**Errors and Debugging**

**IDEs**

IDEs (Integrated Development Environments) are a place where you can write code and it can be compiled for the computer to understand. You are probably using jGrasp. Another common IDE for Java development is Eclipse.

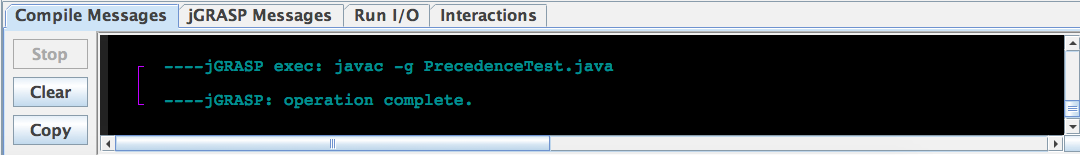
For now, jGrasp is a good IDE because it is very primitive. It basically only functions as a place where you can compile and run your code. Other IDEs do much more than that, but it’s a good idea to learn in a simple IDE.

After you finish writing code, you compile it, which just means that it turns into a format that computers can understand, called byte code. When you run the program, it is just sending the computer commands to begin the program.

**Common Errors**

When you are programming, there will never be a time that you don’t have errors. Some errors will be very confusing and take a long time to figure out, but some are very common and happen to most people.

First of all, this is what it generally looks like when the program compiles successfully:



However, if you have an error, it will show up as (usually) green text in the console when you try to compile the code. These are called compile errors. Most of the time, they are syntax errors. Syntax is the “grammar” rules that the code has to follow. You can get a syntax error by misspelling something, putting a semicolon or bracket in the wrong place, or any other typos.

Here are some common errors:

* ';' expected – This means that you should have a semicolon somewhere and you don’t.
* class, interface, or enum expected – You probably have an extra curly bracket somewhere in the program.
* reached end of file while parsing – You need to add a curly bracket somewhere. A helpful tool for dealing with this or the error above is the CSD tool. It lines up all of the code, unless you are missing or have an extra bracket. It looks like this on the toolbar menu:



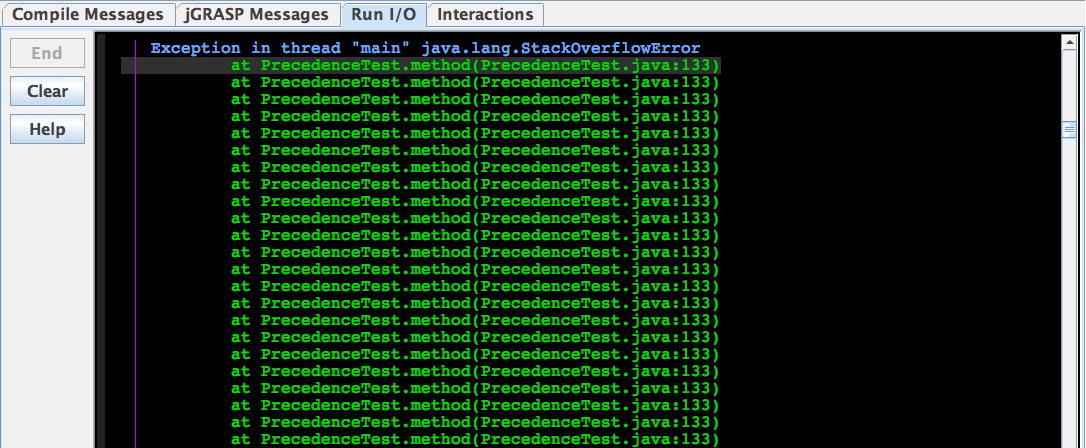
The one on the right is how to get rid of the lines that show up.

* cannot find symbol – You spelled something wrong (most of the time)
* Putting a semicolon at the end of a loop or if-statement causes some weird errors. It might say “else without if” or it might just do weird things that you don’t expect.

There are many other compile errors that can occur. Most of them you will get used to seeing and be able to figure it out.

**Exception Handling**

Some errors don’t occur when you compile, but instead occur when you run the program. These are called exceptions. They show up as long, bright green messages that usually have many more lines than compile errors:



Notice that this time, it shows up in the Run I/O section, and that it is much, much longer. It’s a good idea to build into your code a way to deal with exceptions that doesn’t crash the program.

Right now, the most common exception you’ll have is ArrayIndexOutOfBoundsException, which just means that you are trying to access an index in the array that doesn’t exist. The best way to deal with this is just to go through your code and make sure that you don’t have any calls to the array that are negative or that are bigger than (or equal to) the size of the array.

However, you will come across many other exceptions as you get more advanced with programming. One common exception is the FileNotFoundException, which occurs often if you are trying to access a file from your program. There is one common way to deal with exceptions like this, where the error is not something you necessarily did wrong but something that the user might have done. It is called a try-catch block:

try

{

//The code that might cause the exception

//in the example above, accessing the file

}

catch(Exception e)

{

//Usually print out an error message and exit the program

}

If you know exactly which type of exception you are trying to catch, you can alter the catch part slightly. For our example, we know we are trying to catch the FileNotFound exception;

catch(FileNotFoundException e)

{

//Usually print out an error message and exit the program

}

**Debugging**

Soon enough, you’ll find an error that you can’t solve. That’s where debugging comes in. Most IDEs have a built in debugger. We will briefly go over how to use the debugger in jGrasp:

Click on this:

[The Case of the Messed Up Program](The%20Case%20of%20the%20Messed%20Up%20Program.docx)

**Projects**

Here are some projects that will make sure that you know everything you need to from this lesson:

Don’t forget, you can always ask and answer questions on the website, and if that doesn’t help, bring your questions to our next meeting.