# Getting Started in Eclipse for Project 1 for CSE332

1. Open Eclipse; create a workspace if necessary
2. Add the provided Java code to the project; (grab the given wav files as well and store them somewhere you'll remember on the filesystem; they'll come in handy when you need to test your code)
3. There should be little red Xs on some of the files and folders - this means that the code has errors in it. As you edit the code, Eclipse will automatically rebuild and tell you if you've made an error.
4. Find Reverse.java in the project and double-click. The file pops up. Now you can scroll down and hover on little light bulb in the left margin to see what the errors in the code are.
5. Looks like we don't have classes named ListStack or ArrayStack yet - which makes sense, as those are the files you need to write for your assignment. Here's where Eclipse gets really useful. Click on the lightbulb - Eclipse pops up a list of possible remedies.
6. We want to create a new file, so choose the appropriate option. A dialog window pops up with various options - click Finish.
7. Eclipse auto-generates the file ListStack.java (or ArrayStack.java, depending which lightbulb you clicked) with stub methods for all of the methods in the DStack interface. Nifty, huh?
8. At this point I'll let you figure out how to work in Eclipse on your own. For this assignment, you won't really need any of Eclipse's cool features - but feel free to explore!
9. When you've reached a point at which you want to test your code, you'll need to do a bit of extra work in order to pass command-line arguments to the program:
    1. Right-click on the Reverse.java file in the project you created. Go to Run As -> Run Configurations. Create a new Run Configuration by double clicking 'Java Application'
    2. Click on the Arguments tab.
    3. In the Program arguments box, put four arguments, as specified in the project specifications: the type of stack, double or generic, the input file and the output file. or example:

        ```
        array double /homes/iws/username/Desktop/bot.dat /homes/iws/username/Desktop/out.dat
        ```

        (That should all be on one line.) where /homes/iws/username/Desktop is your Linux Desktop. Java will separate your arguments by spaces; that is, if your argument box contains

        first second third

        String[] args will be an array of size 3. Since it is not uncommon for Windows path names to include spaces, this can be a problem; to get around this, you can put arguments in double quotes, like so:

        "first second third"

        Which will result in just one argument; "first second third".
    4. Click Run. The program should run, with output appearing in the lower right window.
    5. To run the program again with the same arguments, press the Run button in the toolbar. If you want to run the program again with different arguments, you'll need to go back to the run dialog and change what you typed in earlier.
    6. You'll need to run Sox on your output files separately.
10. When you've finished the project and want to turn in ArrayStack.java and ListStack.java, an easy way to get the individual files is to click on each one in Eclipse's left-hand window and drag to the Desktop. This copies the file from the Eclipse workspace to /homes/iws/<username>/Desktop.