

Re: Need help with my logic

Source: <http://coding.derkeiler.com/Archive/Java/comp.lang.java.programmer/2006-10/msg01263.html>

- *From:* Martin Gregorie <martin@xxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Tue, 10 Oct 2006 13:05:52 +0100
-

Mark Space wrote:

Patricia Shanahan wrote:

See http://home.earthlink.net/~patricia_shanahan/debug/ for my favorite approach.

I didn't read all the way through that, so sorry if I missed this, but there's two things I thought I'd mention, since the OP says she couldn't understand the link.

1. Hand Execution

Learn how to execute your programs by hand. This is probably the best thing you can ever learn if you are a programmer.

Print out your program and go through it line by line. Write variable names on a piece of paper as you encounter them, and write their value next to them. When the variable changes value later, cross out the old value and write the new one next to it on the right.

This will solve a lot of problems, and also make you much better about designing programs in the future.

2. Divide and Conquer

Once your programs get very large, you really can't hand execute the whole thing any more. Learn how to divide your program into chunks (modules, classes, methods, subroutines, etc....) Then make sure each chunk is working correctly before putting them all together. This will save you a lot of time when you write (and debug) large programs.

Also see "Unit Testing."

3. Diagnostic tracing:

Add a boolean debug variable to your classes to control diagnostic output. Set it either through the constructor or with a separate method:

Re: Need help with my logic

```
void setDebug(boolean d)
{
debug = d;
}
```

In each method put at least one debugging statement at the head of a method that accepts information to show the arguments:

```
if (debug)
System.stderr.println("method(" + variable + ", "...");
```

and at the tail of a method that returns anything put a debugging statement to show that is returned:

```
if (debug)
System.stderr.println(retval + " = method());
```

If the method is short, a single statement at the end can often be used to handle both the arguments and the returned value. You might also trace the result of particularly significant operations within the method and in the constructor if it does anything significant apart from initializing variables.

Now you can easily trace the execution flow through your program and see results at significant points within methods. You'll find that this approach is **much** faster than stepping through the code with a debugger if the application is of any significant size and – bonus – can be activated in a live environment to diagnose problems caused by unexpected data. I normally use a `-d` option passed into the `main()` method to control debugging throughout the program for exactly this reason. The "if (debug) statement;" construct adds almost no runtime overhead.

4. Learn about good program design and development.

Get a copy of Kernighan & Pike's "The Practice of Programming" (Addison Wesley). Its applicable to all modern programming languages, contains examples in Java, C and C++ and is very good on all aspects of programming from design to testing and debugging.

HTH

--

martin@ | Martin Gregorie
gregorie. | Essex, UK
org |

.