

Eclipse

So far many of you have been using some variant of Vim or Emacs to write your code. While these text editors are sufficient for small classes with few files, there are more powerful tools for large projects with dozens or even hundreds of classes. One of these is known as an **Integrated Development Environment**, or **IDE**. These are usually a combination of tools, such as a text editor, compiler, and debugger all in one. We will be using Eclipse, one of the most popular IDEs that is also free and open-source. Some of you have already been using it.

Installing Eclipse and the JDK

<https://eclipse.org/downloads/>

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

Before we can begin, if you haven't done so already, install the Java Development Kit and Eclipse. The **Java Development Kit**, or **JDK**, allows us to program Java on our computers, and Eclipse is the IDE you'll be using. Click the links above, find the download link for your OS, and follow the instructions to install. For Eclipse, download the "Eclipse IDE for Java Developers."

1. Start up Eclipse and when it says to "Select a workspace", navigate to your desktop and click OK.
2. Exit the Welcome screen by closing the current tab.
3. Create a new Java Project by navigating to *File=>New=>Java Project*. Name your project "My Project". Do not change anything else. Click *Finish*.
4. Make a new class by navigating to *File=>New=>Class*. Name your class "Driver". Do not change anything else. Click *Finish*.

At this point you will notice that Eclipse has already created "public class Driver {}" for you. This is one of the many benefits of using an IDE - many parameter fields and methods will be autocompleted if Eclipse can predict what the programmer is trying to do. Now type or copy and paste the following into the Driver class EXACTLY as it is shown.

```
public static void main(int[] args) {  
    System.out.println("So many errors...");  
    String[] array = new String[5];  
    dividebyzero(array.size);  
}  
public int dividebyzero(int i) {  
    return i/0  
}
```

The Tools of Eclipse

As you can see, there are quite a few errors in this code, but Eclipse will mark them for you. Eclipse is quite powerful in this regard - it detects everything from as simple as missing semicolons or quotation marks, to incorrect parameter types and return types. It can even detect whether the programmer attempts to use a method or variable that does not exist or cannot be accessed.

Fix the errors underlined by red lines, until they are gone. If you are not sure what is causing the error, hover your mouse over the red line and Eclipse will tell you why it is there.

Once you finish, you'll notice that despite getting rid of all the red lines there are still bugs in the code - specifically runtime errors. Though Eclipse is very powerful, it is still un-capable of detecting code that will cause errors at runtime.

Run the code with the errors by clicking the green play button at the top of the window.

As expected an *ArrayIndexOutOfBoundsException* occurs. The *ArithmeticException* is uncaught as well.

At this point some of you may be wondering why the code did not need to be compiled. Well, it was!

Eclipse compiles behind the scenes and more importantly, piece by piece. Every time a change is made, Eclipse recompiles the code that was changed or added, so it can be run instantaneously on demand.

Credit: Albert Tang and Andy Nguyen