

# Scripting Languages G22.3033-002 Summer 2008

## hw03

Assigned Th 6/5/2008, due Fr 6/13 at 9pm. 50 points.

<http://www.cs.nyu.edu/courses/summer08/G22.3033-002/>

### Homework instructions

Homeworks are due on Fridays at 9pm. This deadline will be strictly enforced.

Email your answers to Robert Soulé [robert.soule@gmail.com](mailto:robert.soule@gmail.com). Please put your solutions to VBA programming problems in a powerpoint presentation. For all other questions (including programming problems in other languages), just send a simple text file, such as what you get when using Emacs, Vi, Notepad, or the "save as text only" feature in Word.

Please make sure that your code works with the compilers and tools installed at CIWW. Specifically, please test:

#### JAVASCRIPT

Your JavaScript code must run with **both** Mozilla's Firefox browser **and** with Microsoft's Internet Explorer browser.

PHP Your PHP code must run with PHP 5.1.6 on the CIMS web servers, see <http://www.cims.nyu.edu/systems/userservices/webhosting/>.

PERL Your Perl code must run with Perl 5.8.8 for Linux/x86, for example on doowop1 (see <http://www.cims.nyu.edu/systems/resources/computeservers/>).

VBA Your VBA code must run with Microsoft Office 2003 for Windows on the machines in the labs CIWW 502 or CIWW 624.

### Reading assignments

#### Read for lecture on 6/12:

Gilad Bracha. Pluggable Type Systems. OOPSLA Workshop on Revival of Dynamic Languages (RDL), 2004. Available online at <http://pico.vub.ac.be/~wdmeuter/RDL04/papers/Bracha.pdf>

### Concept questions

#### hw03-1 Regular expressions

(7+6 = 13 points)

- Consider the following Perl regular expression:  $(0[xX])?[a-zA-F0-9]^+$ . Rewrite this regular expression using only the "essential" features of formal regular expressions.
- Write a regular expression that captures the PID and PPID from a line of output of the "ps -l" command. For example, consider the following output:

```
doowop1:~> ps -l
 F S  UID    PID  PPID C  PRI  NI ADDR SZ  WCHAN TTY      TIME CMD
 0 S  1088  23431 23421 0   75   0 - 18991 wait pts/4 00:00:00 bash
 0 R  1088  23641 23431 0   77   0 - 15863 -    pts/4 00:00:00 ps
```

The PID and PPID of the line of output for bash are 23431 and 23421.

## hw03-2 Static and dynamic scoping

(5 + 7 = 12 points)

- a. What does the following Perl program print, and why?

```
#!/usr/bin/perl -w
$a = "-";
sub f {
    our $a = "f";
    sub g { local $a = "g"; h() }
    sub h { print $a, "\n" }
    g();
    print $a, "\n"
}
f()
```

- b. When a variable in a Perl function has only an implicit declaration, how is it scoped? Give an example that illustrates your answer.

## Programming exercises

This homework practices the “How to learn a language” steps from the lecture on 5/29/2008:

### I. Use peers & gurus.

Ask around among your friends to find out who knows Perl. Try learning things for yourself, but if you get stuck, ask someone for help.

### II. Install tools.

Perl is already installed on the CIMS Linux machines, so you just need to double-check that the interpreter works for you. Try it on Linux/x86 machines, for example, `doowop1`. Here is a list of compute servers: <http://www.cims.nyu.edu/systems/resources/computeservers/>.

### III. Read tutorial.

This was last week’s reading assignment: Kirrily “Skud” Robert. `perlintro(1)` man page. Perl 5 documentation. Available on machines that have Perl properly installed, and also available online at <http://perldoc.perl.org/perlintro.html>.

### IV. Find language & library reference.

The easiest place to look are the man pages, including `perlsyn(1)`, `perlop(1)`, `perlfunc(1)`, `perlre(1)`. See `perl(1)` for an overview.

### V. Read example programs.

There was example code on the lecture slides. Problem hw03-3 below asks you to read and understand another small script.

### VI. Write example programs.

Problems hw03-4 and hw03-5 below ask you to write some Perl code that exercises core features.

## VII. Understand error messages.

As you solve the programming problems (hw03-4 and hw03-5), you should pay attention to what error messages you get. If any error message is confusing, remember it so that it is familiar the next time you get it.

## VIII. Practice

There will be more Perl programming exercises in hw04. But you should also think about little projects of your own. Ideally, Perl will make you more productive at things you have to do anyway outside of this class.

### hw03-3 CSV to fixed-width (Perl)

(2+0+2+2 = 6 points) Consider the following Perl script:

```
#!/usr/bin/perl -w
while (<>) {
    chomp;
    push @rows, [ split /\s/ ];
}
for $row (@rows) {
    for ($i = 0; $i < @$row; $i++) {
        $w = length $row->[$i];
        $widths[$i] = $w if !$widths[$i] || $w > $widths[$i];
    }
}
for $row (@rows) {
    for ($i = 0; $i < @$row; $i++) {
        print " " x (1 + $widths[$i] - length $row->[$i]);
        print $row->[$i];
    }
    print "\n";
}
```

- a. Run the script. What does it print for the following input? (To answer this question, you need to put the script in a file, for example, `csv2fixed.pl`, and the input into another file, for example, `table.csv`. Then, you do `perl csv2fixed.pl < input.csv`).

```
0, 0, 10, 0
10, 0, 10, 0
10, -10, 10, 0
```

- b. Look up functions (what is `chomp`, `push`, `split`), operators (what is “`x`”), and control constructs (is the “`if`” a proper statement or a statement modifier?) in the reference.
- c. Where does this Perl script use the default variable `$_`?
- d. What does `@$row` return, and why? (This might be one of those cases where you need to ask a Perl-guru if you can’t figure it out yourself.)

### hw03-4 Average column (Perl)

(8 points) Extend the script from problem hw03-3 so it computes the average of each column, and prints it as an added row at the bottom of the output. For example, given this input:

```
3, 4, 3
0, 0, 0
4, 5, 2.5
2, -1, 4
```

your script should print this output:

```
3 4 3
0 0 0
4 5 2.5
2 -1 4
2.25 2 2.375
```

For example, the average of column 3,0,4,2 is 2.25.

### hw03-5 Euclidian distance (Perl)

(8 points) Extend the script from problem hw03-3 so it computes the Euclidian distance of each row to the first row, and prints it as an added column at the right of the output. For example, given this input:

```
3, 4, 3
0, 0, 0
4, 5, 2.5
2, -1, 4
```

your script should print this output:

```
3 4 3 0
0 0 0 5.8309518948453
4 5 2.5 1.5
2 -1 4 5.19615242270663
```

For example, the first row has zero distance from itself. The second row has Euclidian distance 5.8309518948453 from the first row.