CS 61B

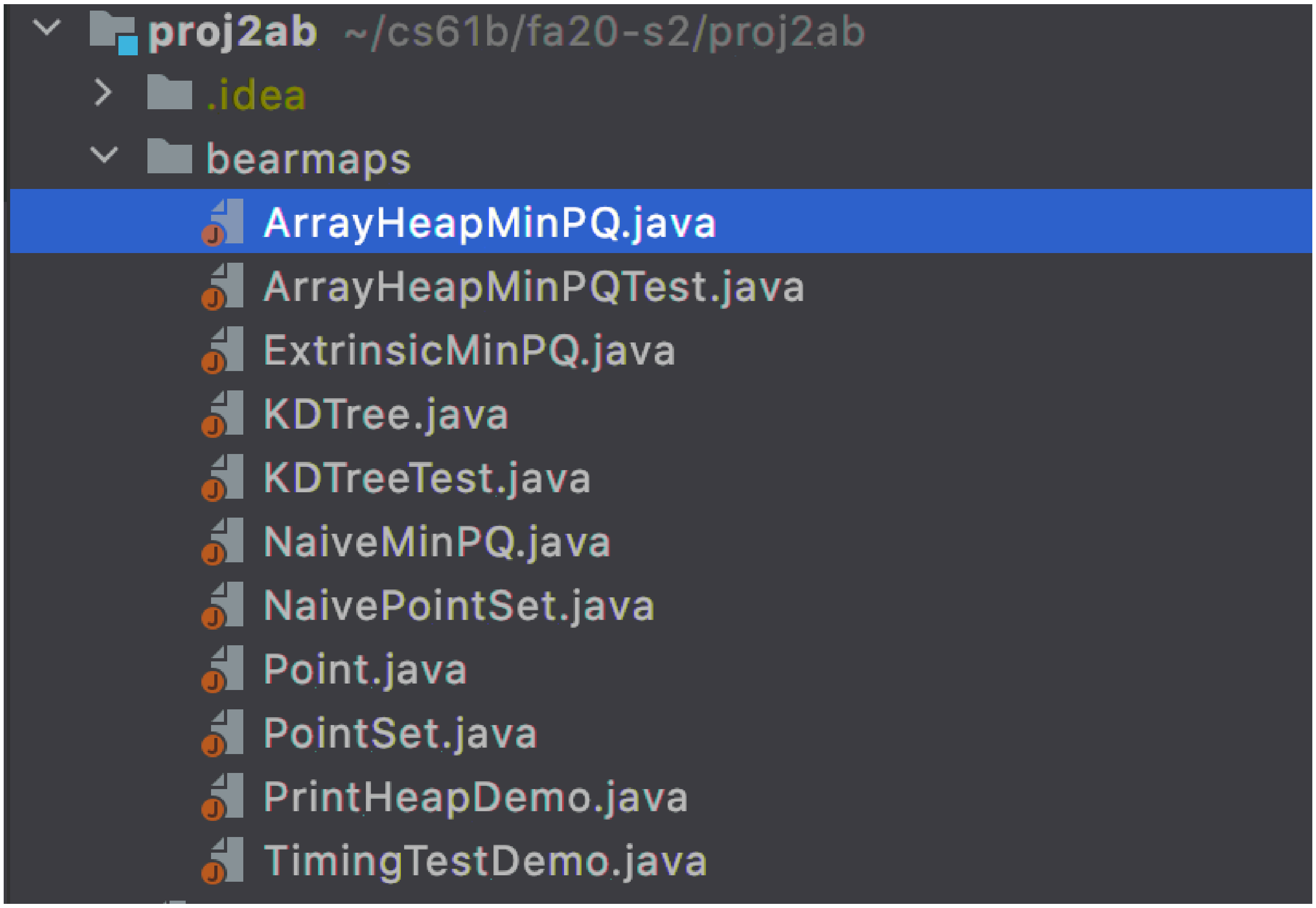
Guides / IntelliJ / IntelliJ WTFS

IntelliJ WTFS

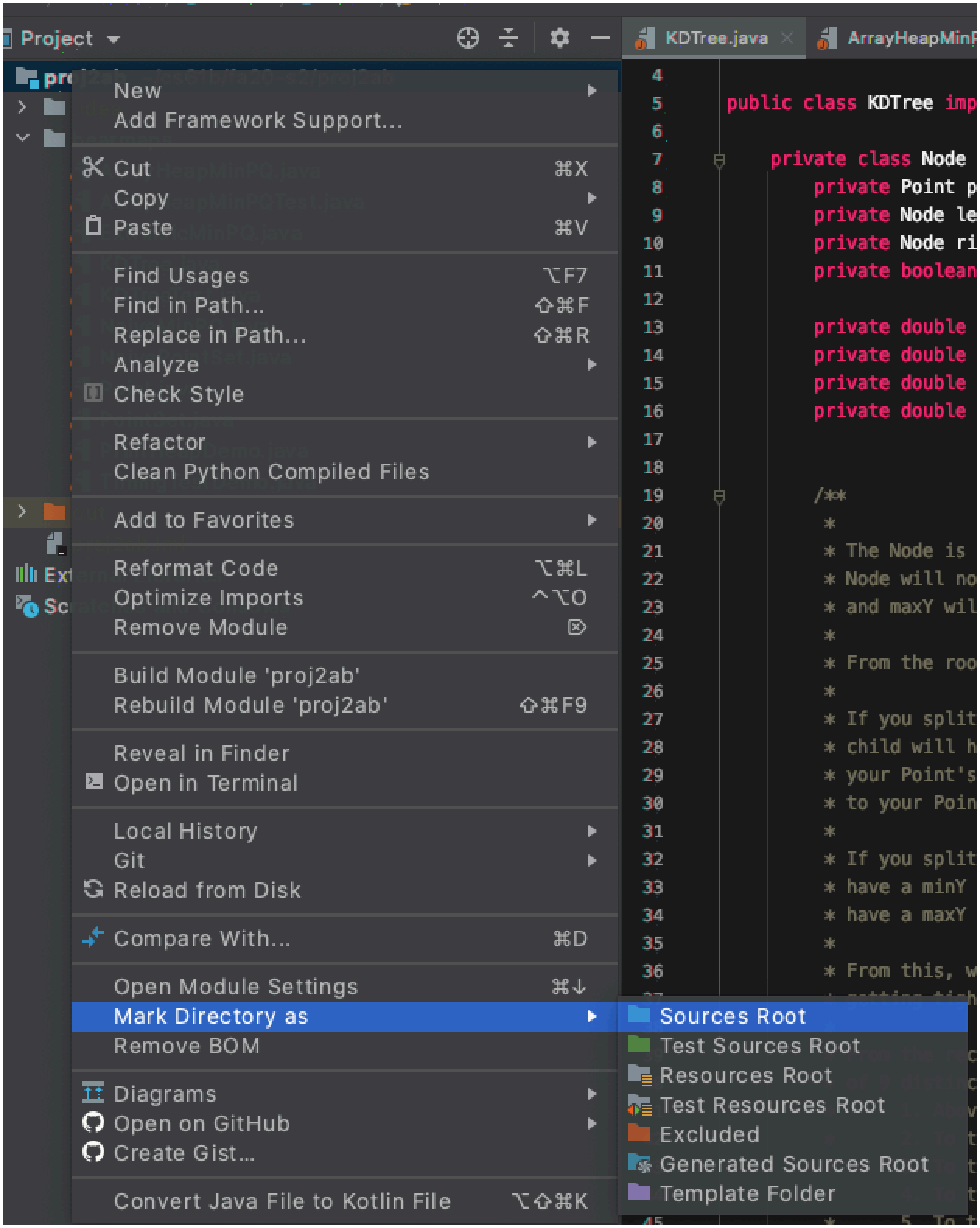
Author: Brandon Lee, Omar Khan

This document is intended to help you through frequently encountered weird technical failure scenarios (WTFS) in IntelliJ. It will be populated as questions arise.

I can't run my Java file/Files don't show up as java files



If your files look like this then you haven't properly imported the project. To fix this, you must simply right-click the outer-most folder which will be your assignment folder, in this case, `proj2ab` , and scroll down to "Mark directory as …" and hit "Sources Root". It'll look like this:

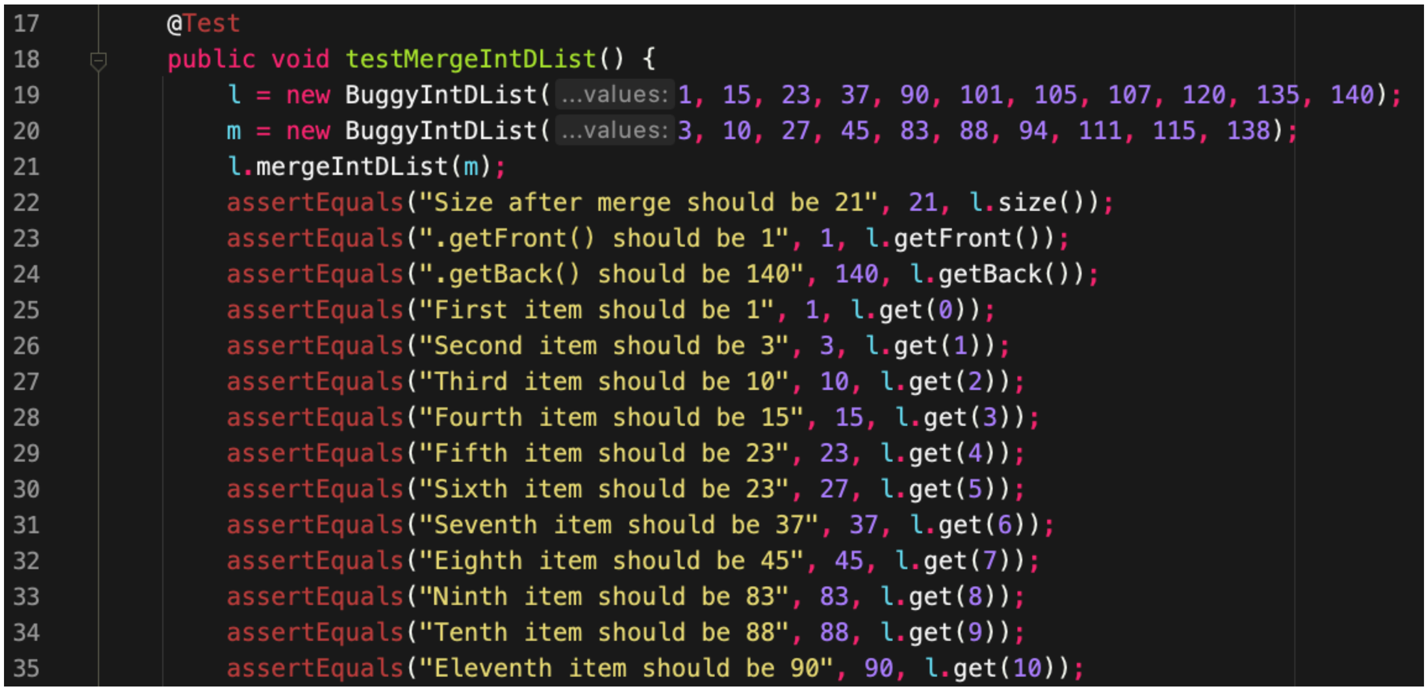


```
public class KDTree imp

    private class Node
        private Point p
        private Node le
        private Node ri
        private boolean

        private double
        private double
        private double
        private double


        /**
         *
         * The Node is
         * Node will no
         * and maxY wil
         *
         * From the roo
         *
         * If you split
         * child will h
         * your Point's
         * to your Poin
         *
         * If you split
         * have a minY
         * have a maxY
         *
         * From this, w
```

You may need to do this with the `src` folder for most assignments, and mark the `tests` folder as a **Test Sources Root**. Your `src` folder should be blue and your `tests` folder should be green.

See Assignment Workflow for additional instructions on importing a new project.

JUnit things show up as red in IntelliJ

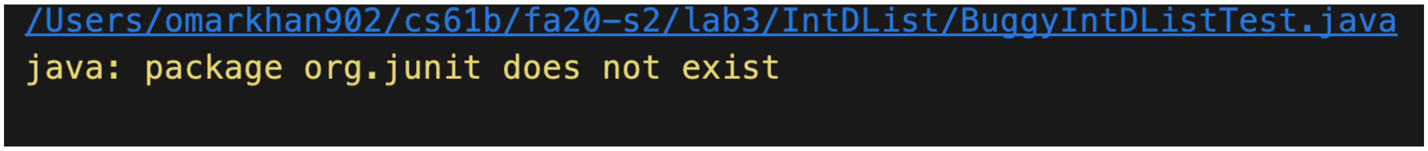


```
    @Test
    public void testMergeIntDList() {
        l = new BuggyIntDList( ...values: 1, 15, 23, 37, 90, 101, 105, 107, 120, 135, 140);
        m = new BuggyIntDList( ...values: 3, 10, 27, 45, 83, 88, 94, 111, 115, 138);
        l.mergeIntDList(m);
        assertEquals("Size after merge should be 21", 21, l.size());
        assertEquals(".getFront() should be 1", 1, l.getFront());
        assertEquals(".getBack() should be 140", 140, l.getBack());
        assertEquals("First item should be 1", 1, l.get(0));
        assertEquals("Second item should be 3", 3, l.get(1));
        assertEquals("Third item should be 10", 10, l.get(2));
        assertEquals("Fourth item should be 15", 15, l.get(3));
        assertEquals("Fifth item should be 23", 23, l.get(4));
        assertEquals("Sixth item should be 23", 27, l.get(5));
        assertEquals("Seventh item should be 37", 37, l.get(6));
        assertEquals("Eighth item should be 45", 45, l.get(7));
        assertEquals("Ninth item should be 83", 83, l.get(8));
        assertEquals("Tenth item should be 88", 88, l.get(9));
        assertEquals("Eleventh item should be 90", 90, l.get(10));
```

This means that you forgot to add the CS 61B `javalib` as a library for this project! IntelliJ cannot find the JUnit specific things like `@Test` or `assertEquals` since you forgot to import them.

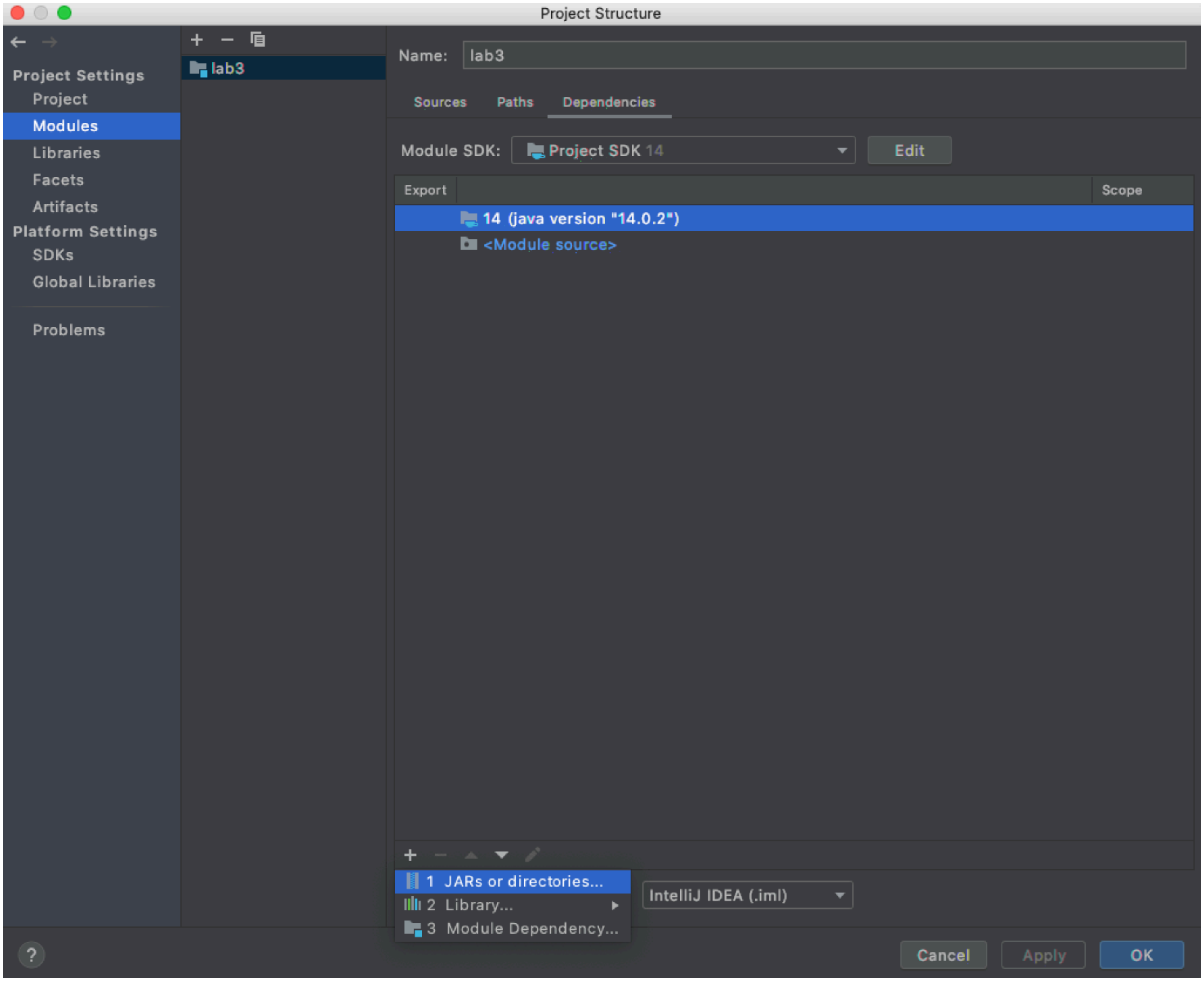
Every time you start a new assignment (unfortunately, EVERY TIME) you must readd the `javalib`. To do this, simply go to "File" -> "Project Structure" -> "Libraries" and add the javalib inside your repo. See Lab 2 Setup for additional instructions on adding Java libraries.

package org.junit does not exist

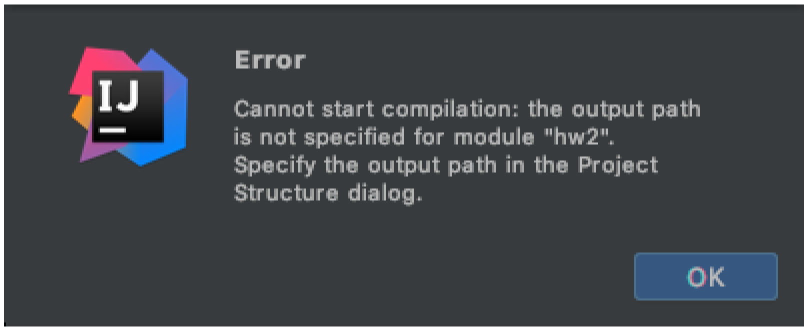


```
/Users/omarkhan902/cs61b/fa20-s2/lab3/IntDList/BuggyIntDListTest.java
java: package org.junit does not exist
```

This sometimes happens with IntelliJ where you've added the correct libraries, but cannot run the code. To fix this, you need to add the libraries as a direct dependency to the module. To do that, go to "File" -> "Project Structure" -> "Modules" -> "Dependencies" and hit the "+" icon in the bottom left and select "JARs or directories". Now, highlight every `.jar` file in your `javalib` folder and add them. This is what the setting should look like right before you add the JARs:



Output path is not specified

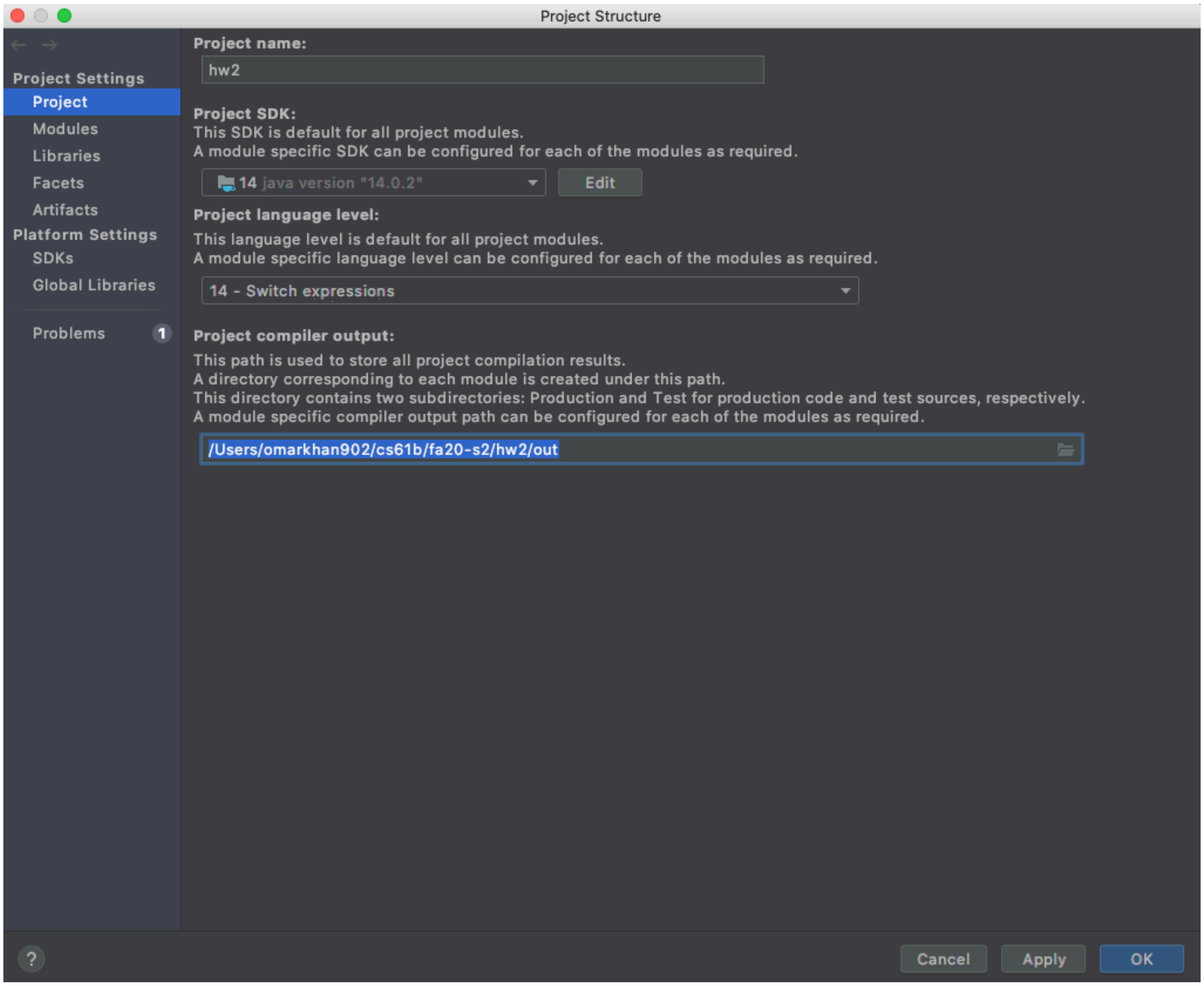


Error

Cannot start compilation: the output path is not specified for module "hw2". Specify the output path in the Project Structure dialog.

IntelliJ puts all the compiled Java `.class` files in a special folder called `out`. You may have seen it before. Usually, IntelliJ will be able to determine where to put the `out` folder, but sometimes it cannot and needs your help. To specify where to put the out foler, go to:

"File" > "Project Structure" > "Project" > "Project compiler output" and simply add the output path. The output path should be of the form `path/to/assignment/out` where the prefix is dependent on the path to your assignment (hw, lab, proj, etc) and the suffix must be `/out` . Example for homework 2:



Project name:
hw2

Project SDK:
This SDK is default for all project modules.
A module specific SDK can be configured for each of the modules as required.
14 java version "14.0.2"

Project language level:
This language level is default for all project modules.
A module specific language level can be configured for each of the modules as required.
14 - Switch expressions

Project compiler output:
This path is used to store all project compilation results.
A directory corresponding to each module is created under this path.
This directory contains two subdirectories: Production and Test for production code and test sources, respectively.
A module specific compiler output path can be configured for each of the modules as required.
/Users/omarkhan902/cs61b/fa20-s2/hw2/out

Everything looks right but it still doesn't work!

Sometimes the easiest thing is to simply do it all over again. Even if you know you just did everything correctly, starting over very often just fixes the problem. First close the project ("File" -> "Close Project"), quit the IntelliJ application, and then reimport the project from the beginning.

Everything looks wrong and nothing works!

Sometimes, the easiest thing is to simply do it ALL over again. Specifically, we need to purge Intellij's memory and make it forget that our project ever existed.

Intellij stores project information as `.idea` folders and `.iml` files. To make it forget our project, we'll be deleting those:

1. Close the project ("File" -> "Close Project")
2. Quit Intellij
3. In File Explorer / Finder, go to the project folder
4. Now, you'll need to show hidden files, so that `.idea` and `.iml` show up:
   - Windows 10 and below: guide
   - Windows 11: guide
   - Mac: Press Command + Shift + . (period) guide
5. Delete any `.idea` folders and `.iml` files you see
6. Re-open the project in Intellij

At this point, the project should look normal, as if you're opening it for the first time (you'll need to import libraries again, if applicable). However, if things still look wrong, you may need to do the following:

1. Open up Project Structure ("File" -> "Project Structure")
2. In the "Project" tab:
   - Set SDK and Language Level
   - Choose Compiler Output folder to be `<path>/<to>/<project>/out` . You may have to create this folder yourself
3. In the "Modules" tab: Delete all modules, if any. Then, create a new module with the default settings
4. In the "Libraries" tab: Re-import libraries for the project.
   - At this point, go back and make sure that all settings were properly applied. Once you verified that everything looks correct, close Project Structure
5. In the Project sidebar on the left, right click on the root folder and mark as source (Right click -> "Mark Directory as" -> "Sources Root")