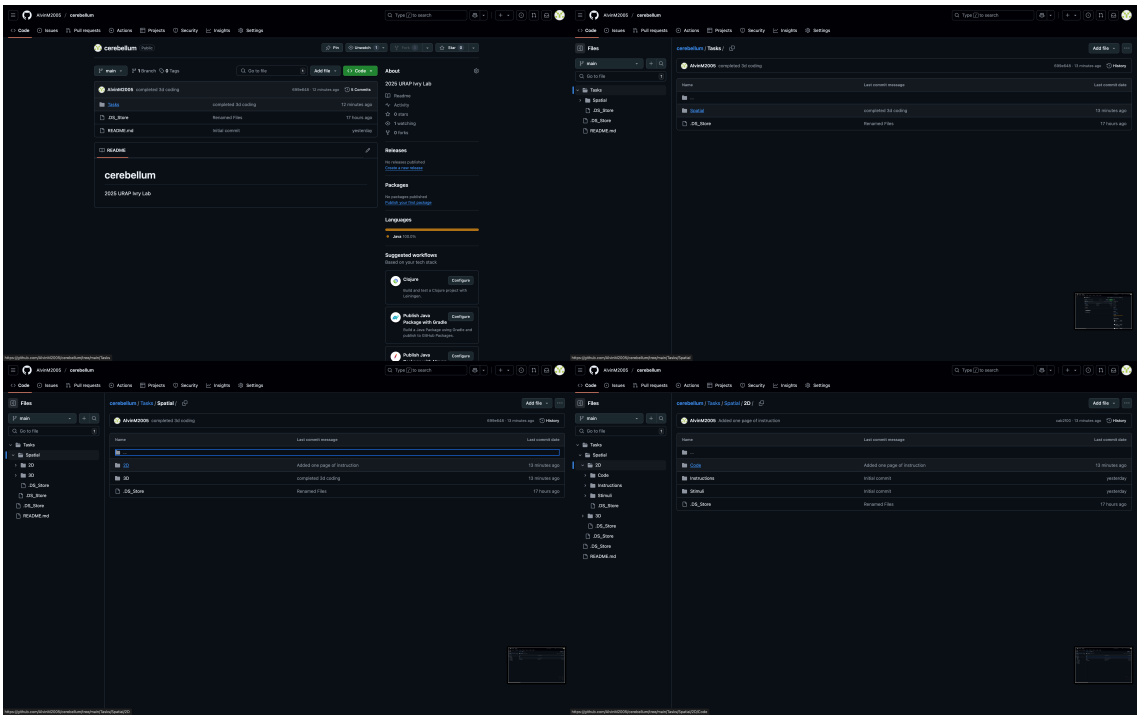
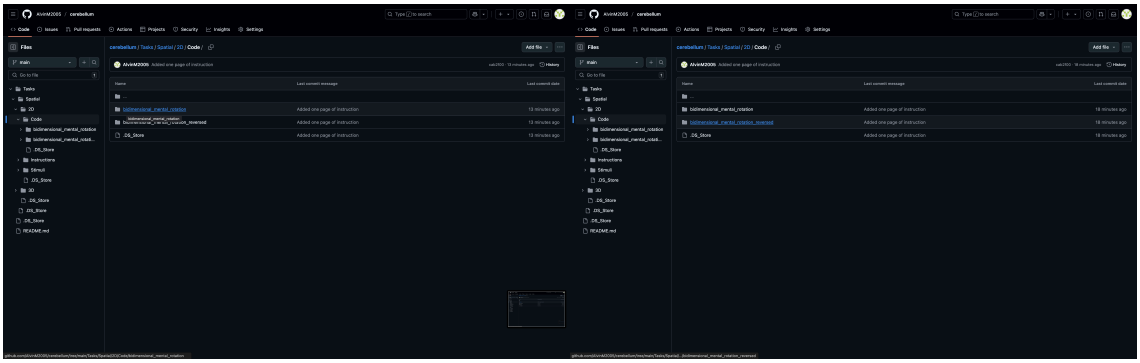
After navigating to the GitHub repository, go to *Tasks-Spatial-2D(or 3D)-Code*.

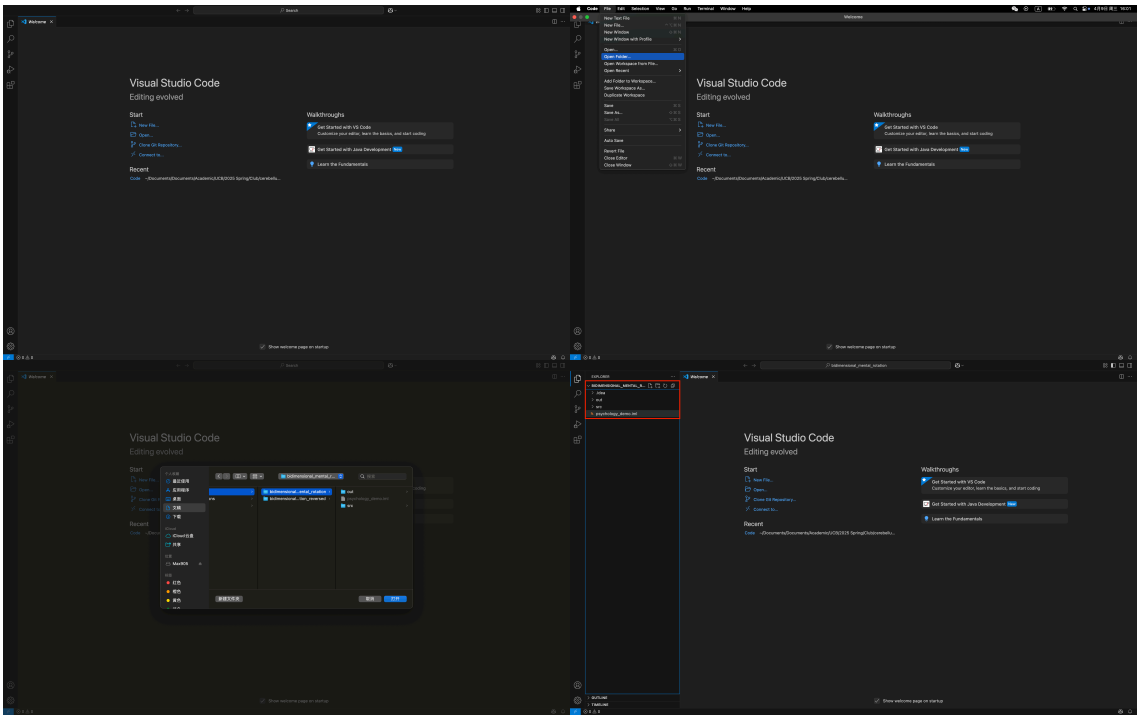


There are two versions of code, normal and reversed, aim to eliminate common-used hands biased. For example, for bidimentional mental rotation tasks, in the normal version, key [V] represents the letter is in the normal state while key [M] represents the letter is in the mirrored state, and the opposite in the reversed version.



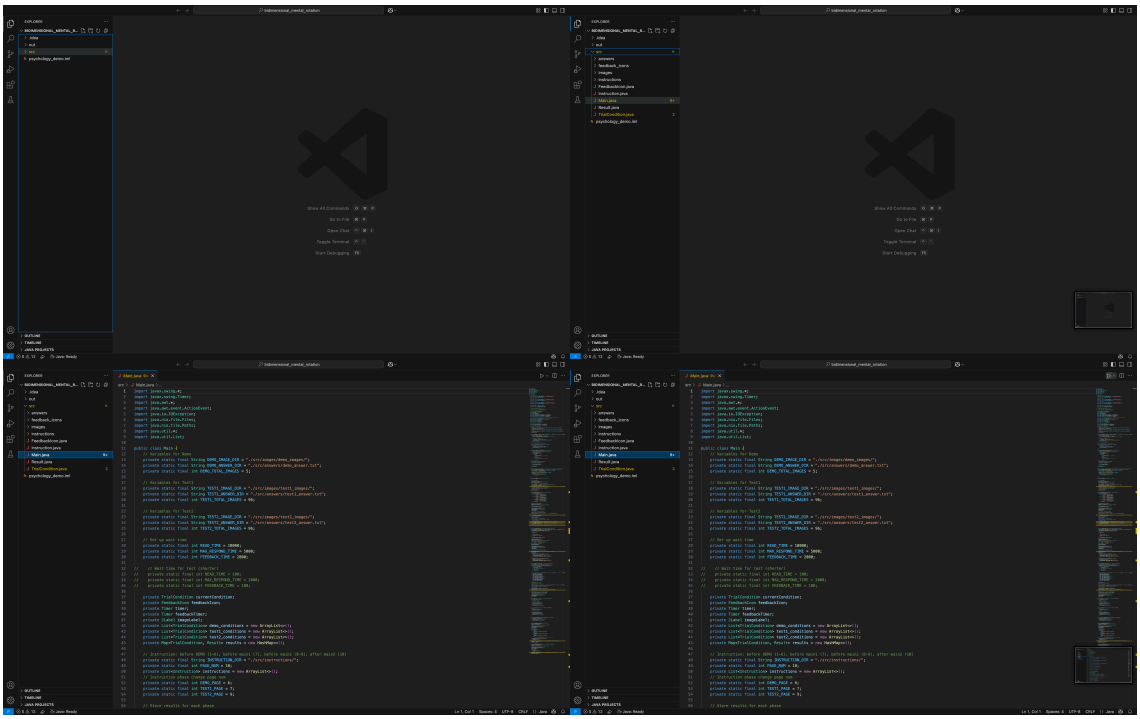
Download the entire folder, and open it with an IDE. I originally coded the tasks using IntelliJ in java. If no java environment had been installed before, I recommend initiate the tasks using VSCode, as it would automatically detect and download all packages necessary, which makes it extremely friendly to first-time users. Therefore, in the rest of this walkthrough, I choose to use VSCode as an example.

Open up a new window of VSCode. Go to *File-Open Folder* (not *Open*!). Navigate to the folder you just downloaded (*bidimensional_mental_rotation*, *bidimensional_mental_rotation_reversed*, *tridimensional_mental_rotation*, *tridimentional_mental_rotation_reversed*). Open up the **entire folder** but not something inside (the program needs to be run in the folder so that the relative file path could work properly). If opened correctly, all the elements in the red square in figure 4 (bottom-right corner) should be visible. Make sure the *src* folder is there. It is the most important component among all.

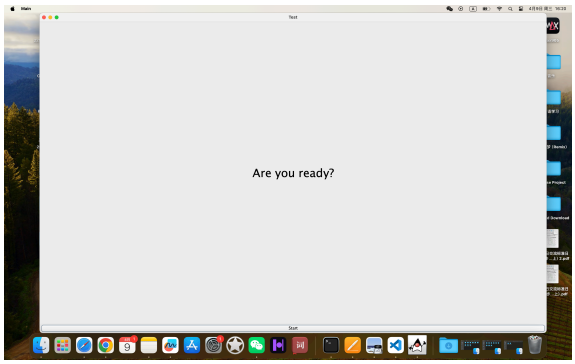


While opening the file, if anything popped up saying “download …”, always choose yes and download the necessary packages. VSCode should be able to identify all necessary packages and give download suggestions to ensure the code could run properly. You might need to restart VSCode and open up the folder again after downloading the packages.

After all packages are properly installed and loaded, click the little triangle on the left of the *src* folder, and choose *Main.java* in the drop down list. Single click to open the java file. After the file is loaded, there should be a triangle icon on the top-right corner of the screen. If you hang your mouse over it, you will see the option “run java”. Single click the triangle icon to initiate the program, and wait for a while for the task to load.

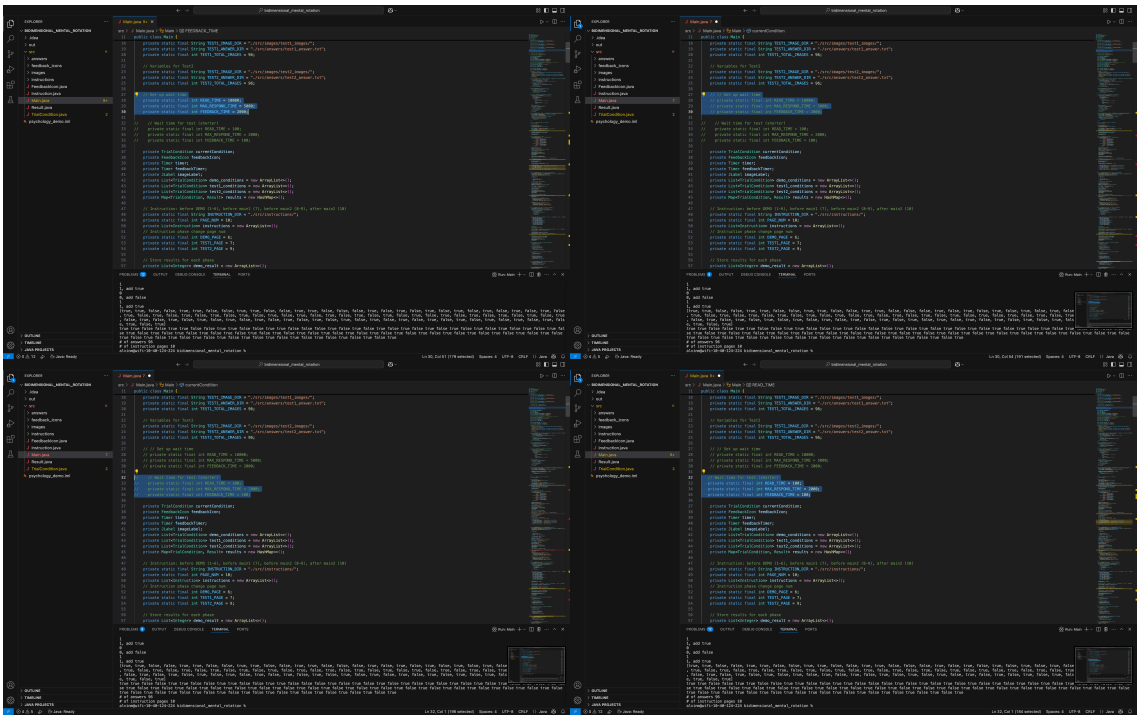


If everything is working properly, you will be able to see the task (shown below)! Either click the button at the bottom of the screen or simply press **[SPACE]** to start.



The code is initially set for the testing purposes, so it goes slowly. I added a button lock so that the participants must read every page of instruction for at least 10 seconds to proceed to the next one. For researchers, if you want to skim through the task quickly and get an overall idea of it, reset the timer inside the Main.java file following the steps shown below.

Highlight line 27-30, and then press **[COMMAND]** + **[/]** on the keyboard. This would comment out these lines. After that, highlight line 32-35, and then press **[COMMAND]** + **[/]** on the keyboard. This would uncomment these lines. If you finish skimming and want to show the task to participants, repeat this step again. This time, line 27-30 will be uncommented while line 32-35 will be commented. Same line numbers apply for all four coding files, so you can follow the exact same steps to modify any of them.



After the participant complete all tasks (demo + test1 + test2), the program will automatically quite. The result (number of correct, incorrect, and time out responses) will be displayed in the bottom pane (marked with the red square in the figure below).