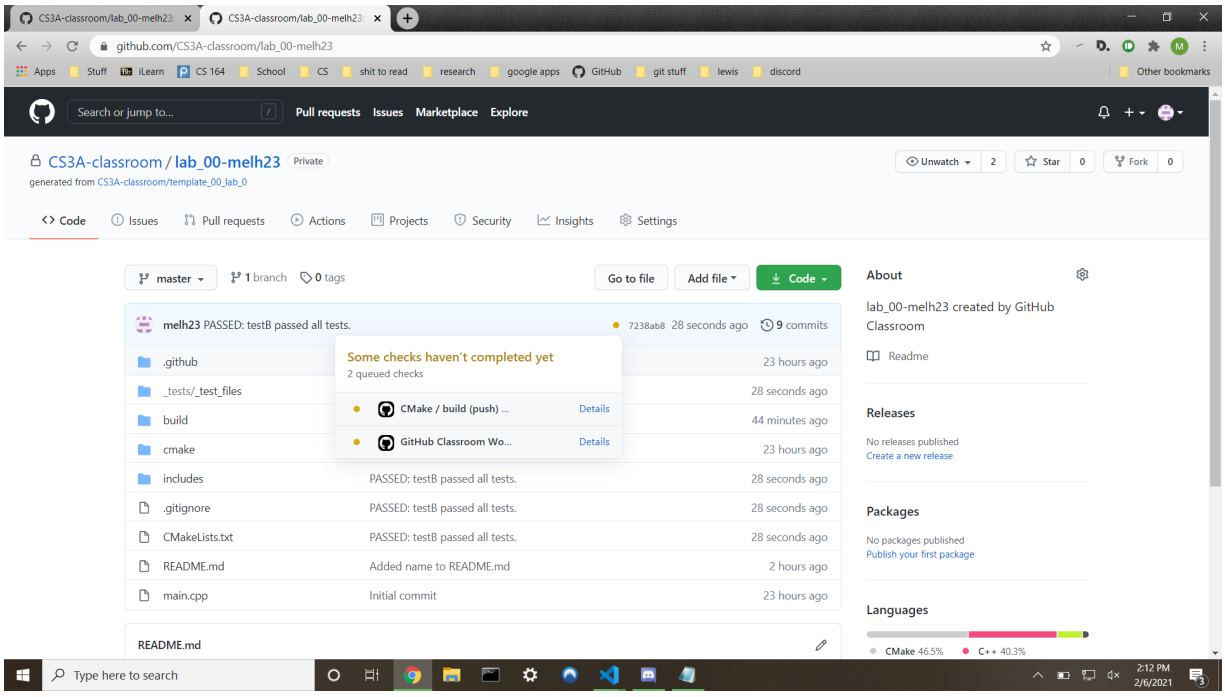
My commit message will let me know what stage of the development I am in. I have just PASSED both the `basic_test` and `testB`

```
build git:(lab0_solutions) ✗  🤬$> git status
On branch lab0_solutions
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   ../_tests/_test_files/basic_test.cpp
        modified:   ../_tests/_test_files/testB.cpp
        modified:   ../includes/array_functions/array_functions.cpp

no changes added to commit (use "git add" and/or "git commit -a")
build git:(lab0_solutions) ✗  🤬$> cd ..
lab-00 git:(lab0_solutions) ✗  🤬$> git add .
lab-00 git:(lab0_solutions) ✗  🤬$> git commit -m "PASSED: testB, basic_test"
[lab0_solutions 765dc96] PASSED: testB, basic_test
 3 files changed, 181 insertions(+), 45 deletions(-)
```

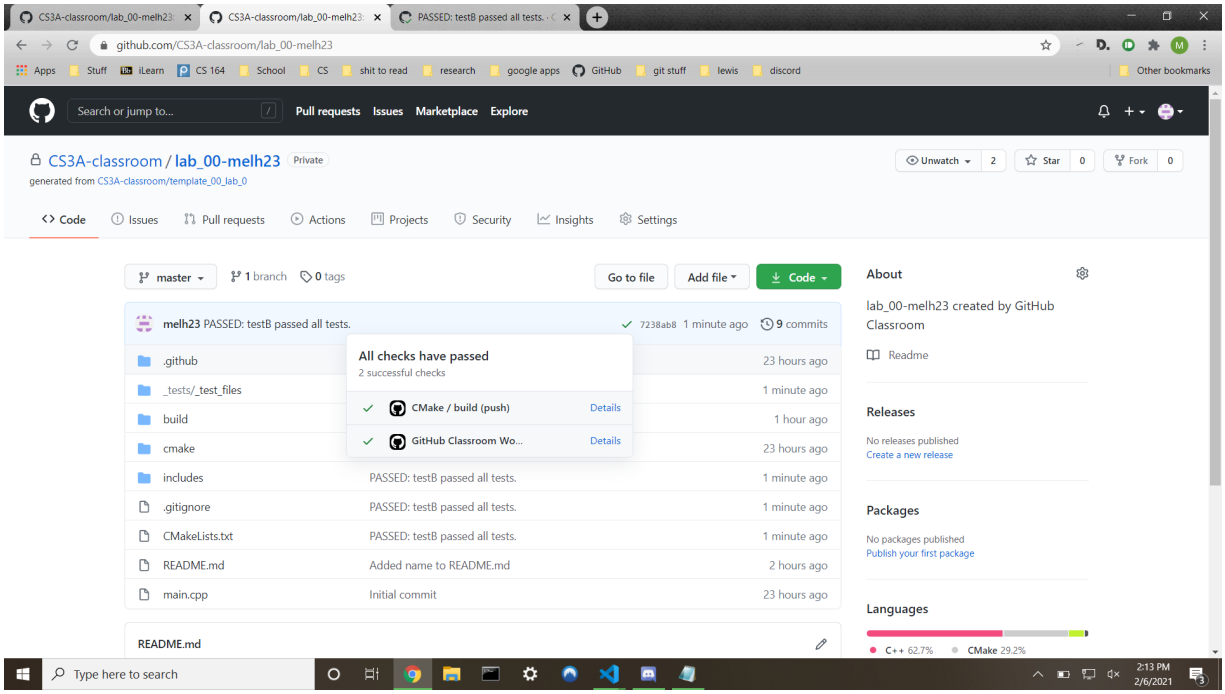
## Autograder Status

You can keep track of the grading status of your project on the assigmment page. A yellow dot means that the tests are still being compiled and run by the autograder.This shouldn't take more than a minute or two. You can refresh the page to update the status. A green checkmark neans that all your tests have passed. A red x means that at least one test failed.

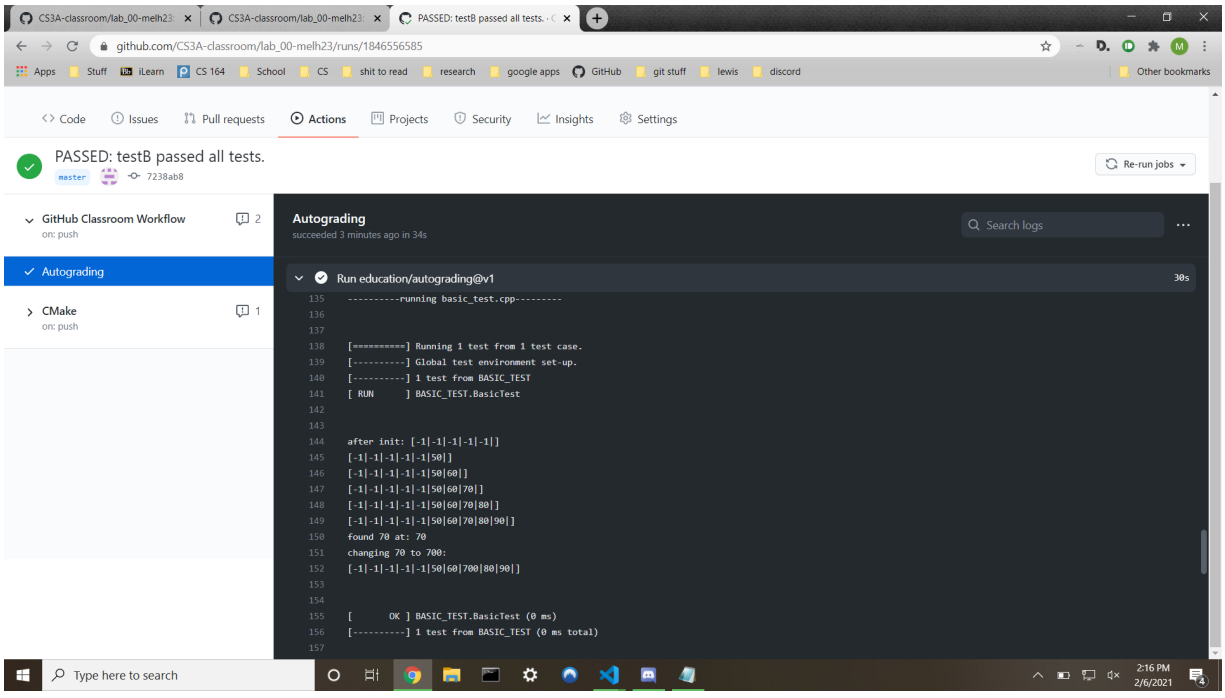


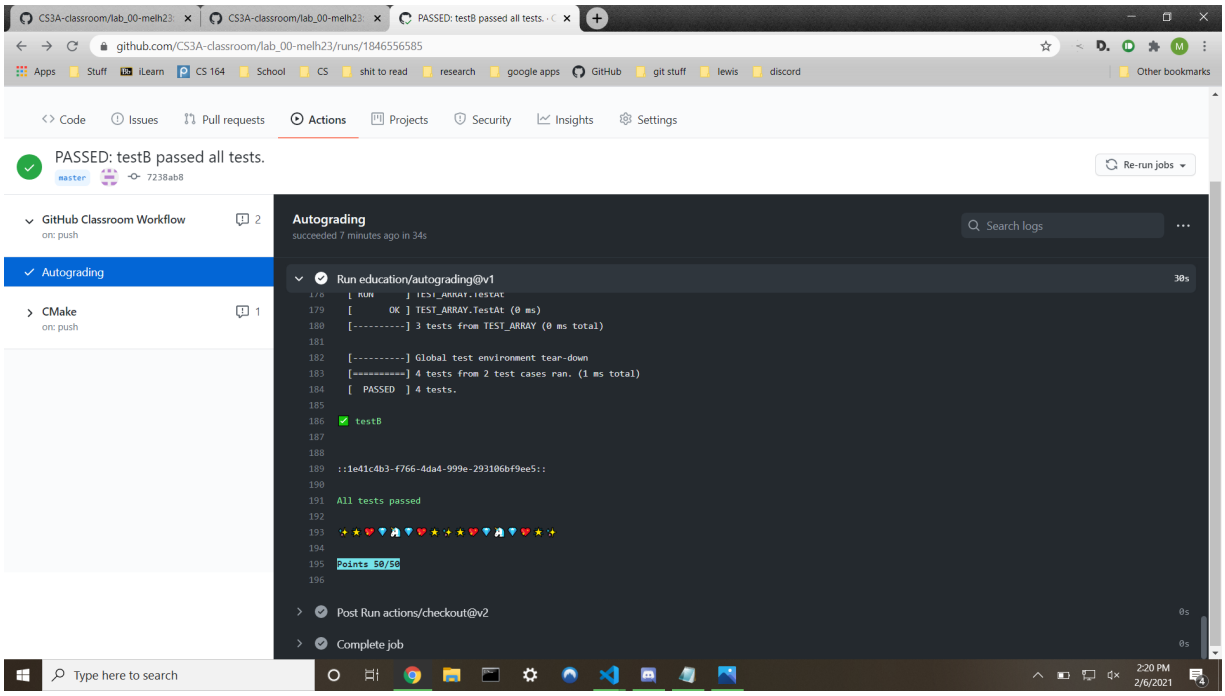



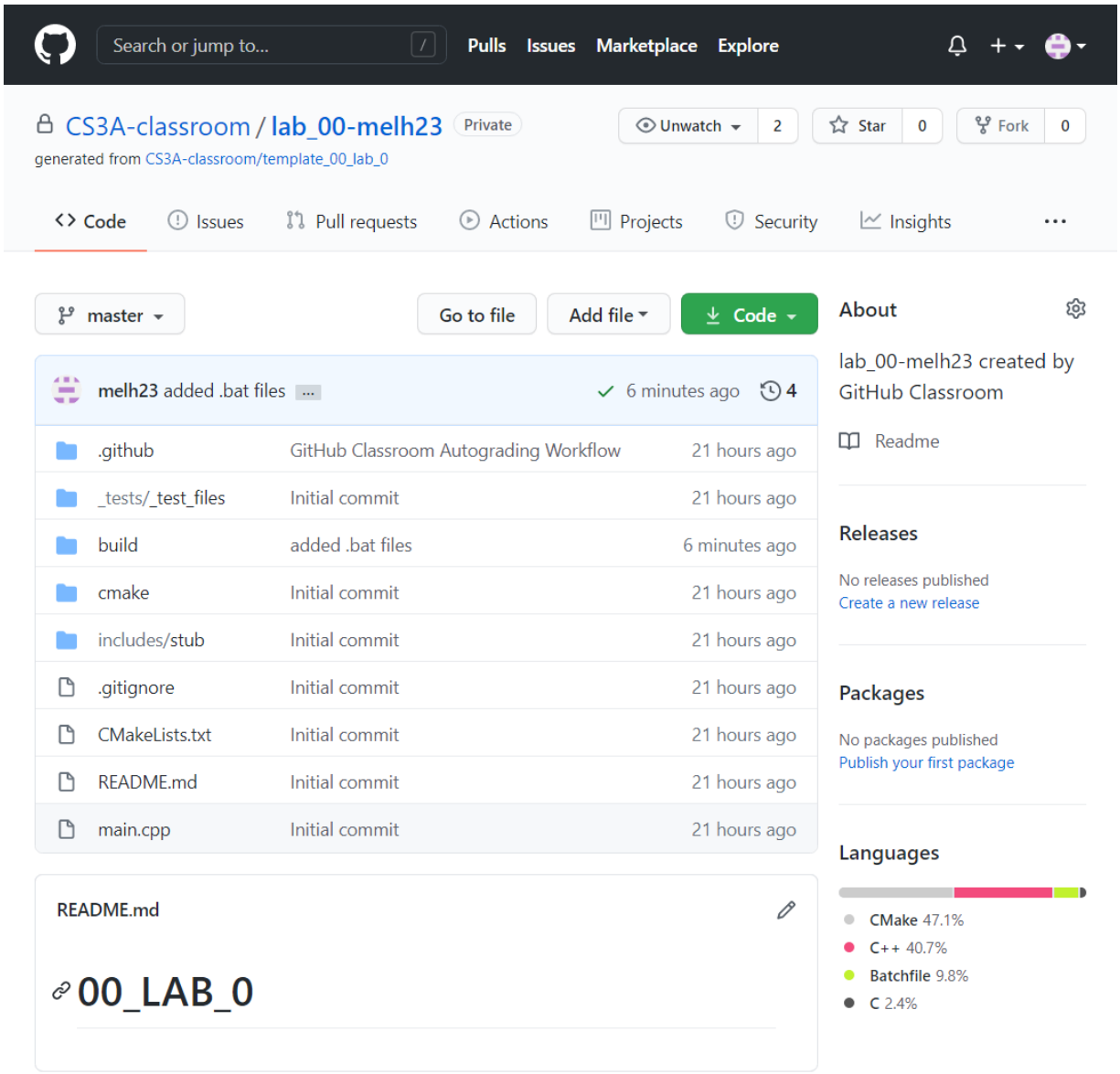
To see a more in depth output of the autograder test runs, click on the Details link:

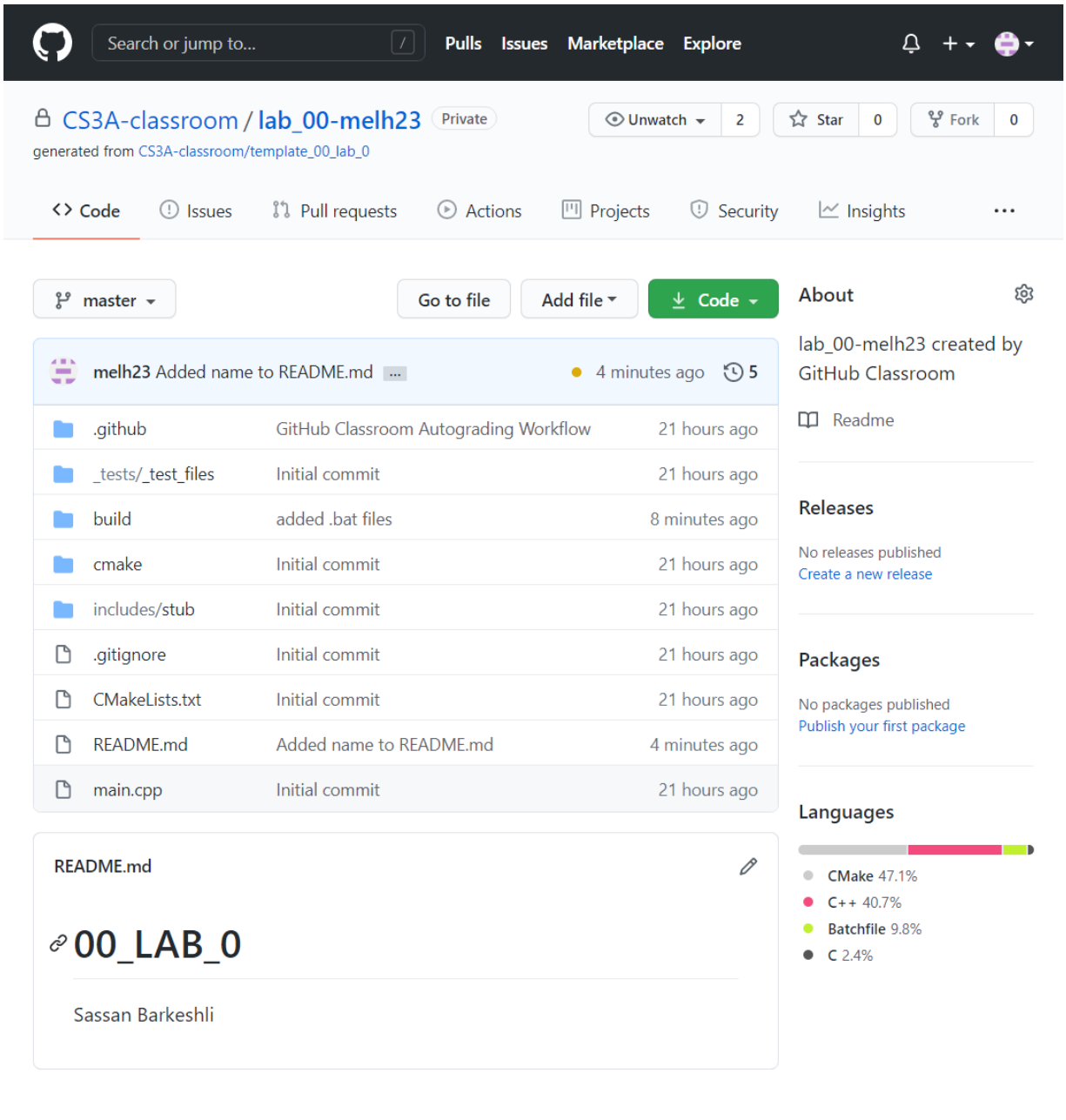


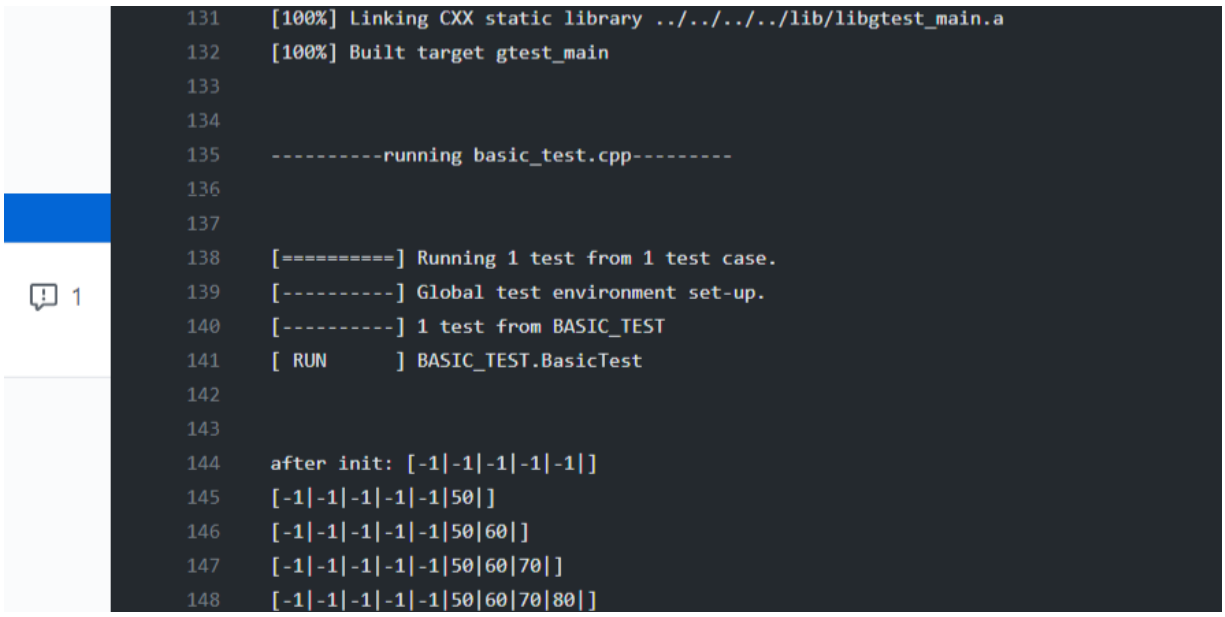

It will take you to this page:



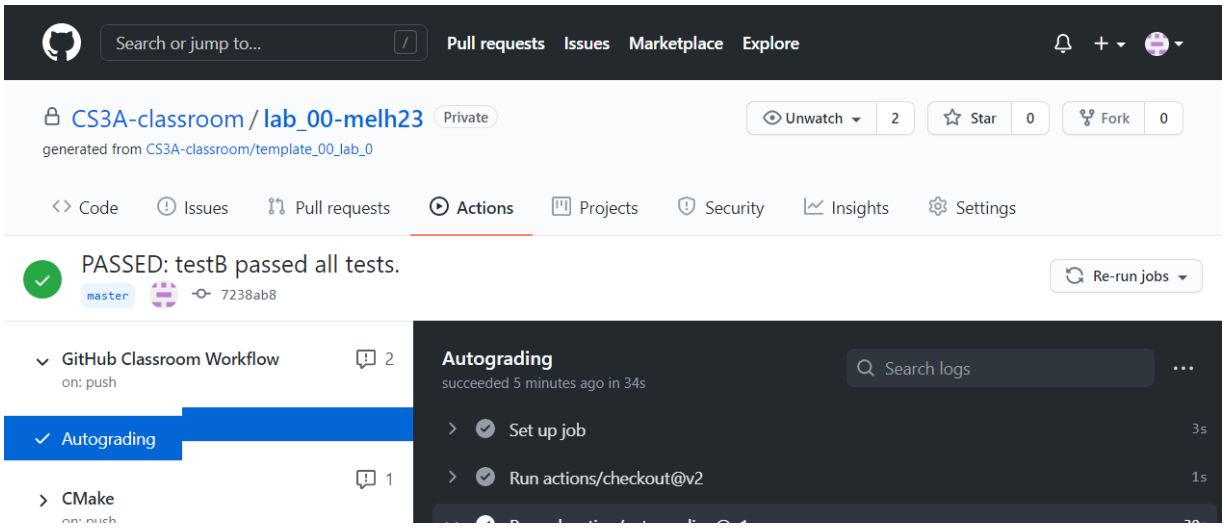

github.com/CS3A-classroom/lab_00-melh23/runs/1846556585
Code
Issues
Pull requests
Actions
Projects
Security
Insights
Settings
PASSED: testB passed all tests.
master
7238ab8
Re-run jobs
GitHub Classroom Workflow
on: push
Autograding
CMake
on: push
Autograding
succeeded 7 minutes ago in 34s
Search logs
Run education/autograding@v1
30s
[ RUN ] TEST_ARRAY.TestAt
[ OK ] TEST_ARRAY.TestAt (0 ms)
[----------] 3 tests from TEST_ARRAY (0 ms total)
[----------] Global test environment tear-down
[==========] 4 tests from 2 test cases ran. (1 ms total)
[ PASSED ] 4 tests.
testB
::1e41c4b3-f766-4da4-999e-293106bf9ee5::
All tests passed
Points 50/50
Post Run actions/checkout@v2
0s
Complete job
0s
Type here to search
2:20 PM
2/6/2021

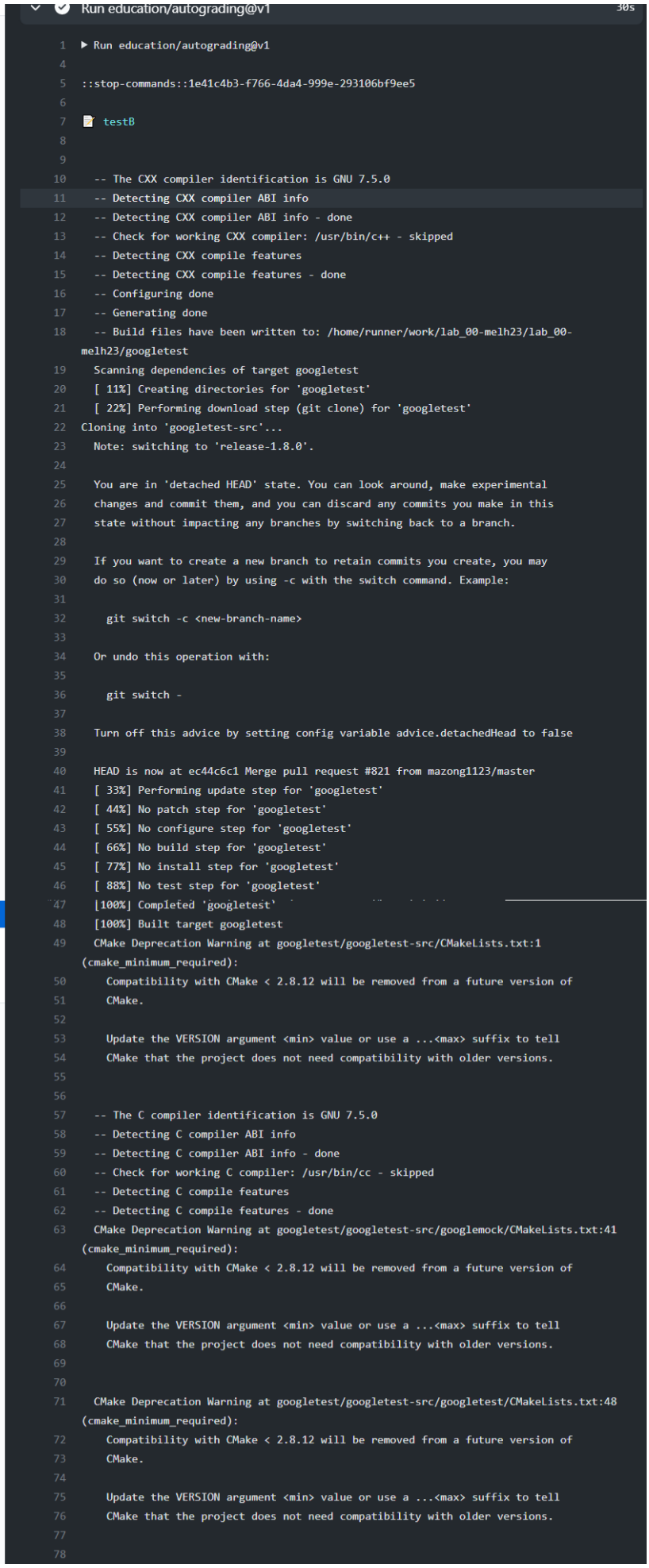

Search or jump to...
Pulls
Issues
Marketplace
Explore
CS3A-classroom / lab_00-melh23
Private
Unwatch 2
Star 0
Fork 0
generated from CS3A-classroom/template_00_lab_0
Code
Issues
Pull requests
Actions
Projects
Security
Insights
master
Go to file
Add file
Code
melh23 added .bat files ...
6 minutes ago
4
.github GitHub Classroom Autograding Workflow 21 hours ago
_tests/_test_files Initial commit 21 hours ago
build added .bat files 6 minutes ago
cmake Initial commit 21 hours ago
includes/stub Initial commit 21 hours ago
.gitignore Initial commit 21 hours ago
CMakeLists.txt Initial commit 21 hours ago
README.md Initial commit 21 hours ago
main.cpp Initial commit 21 hours ago
README.md
00_LAB_0
About
lab_00-melh23 created by GitHub Classroom
Readme
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package
Languages
CMake 47.1%
C++ 40.7%
Batchfile 9.8%
C 2.4%


Search or jump to...
Pulls
Issues
Marketplace
Explore
CS3A-classroom / lab_00-melh23
Private
Unwatch 2
Star 0
Fork 0
generated from CS3A-classroom/template_00_lab_0
Code
Issues
Pull requests
Actions
Projects
Security
Insights
master
Go to file
Add file
Code
melh23 Added name to README.md ... 4 minutes ago 5
.github GitHub Classroom Autograding Workflow 21 hours ago
_tests/_test_files Initial commit 21 hours ago
build added .bat files 8 minutes ago
cmake Initial commit 21 hours ago
includes/stub Initial commit 21 hours ago
.gitignore Initial commit 21 hours ago
CMakeLists.txt Initial commit 21 hours ago
README.md Added name to README.md 4 minutes ago
main.cpp Initial commit 21 hours ago
README.md
00_LAB_0
Sassan Barkeshli
About
lab_00-melh23 created by GitHub Classroom
Readme
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package
Languages
CMake 47.1%
C++ 40.7%
Batchfile 9.8%
C 2.4%


131 [100%] Linking CXX static library ../../../../lib/libgtest_main.a
132 [100%] Built target gtest_main
133
134
135 ----------running basic_test.cpp---------
136
137
138 [==========] Running 1 test from 1 test case.
139 [----------] Global test environment set-up.
140 [----------] 1 test from BASIC_TEST
141 [ RUN ] BASIC_TEST.BasicTest
142
143
144 after init: [-1|-1|-1|-1|-1|]
145 [-1|-1|-1|-1|-1|50|]
146 [-1|-1|-1|-1|-1|50|60|]
147 [-1|-1|-1|-1|-1|50|60|70|]
148 [-1|-1|-1|-1|-1|50|60|70|80|]
1

```
[-1|-1|-1|-1|-1|50|60|70|80|90|]
found 70 at: 70
changing 70 to 700:
[-1|-1|-1|-1|-1|50|60|700|80|90|]


[       OK ] BASIC_TEST.BasicTest (0 ms)
[----------] 1 test from BASIC_TEST (0 ms total)

[----------] Global test environment tear-down
[==========] 1 test from 1 test case ran. (0 ms total)
[  PASSED  ] 1 test.


----------running testB.cpp---------


[==========] Running 4 tests from 2 test cases.
[----------] Global test environment set-up.
[----------] 1 test from TEST_STUB
[ RUN      ] TEST_STUB.TestStub
[       OK ] TEST_STUB.TestStub (0 ms)
[----------] 1 test from TEST_STUB (0 ms total)

[----------] 3 tests from TEST_ARRAY
[ RUN      ] TEST_ARRAY.TestInit
[       OK ] TEST_ARRAY.TestInit (0 ms)
[ RUN      ] TEST_ARRAY.TestAppend
[       OK ] TEST_ARRAY.TestAppend (0 ms)
[ RUN      ] TEST_ARRAY.TestAt
[       OK ] TEST_ARRAY.TestAt (0 ms)
[----------] 3 tests from TEST_ARRAY (0 ms total)

[----------] Global test environment tear-down
[==========] 4 tests from 2 test cases ran. (1 ms total)
[  PASSED  ] 4 tests.

✅ testB

```




Run education/autograding@v1
30s
▶ Run education/autograding@v1
::stop-commands::1e41c4b3-f766-4da4-999e-293106bf9ee5
testB
-- The CXX compiler identification is GNU 7.5.0
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++ - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /home/runner/work/lab_00-melh23/lab_00-melh23/googletest
Scanning dependencies of target googletest
[ 11%] Creating directories for 'googletest'
[ 22%] Performing download step (git clone) for 'googletest'
Cloning into 'googletest-src'...
Note: switching to 'release-1.8.0'.
You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.
If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:
git switch -c <new-branch-name>
Or undo this operation with:
git switch -
Turn off this advice by setting config variable advice.detachedHead to false
HEAD is now at ec44c6c1 Merge pull request #821 from mazong1123/master
[ 33%] Performing update step for 'googletest'
[ 44%] No patch step for 'googletest'
[ 55%] No configure step for 'googletest'
[ 66%] No build step for 'googletest'
[ 77%] No install step for 'googletest'
[ 88%] No test step for 'googletest'
[100%] Completed 'googletest'
[100%] Built target googletest
CMake Deprecation Warning at googletest/googletest-src/CMakeLists.txt:1 (cmake_minimum_required):
Compatibility with CMake < 2.8.12 will be removed from a future version of
CMake.
Update the VERSION argument <min> value or use a ...<max> suffix to tell
CMake that the project does not need compatibility with older versions.
-- The C compiler identification is GNU 7.5.0
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: /usr/bin/cc - skipped
-- Detecting C compile features
-- Detecting C compile features - done
CMake Deprecation Warning at googletest/googletest-src/googlemock/CMakeLists.txt:41 (cmake_minimum_required):
Compatibility with CMake < 2.8.12 will be removed from a future version of
CMake.
Update the VERSION argument <min> value or use a ...<max> suffix to tell
CMake that the project does not need compatibility with older versions.
CMake Deprecation Warning at googletest/googletest-src/googletest/CMakeLists.txt:48 (cmake_minimum_required):
Compatibility with CMake < 2.8.12 will be removed from a future version of
CMake.
Update the VERSION argument <min> value or use a ...<max> suffix to tell
CMake that the project does not need compatibility with older versions.

```
-- Found PythonInterp: /usr/bin/python (found version "2.7.17")
-- Looking for pthread.h
-- Looking for pthread.h - found
-- Performing Test CMAKE_HAVE_LIBC_PTHREAD
-- Performing Test CMAKE_HAVE_LIBC_PTHREAD - Failed
-- Looking for pthread_create in pthreads
-- Looking for pthread_create in pthreads - not found
-- Looking for pthread_create in pthread
-- Looking for pthread_create in pthread - found
-- Found Threads: TRUE
-- Configuring done
-- Generating done
-- Build files have been written to: /home/runner/work/lab_00-melh23/lab_00-melh23

Scanning dependencies of target main
[  4%] Building CXX object CMakeFiles/main.dir/main.cpp.o
[  8%] Building CXX object CMakeFiles/main.dir/includes/stub/stub.cpp.o
```

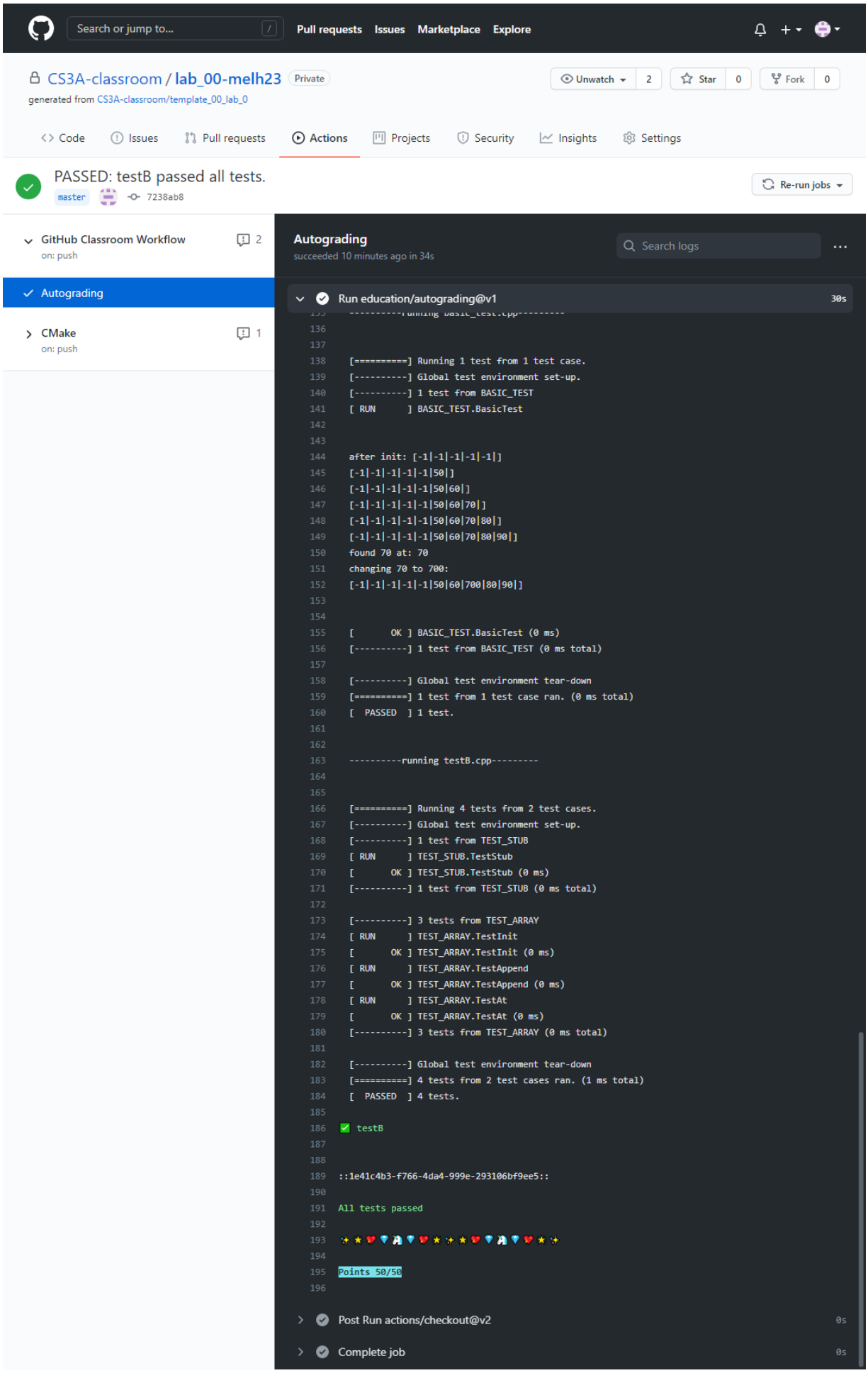
1

```
[ 12%] Linking CXX executable bin/main
[ 12%] Built target main
Scanning dependencies of target gtest
[ 16%] Building CXX object googletest/googletest-build/googlemock/gtest/CMakeFiles/gtest.dir/src/gtest-all.cc.o
[ 20%] Linking CXX static library ../../../../lib/libgtest.a
[ 20%] Built target gtest
Scanning dependencies of target basic_test
[ 25%] Building CXX object CMakeFiles/basic_test.dir/_tests/_test_files/basic_test.cpp.o
[ 29%] Building CXX object CMakeFiles/basic_test.dir/includes/array_functions/array_functions.cpp.o
[ 33%] Linking CXX executable bin/basic_test
[ 33%] Built target basic_test
Scanning dependencies of target testA
[ 37%] Building CXX object CMakeFiles/testA.dir/_tests/_test_files/testA.cpp.o
[ 41%] Building CXX object CMakeFiles/testA.dir/includes/array_functions/array_functions.cpp.o
[ 45%] Linking CXX executable bin/testA
[ 45%] Built target testA
Scanning dependencies of target testB
[ 50%] Building CXX object CMakeFiles/testB.dir/_tests/_test_files/testB.cpp.o
[ 54%] Building CXX object CMakeFiles/testB.dir/includes/array_functions/array_functions.cpp.o
[ 58%] Building CXX object CMakeFiles/testB.dir/includes/stub/stub.cpp.o
[ 62%] Linking CXX executable bin/testB
[ 62%] Built target testB
Scanning dependencies of target gmock_main
[ 66%] Building CXX object googletest/googletest-build/googlemock/CMakeFiles/gmock_main.dir/__/googletest/src/gtest-all.cc.o
[ 70%] Building CXX object googletest/googletest-build/googlemock/CMakeFiles/gmock_main.dir/src/gmock-all.cc.o
[ 75%] Building CXX object googletest/googletest-build/googlemock/CMakeFiles/gmock_main.dir/src/gmock_main.cc.o
[ 79%] Linking CXX static library ../../../lib/libgmock_main.a
[ 79%] Built target gmock_main
Scanning dependencies of target gmock
[ 83%] Building CXX object googletest/googletest-build/googlemock/CMakeFiles/gmock.dir/__/googletest/src/gtest-all.cc.o
[ 87%] Building CXX object googletest/googletest-build/googlemock/CMakeFiles/gmock.dir/src/gmock-all.cc.o
[ 91%] Linking CXX static library ../../../lib/libgmock.a
[ 91%] Built target gmock
Scanning dependencies of target gtest_main
[ 95%] Building CXX object googletest/googletest-build/googlemock/gtest/CMakeFiles/gtest_main.dir/src/gtest_main.cc.o
[100%] Linking CXX static library ../../../../lib/libgtest_main.a
[100%] Built target gtest_main


----------running basic_test.cpp---------
```

1

```
[==========] Running 1 test from 1 test case.
[----------] Global test environment set-up.
[----------] 1 test from BASIC_TEST
[ RUN      ] BASIC_TEST.BasicTest


after init: [-1|-1|-1|-1|-1|]
[-1|-1|-1|-1|-1|50|]
[-1|-1|-1|-1|-1|50|60|]
[-1|-1|-1|-1|-1|50|60|70|]
[-1|-1|-1|-1|-1|50|60|70|80|]
[-1|-1|-1|-1|-1|50|60|70|80|90|]
```

```
found 70 at: 70
changing 70 to 700:
[-1|-1|-1|-1|-1|50|60|700|80|90|]


[       OK ] BASIC_TEST.BasicTest (0 ms)
[----------] 1 test from BASIC_TEST (0 ms total)

[----------] Global test environment tear-down
[==========] 1 test from 1 test case ran. (0 ms total)
[  PASSED  ] 1 test.


----------running testB.cpp---------


[==========] Running 4 tests from 2 test cases.
[----------] Global test environment set-up.
[----------] 1 test from TEST_STUB
[ RUN      ] TEST_STUB.TestStub
[       OK ] TEST_STUB.TestStub (0 ms)
[----------] 1 test from TEST_STUB (0 ms total)

[----------] 3 tests from TEST_ARRAY
[ RUN      ] TEST_ARRAY.TestInit
[       OK ] TEST_ARRAY.TestInit (0 ms)
[ RUN      ] TEST_ARRAY.TestAppend
[       OK ] TEST_ARRAY.TestAppend (0 ms)
[ RUN      ] TEST_ARRAY.TestAt
[       OK ] TEST_ARRAY.TestAt (0 ms)
[----------] 3 tests from TEST_ARRAY (0 ms total)

[----------] Global test environment tear-down
[==========] 4 tests from 2 test cases ran. (1 ms total)
[  PASSED  ] 4 tests.

✅ testB
```


Search or jump to...
Pull requests
Issues
Marketplace
Explore
CS3A-classroom / lab_00-melh23
Private
generated from CS3A-classroom/template_00_lab_0
Unwatch 2
Star 0
Fork 0
Code
Issues
Pull requests
Actions
Projects
Security
Insights
Settings
PASSED: testB passed all tests.
master
7238ab8
Re-run jobs
GitHub Classroom Workflow
on: push
2
Autograding
CMake
on: push
1
Autograding
succeeded 10 minutes ago in 34s
Search logs
Run education/autograding@v1
30s
[==========] Running 1 test from 1 test case.
[----------] Global test environment set-up.
[----------] 1 test from BASIC_TEST
[ RUN ] BASIC_TEST.BasicTest
after init: [-1|-1|-1|-1|-1|]
[-1|-1|-1|-1|-1|50|]
[-1|-1|-1|-1|-1|50|60|]
[-1|-1|-1|-1|-1|50|60|70|]
[-1|-1|-1|-1|-1|50|60|70|80|]
[-1|-1|-1|-1|-1|50|60|70|80|90|]
found 70 at: 70
changing 70 to 700:
[-1|-1|-1|-1|-1|50|60|700|80|90|]
[ OK ] BASIC_TEST.BasicTest (0 ms)
[----------] 1 test from BASIC_TEST (0 ms total)
[----------] Global test environment tear-down
[==========] 1 test from 1 test case ran. (0 ms total)
[ PASSED ] 1 test.
----------running testB.cpp---------
[==========] Running 4 tests from 2 test cases.
[----------] Global test environment set-up.
[----------] 1 test from TEST_STUB
[ RUN ] TEST_STUB.TestStub
[ OK ] TEST_STUB.TestStub (0 ms)
[----------] 1 test from TEST_STUB (0 ms total)
[----------] 3 tests from TEST_ARRAY
[ RUN ] TEST_ARRAY.TestInit
[ OK ] TEST_ARRAY.TestInit (0 ms)
[ RUN ] TEST_ARRAY.TestAppend
[ OK ] TEST_ARRAY.TestAppend (0 ms)
[ RUN ] TEST_ARRAY.TestAt
[ OK ] TEST_ARRAY.TestAt (0 ms)
[----------] 3 tests from TEST_ARRAY (0 ms total)
[----------] Global test environment tear-down
[==========] 4 tests from 2 test cases ran. (1 ms total)
[ PASSED ] 4 tests.
testB
::1e41c4b3-f766-4da4-999e-293106bf9ee5::
All tests passed
Points 50/50
Post Run actions/checkout@v2
0s
Complete job
0s