Perl Lesson 2: Loops, Input, and Debugging

L615

Spring 2013

1. Some basic control operators: if and while

   • if — simply tests if a condition is true
     – Comparison operators for numbers: ==, !=, <, >, <=, >=
     – Comparison operators for strings: eq, ne, lt, gt, le, ge

     if ($a >= $b) {
         print "$a is larger than $b"
     }
     # note the parentheses around the condition!
     if ($name1 lt $name2) {
         print "$name1 alphabetically precedes $name2"
     }

   • while — repeat a block of code as long as a condition is true

     $counter = 1;
     while ($counter < 11) {
         print "I can count to $counter"
         ++$counter;
     }

2. Basic Input

   • <STDIN> — get user input: reads up the first newline

     print "Enter a value"
     $user_value = <STDIN>;

   • chomp — there’s a newline for every <STDIN> invocation, so we like to chomp that off

     print "Enter some text"
     $line = <STDIN>
     chomp($line);

     # or put it all into one line:
     chomp($line = <STDIN>);

   • Can loop over all input lines (use Ctrl-D on Unix, Ctrl-Z on Windows to indicate end of input):

     while (<STDIN>) {
         # $_ = special default variable
         print "Your line is: ";
     }
3. `perl -w` — turns warnings on

```ruby
$a = 1;
print "$a\n";
print "$b\n";
```

- `perl temp.pl` prints out 1
- `perl -w temp.pl` prints out a warning: 'Name "main::b" used only once: possible typo at temp.pl line 5.'
  - Very useful in tracking down what’s going wrong

4. When your program fails to run ...

- Check that each command ends with a semi-colon
- If Perl complains about a particular line, verify that every line up to that one works as you expect it to:
  - Print out values of different variables up to that point
  - Try commenting out that line and seeing if your program runs → the problem could actually be elsewhere
- Try as much as you can to understand the Perl complaint
  - If Perl says there is a syntax error, there really is an error (and it isn’t that your computer’s broken)—so, verify that each bit of syntax is correct
  - Use the internet to search for that error line, or for general tips about fixing errors in Perl, e.g., [http://www.cs.cf.ac.uk/Dave/PERL/node146.html](http://www.cs.cf.ac.uk/Dave/PERL/node146.html)

5. When your program runs, but doesn’t work as you expect it to ...

- As above, printing out variable values at each step and/or commenting out lines will help you track down where something is going wrong
- Walk through the program by hand
  - Did you write the program you wanted to write?
- Get a fresh set of eyes to help out