Today Topics

• Algorithmic thinking: a worked example
• Perl References
• Perl Modules: cpanm

Reminder:
• first programming homework due tomorrow at midnight!
I am a 3rd year doctoral student in the Accounting department. I am looking for a graduate student that is able to write a simple perl script for me. After speaking with individuals who are familiar with coding, it is my understanding that my task is a simple task for someone who is proficient at perl. I am coauthoring a project with a professor in the department and I can offer compensation to the student. Could you send me an email of a student who would be able and interested?

Spencer Young

• spencery@email.arizona.edu
Case study: Repeated Word Detection

- Microsoft Word:
  - I saw the cat on the mat.
  - The the cat sat
  - I wish that that question had an answer.
  - He had had too many beers
  - Buffalo buffalo Buffalo buffalo buffalo buffalo Buffalo
Case study: Repeated Word Detection

- Key: think algorithmically...
  - think of a specific example first

\[ w_1 \quad w_2 \quad w_3 \quad w_4 \quad w_5 \]

- Compare \( w_1 \) with \( w_2 \)
- Compare \( w_2 \) with \( w_3 \)
- Compare \( w_3 \) with \( w_4 \)
- Compare \( w_4 \) with \( w_5 \)
Case study: Repeated Word Detection

• Generalize specific example, then code it up

\[ \text{array: } @\text{words} \]

\[ w_1 \ w_2 \ w_3 \ldots \ w_{n-1} \ w_n \]

Compare \( w_1 \) with \( w_{1+1} \)

Compare \( w_2 \) with \( w_{2+1} \)

\ldots

Compare \( w_{n-2} \) with \( w_{n-2+1} \)

Compare \( w_{n-1} \) with \( w_n \)

"for" loop implementation

Array indices start from 0...

Array indices end just before $\#\text{words}$...
Case study: Repeated Word Detection

```perl
1 open($fh, $ARGV[0]) or die "ARGV[0] not found!\n";
2
3
4 while ($line = <$fh>) {
5
6
7
8
9
10
11
12
13
14
15}
```
Case study: Repeated Word Detection

```perl
1 open($fh, $ARGV[0]) or die "$ARGV[0] not found!\n"
2
3
4 while ($line = <$fh>) {
5    chomp $line;
6    chop $line if (substr($line,-1) eq ".");
7
8
9
10
11
12
13
14
15}
```
Case study: Repeated Word Detection

```perl
open($fh, $ARGV[0]) or die "$ARGV[0] not found!\n"

while ($line = <$fh>) {
    chomp $line;
    chop $line if (substr($line,-1) eq ".");
    @words = split " ", $line;
}
```
Case study: Repeated Word Detection

```perl
open($fh, $ARGV[0]) or die "$ARGV[0] not found!\n";

while ($line = <$fh>) {
    chomp $line;
    chop $line if (substr($line,-1) eq ".");

    @words = split " ", $line;
    for ($i=0;$i<#words;$i++) {
        
    }
}
```
Case study: Repeated Word Detection

```perl
open($fh, $ARGV[0]) or die "$ARGV[0] not found!\n";

while ($line = <$fh>) {
    chomp $line;
    chop $line if (substr($line, -1) eq ".");

    @words = split " ", $line;
    for ($i=0; $i<@words; $i++) {
        if (lc($words[$i]) eq lc($words[$i+1])) {
        }
    }
}
```
Case study: Repeated Word Detection

```perl
open($fh, $ARGV[0]) or die "$ARGV[0] not found!\n";

while ($line = <$fh>) {
    chomp $line;
    chop $line if (substr($line,-1) eq ".");

    @words = split " ", $line;
    for ($i=0; $i<@words; $i++) {
        if (lc($words[$i]) eq lc($words[$i+1])) {
            print "Line $line_num: \"$words[$i] at position \", $i+1, \" repeated\n"
        }
    }
}
```
Case study: Repeated Word Detection

```perl
1 open($fh, $ARGV[0]) or die "$ARGV[0] not found!\n";
2 $line_num = 0;
3
4 while ($line = <$fh>) {
  5     chomp $line;
  6     chop $line if (substr($line,-1) eq ".");
  7     $line_num++;
  8
  9     @words = split " ", $line;
10    for ($i=0;$i<=$#words;$i++) {
11       if (lc($words[$i]) eq lc($words[$i+1])) {
12          print "Line $line_num: "$words[$i] at position ", $i+1, " repeated\n";
13     }
14 } }
```

*a decent first pass ...*
Case study: Repeated Word Detection

• Sample data file:

> Love for the the Bronx Bombers bubbled up in the absence of some local franchise.  
> On on on the table, we have some eggs.  
> I have home sharing turned on, on both my computer and my apple tv.  
> I wish that that question had an answer.  
> Because he had had too many beers already, he skipped the Friday office happy hour.  
> Buffalo buffalo Buffalo buffalo buffalo buffalo buffalo buffalo Buffalo buffalo.

• Output:

```
SBS2893:ling538-14 sandiway$ perl repeated.perl hw2.txt
Line 1: 'the at position 3 repeated
Line 2: 'On at position 1 repeated
Line 2: 'on at position 2 repeated
Line 4: 'that at position 3 repeated
Line 5: 'had at position 3 repeated
Line 6: 'Buffalo at position 1 repeated
Line 6: 'buffalo at position 2 repeated
Line 6: 'Buffalo at position 3 repeated
Line 6: 'buffalo at position 4 repeated
Line 6: 'buffalo at position 5 repeated
Line 6: 'buffalo at position 6 repeated
Line 6: 'Buffalo at position 7 repeated
```
Case study: Repeated Word Detection

- Second try: *merging multiple occurrences*
- Sample data file:

  Love for the the Bronx Bombers bubbled up in the absence of some local franchise. On on on the table, we have some eggs. I have home sharing turned on, on both my computer and my apple tv. I wish that that question had an answer. Because he had had too many beers already, he skipped the Friday office happy hour. Buffalo buffalo Buffalo buffalo buffalo buffalo buffalo buffalo.

- Output:

  ```bash
  SBS2893:ling538-14 sandiway$ perl repeated2.pl hw2.txt
  Line 1: 'the at position 3 occurs 2 times
  Line 2: 'On at position 1 occurs 3 times
  Line 4: 'that at position 3 occurs 2 times
  Line 5: 'had at position 3 occurs 2 times
  Line 6: 'Buffalo at position 1 occurs 8 times
  ```
Case study: Repeated Word Detection

- Second try... *merging multiple occurrences*

```perl
1 open($fh, ARGV[0]) or die "ARGV[0] not found!\n";
2 $line_num = 0;
3
4 while ($line = <$fh>) {
5    chomp $line;
6    chop $line if (substr($line, -1) eq ".");
7    $line_num++;
8
9    $repeated_word = ""; $repeats = 0;
10   @words = split " ", $line;
11   for ($i=0; $i<@words; $i++) {
12      if (lc($words[$i]) ne lc($words[$i+1])) {
13          if ($repeats > 0) {
14              print "Line $line_num: '$repeated_word' at position $position occurs $repeats times\n";
15              $repeated_word = ""; $repeats = 0;
16          } else {
17              if ($repeated_word) {
18                  $repeats++
19              } else {
20                  $repeated_word = $words[$i]; $position = $i+1; $repeats = 2;
21              }
22          }
23      } else {
24          if ($repeated_word) {
25              $repeats++
26          } else {
27              $repeated_word = $words[$i]; $position = $i+1; $repeats = 2;
28          }
29      }
30  }
31
32  if ($repeated_word) {
33      print "Line $line_num: '$repeated_word' at position $position occurs $repeats times\n";
34  }
```
Case study: Repeated Word Detection

- Third try. *implementing a simple table of exceptions*

```perl
1 open($fh, $ARGV[0]) or die "$ARGV[0] not found!\n";
2
3 $table = ("had", 2, "that", 2);
4
5 $line_num = 0;
6
7 while ($line = <$fh>) {
8     chomp $line;
9     chop $line if (substr($line,-1) eq ".");
10
11     $line_num++;
12
13     $repeated_word = ""; $repeats = 0;
14
15     @words = split " ", $line;
16     for ($i=0; $i<@words; $i++) {
17         if (lc($words[$i]) ne lc($words[$i+1])) {
18             if ($repeats > $table[$repeated_word]) {
19                 print "Line $line_num: $repeated_word at position $position occurs $repeats times\n";
20             }
21             $repeated_word = ""; $repeats = 0;
22         } else {
23             if ($repeated_word) {
24                 $repeats++
25             } else {
26                 $repeated_word = $words[$i]; $position = $i+1; $repeats = 2;
27             }
28         }
29     } if ($repeats > 0) {
30         print "Line $line_num: $repeated_word at position $position occurs $repeats times\n";
31     }
32 }
```
Case study: Repeated Word Detection

- Third try. *table of exceptions*
- Sample data file:

```
Love for the the Bronx Bombers bubbled up in the absence of some local franchise.
On on on the table, we have some eggs.
I have home sharing turned on, on both my computer and my apple tv.
I wish that that question had an answer.
Because he had had too many beers already, he skipped the Friday office happy hour.
Buffalo buffalo Buffalo buffalo buffalo buffalo Buffalo buffalo.
```

- Output:

```
SBS2893:ling538-14 sandiway$ perl repeated3.pl hw2.txt
Line 1: the at position 3 occurs 2 times
Line 2: On at position 1 occurs 3 times
Line 6: Buffalo at position 1 occurs 8 times
```

```
SBS2893:ling538-14 sandiway$ perl repeated2.pl hw2.txt
Line 1: 'the at position 3 occurs 2 times
Line 2: 'On at position 1 occurs 3 times
Line 4: 'that at position 3 occurs 2 times
Line 5: 'had at position 3 occurs 2 times
Line 6: 'Buffalo at position 1 occurs 8 times
```
More complex data structures

• Arrays and hashes may only contain scalars

• **Question:** How to accomplish nesting, i.e. put non-scalars inside?

• **Answer:** use references (pointers), which happen to be scalars

  (actually a reference is just an unsigned number: computer address)

• [http://perldoc.perl.org/perlreftut.html](http://perldoc.perl.org/perlreftut.html)
References

• Two ways to make a reference:

Remember **bracketing** when initializing:

- ( ) List – can be used for both arrays and hashes
- [] Reference to an array
- {} Reference to a hash
References

• Example: array of arrays

```c
$a = ([1, 2, 3],
     [4, 5, 6],
     [7, 8, 9]);
```

**Note:** uses Make Rule 2: square brackets

• Let’s figure out what the following mean:

```c
$a[1],
$s{a[1]}[1],
$a[1] => [0],
$a[1][2],
```

**de-reference**

<table>
<thead>
<tr>
<th>Arrow Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>In between two subscripts, the arrow is optional.</td>
</tr>
<tr>
<td>Instead of $a[1] =&gt; [2], we can write $a[1][2];</td>
</tr>
</tbody>
</table>
References

- Looping (using for/foreach) with array/hash references:

```
1. for my $element (@array) {
2.     ...
3. }
```

- Replace the array name, `@array`, with the reference:

```
1. for my $element (@{$aref}) {
2.     ...
3. }
```

- `${$aref}[3]` is too hard to read, so you can write `$aref->[3]` instead.
- `${$shref}{red}` is too hard to read, so you can write `$shref->{red}` instead.

**Careful!** `$aref->[3]` and `$aref[3]` are different
References

• Perl code:
  
  ```perl
  $a = [1, 2, 3, 4, 5];
  print $a+1, "\n";
  ```

• What happens here?
References

• Looping (using **for/foreach**) with array/hash references:

```perl
1. for my $key (keys %hash) {  
   2.   print "$key => $hash{$key}\n";
   3. }

And then replace the hash name with the reference:

```perl
1. for my $key (keys %$href) {  
   2.   print "$key => @{$href}{$key}\n";
   3. }
```

`${saref}[3]` is too hard to read, so you can write `$aref->[3]` instead.

`${$href}{red}` is too hard to read, so you can write `$href->{red}` instead.

**Careful!** `$href->{‘red’}` and `$href{‘red’}` are different.
CPAN Minus

- [https://www.cpan.org](https://www.cpan.org)

Various installers for Perl modules:
- `cpanm` (CPAN Minus)
- `install (various ways)
  1. cd ~/bin  *(a directory where you want it to be)*
  2. curl -L https://cpanmin.us/  --o cpanm
  3. chmod +x cpanm
- `cpan`
- `cpanp`
CPAN Minus

- Ubuntu:

```
$ which cpanm
```

The program 'cpanm' is currently not installed. You can install it by typing:

```
sudo apt-get install cpanminus
```

**INSTALLATION**

There are several ways to install cpanminus to your system.

**Package management system**

There are Debian packages, RPMs, FreeBSD ports, and packages for other operation systems available. If you want to use the package management system, search for cpanminus and use the appropriate command to install. This makes it easy to install cpanm to your system without thinking about where to install, and later upgrade.

**Installing to system perl**

You can also use the latest cpanminus to install cpanminus itself:

```
curl -L https://cpanmin.us | perl - --sudo App::cpanminus
```

This will install cpanm to your bin directory like `usr/local/bin` (unless you configured INSTALL_BASE with `local::lib`), so you probably need the `--sudo` option.
Where are Perl modules stored?

- @INC array variable

```perl
perl -e 'print "$_\n" foreach @INC'
```

- `/opt/local/lib/perl5/site_perl/5.24/darwin-thread-multi-2level`
- `/opt/local/lib/perl5/site_perl/5.24`
- `/opt/local/lib/perl5/vendor_perl/5.24/darwin-thread-multi-2level`
- `/opt/local/lib/perl5/vendor_perl/5.24`
- `/opt/local/lib/perl5/5.24/darwin-thread-multi-2level`
- `/opt/local/lib/perl5/5.24`

```
perldoc -lm Lingua::EN::CMUDict (from lecture2)
```

- `/opt/local/lib/perl5/site_perl/5.24/Lingua/EN/CMUDict.pm`
Example: a day of week calculator

```perl
dow.perl

use Date::Calc qw(:all);

if ($#ARGV != 2) {
    die "usage: month day year\n"
}
$month = $ARGV[0];
$day = $ARGV[1];
$year = $ARGV[2];

$dow = Day_of_Week_to_Text(Day_of_Week($year, $month, $day));
print "$month/$day/$year falls on a $dow\n";
```

```
perl dow.perl

Can't locate Date/Calc.pm in @INC (you may need to install the Date::Calc module) (@INC contains:
/opt/local/lib/perl5/site_perl/5.24/dar
win-thread-multi-2level
/opt/local/lib/perl5/site_perl/5.24
/opt/local/lib/perl5/vendor_perl/5.24/dar
win-thread-multi-2level
/opt/local/lib/perl5/vendor_perl/5.24
/opt/local/lib/perl5/5.24/darwin-thread-multi-2level
/opt/local/lib/perl5/5.24 .) at dow.perl
line 1.
BEGIN failed--compilation aborted at dow.perl line 1.
```
Example: a day of week calculator

```
sudo cpanm Date::Calc
Password:
--> Working on Date::Calc
Fetching http://www.cpan.org/authors/id/S/ST/STBEY/Date-Calc-6.4.tar.gz ... OK
Configuring Date-Calc-6.4 ... OK
==> Found dependencies: Bit::Vector, Carp::Clan
--> Working on Bit::Vector
Fetching http://www.cpan.org/authors/id/S/ST/STBEY/Bit-Vector-7.4.tar.gz ... OK
Configuring Bit-Vector-7.4 ... OK
==> Found dependencies: Carp::Clan
--> Working on Carp::Clan
Fetching http://www.cpan.org/authors/id/K/KE/KENTNL/Carp-Clan-6.06.tar.gz ... OK
Configuring Carp-Clan-6.06 ... OK
==> Found dependencies: Test::Exception
--> Working on Test::Exception
Fetching http://www.cpan.org/authors/id/E/EX/EXODIST/Test-Exception-0.43.tar.gz ... OK
Configuring Test-Exception-0.43 ... OK
==> Found dependencies: Sub::Uplevel
--> Working on Sub::Uplevel
Fetching http://www.cpan.org/authors/id/D/DA/DAGOLDEN/Sub-Uplevel-0.2800.tar.gz ... OK
Configuring Sub-Uplevel-0.2800 ... OK
Building and testing Sub-Uplevel-0.2800 ... OK
Successfully installed Sub-Uplevel-0.2800
Building and testing Test-Exception-0.43 ... OK
Successfully installed Test-Exception-0.43
Building and testing Carp-Clan-6.06 ... OK
Successfully installed Carp-Clan-6.06
Building and testing Bit-Vector-7.4 ... OK
Successfully installed Bit-Vector-7.4
Building and testing Date-Calc-6.4 ... OK
Successfully installed Date-Calc-6.4
5 distributions installed
```

```
perldoc -lm Date::Calc
/opt/local/lib/perl5/site_perl/5.24/Date/Calc.pm
```
Example: a day of week calculator

perl dow.perl
usage: month day year
perl dow.perl 9 12 2017
9/12/2017 falls on a Tuesday
perl dow.perl 9 12 2016
9/12/2016 falls on a Monday
localtime

• Date::Calc

13($year, $month, $day) = Today();
14print "$month/$day/$year is today\n"

• Built-in:

$now = localtime;
print "$now\n";
($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime; # e.g., "Thu Oct 13 04:54:34 1994"
@a = localtime;
print "@a\n";