Virtual Machine I Translator Notes

The actual translation will happen in VirtualMachineTranslator, using hasMoreCommands(), advance(), commandType(), writeArithmetic(), writePushPop(), arg1() and arg2(), called on the appropriate objects. This will be similar to what you did with the assembler.

Suggestion for organizing your Hack assembly translations:

```
// Basic Hack assembly sequence for the VM's unary arithmetic operators.
private final String[] UNARY = new String[] { "@SP\nA=M-1\nM=", "M\n" };
```

To use, write something like:

```
String out = UNARY[0] + "-" + UNARY[1];
```

This results in the translation for neg.

pointer segment -- two words starting at address 3. Handle the same way as temp.

Remember that:

```
local --> LCL
argument --> ARG
etc.
```

In Parser, the VM command needs to be split. One way is to use a Scanner, but that's overkill. Use String's split method with the regex "\\s+" to split on whitespace.

Simple testing -- Create a vm file, say foo.vm, put a few VM commands in it, and translate. Study the translation.

For example, use the commands in our Google doc. To test further, use different segment names and the different binary arithmetic instructions, etc.