PARROT - BRANCHES

Code gets a little boring without flow control; for starters, Parrot knows about branching and labels. The branch op is equivalent to Perl's goto:

```
         branch TERRY
JOHN:    print "fjords\n"
         branch END
MICHAEL: print " pining"
         branch GRAHAM
TERRY:   print "It's"
         branch MICHAEL
GRAHAM:  print " for the "
         branch JOHN
END:     end
```

It can also perform simple tests to see whether a register contains a true value:

```
         set I1, 12
         set I2, 5
         mod I3, I2, I2
         if I3, REMAIND, DIVISOR

REMAIND: print "5 divides 12 with remainder "
         print I3
         branch DONE

DIVISOR: print "5 is an integer divisor of 12"

DONE:    print "\n"
         end
```

Here's what that would look like in Perl, for comparison:

```
$i1 = 12;
$i2 = 5;
$i3 = $i1 % $i2;

if ($i3) {
   print "5 divides 12 with remainder ";
   print $i3;
} else {
   print "5 is an integer divisor of 12";
}

print "\n";
exit;
```

Parrot Operator

We have the full range of numeric comparators: eq, ne, lt, gt, le and ge. Note that you can't use these operators on arguments of disparate types; you may even need to add the suffix _i or _n to the op, to tell it what type of argument you are using, although the assembler ought to divine this for you, by the time you read this.