

LING/C SC/PSYC 438/538

Lecture 9

Sandiway Fong

Today's Topics

- Homework 6 review
- Perl references
- Perl module example: Time and Date

Homework 6 Review: Part 1

- Well-known English spelling rule:
 - I before E except after C
- Orthography:
 - *recei*ve (digraph *ie*)
 - sci*e*nce (I before E after C: exception to the rule, type 1)
 - fore*e*ign (E before I not after C: exception to the rule type 2)

Homework 6 Review: Part 1

- 3letters.txt , 4letters.txt and 5letters.txt:

```
$ perl -e 'open $fh, $ARGV[0]; for (split " ",<$fh>) {print "$_\n" if (index($_, "CIE") >= 0)}' 3letters.txt
```

```
$ perl -e 'open $fh, $ARGV[0]; for (split " ",<$fh>) {print "$_\n" if (index($_, "CIE") >= 0)}' 4letters.txt
```

CIEL

```
$ perl -e 'open $fh, $ARGV[0]; for (split " ",<$fh>) {print "$_\n" if (index($_, "CIE") >= 0)}' 5letters.txt
```

CIELS

ICIER

LOCIE

Make-up Homework Review

- 3letters.txt:

```
$ perl -e 'open $fh, $ARGV[0]; for (split " ", <$fh>) {print "$_\n" if ($_ =~ /^[^C]EI/)}' 3letters.txt
```

```
DEI  
LEI  
REI  
SEI
```

4 + 1 = 5

EIK

EIDE
EIKS
EILD
EINA
EINE
EISH

36 + 6 = 42

105 + 8 = 113

EIDER
EIDOS
EIGHT
EIGNE
EIKED
EIKON
EILDS
EISEL

- 4letters.txt and 5letters.txt:

```
$ perl -e 'open $fh, $ARGV[0]; for (split " ", <$fh>) {print "$_\n" if ($_ =~ /^[^C]EI/)}' 4letters.txt | wc -l
```

36

```
$ perl -e 'open $fh, $ARGV[0]; for (split " ", <$fh>) {print "$_\n" if ($_ =~ /^[^C]EI/)}' 5letters.txt | wc -l
```

105

Homework 6 review: Part 2

- Part A:
 - write a Perl program to remove vowels *a, e, i, o, u* from words typed into the command line. (Don't worry about *y*.)
- Hint: use `split` from previous lectures
- **Example:**

```
Desktop$ perl dv.perl quick brown fox  
qck brwn fx
```

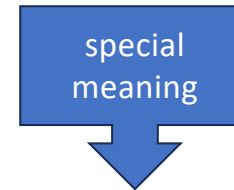
Homework 6 Review: Part 2

- **Basic Idea:**

1. this is a test
2. t h i s i s a t e s t
3. t h s s t s t
4. ths s tst

- **Perl:**

- @ARGV
- for (@ARGV) {@c = split //; ... }
- for \$c (@c) { *if \$c is a vowel, don't print it, else print \$c* }



```
perl -e 'for (@ARGV) {@c = split //; print "@c "; print "\n" } this is a test  
t h i s i s a t e s t'
```

Homework 6 Review: Part 2

```
1my %vowel = qw(a 1 e 1 i 1 o 1 u 1 A 1 E 1 I 1 O 1 U 1);  
2foreach $word (@ARGV) {  
3  
4  
5  
6  
7  
8 }  
9}  
10print "\n"
```

```
1my %vowel = qw(a 1 e 1 i 1 o 1 u 1 A 1 E 1 I 1 O 1 U 1);  
2foreach $word (@ARGV) {  
3  foreach $char (split //, $word) {  
4    unless ($vowel{$char}) {  
5      print $char  
6    }  
7  }  
8  print " "  
9}  
10print "\n"
```


Homework 6 Review: Part 2

- Examples:

- perl hw6.perl suddenly a White Rabbit with pink eyes ran close by her.

sddnly Wht Rbbt wth pnk ys rn cls by hr.

- perl hw6.perl quick brown fox

qck brwn fx

- perl hw6.perl If a sentence is unreadable

f sntnc s nrdbl

Homework 6 Review: Part 2

```
[$ perl -le 'tr/AEIOUaeiou//d for (@ARGV); print "@ARGV"' suddenly a White Rabbit]
with pink eyes ran close by her.
sddnly Wht Rbbt wth pnk ys rn cls by hr.
$ █
```

- *transliterate* for (@ARGV);
 - for-loop on @ARGV with reversed syntax; usually for (@ARGV) {...}
- tr/AEIOUaeiou//d
 - modifies the words stored in @ARGV
- print "@ARGV"
 - puts spaces between each word
- perl -l
 - prints a newline after each print statement

Homework 6 Review: Part 2

- Flag in a loop idea:

- Chars: `s e n t e n c e`

- `$not1st: 0 1 1 1 1 1 1 1`

- Chars: `u n r e a d a b l e`

- `$not1st: 0 1 1 1 1 1 1 1 1 1`

Homework 6 Review: Part 2

flag
unset

flag
set

```
1 my %vowel = qw(a 1 e 1 i 1 o 1 u 1 A 1 E 1 I 1 O 1 U 1);
2 foreach $word (@ARGV) {
3   my $not1st = 0;
4   foreach $char (split //, $word) {
5     unless ($not1st && $vowel{$char}) {
6       print $char
7     }
8     $not1st = 1
9   }
10  print " "
11 }
12 print "\n"
```

Homework 6 Review: Part 2

- Examples:

1. perl hw6b.perl suddenly a White Rabbit with pink eyes
ran close by her.

sddnly a Wht Rbbt wth pnk eys rn cls by hr.

2. perl hw6b.perl quick brown fox

qck brwn fx

3. perl hw6b.perl If a sentence is unreadable

If a sntnc is unrdbl

Homework 6 Review: extra

- Making it switchable (in behavior)
 - Option: `-s` (switch)
 - The `-s` option lets you create your own custom switches. Custom switches are placed after the script name but before any filename arguments. Any custom switches are removed from the `@ARGV` array. Then a scalar variable is named after the switch is created and initialized to 1.

Homework 6 Review: extra

option
-del



```
1 my %vowel = qw(a 1 e 1 i 1 o 1 u 1 A 1 E 1 I 1 O 1 U 1);
2 foreach $word (@ARGV) {
3   my $not1st = $del;
4   foreach $char (split //, $word) {
5     unless ($not1st && %vowel{$char}) {
6       print $char
7     }
8     $not1st = 1
9   }
10  print " "
11}
12 print "\n"
```

Homework 6 Review: extra

- Examples:

```
$ perl hw6c.perl suddenly a White Rabbit with pink eyes ran  
close by her.
```

```
sddnly a Wht Rbbt wth pnk eys rn cls by hr.
```

```
$ perl -s hw6c.perl -del suddenly a White Rabbit with pink  
eyes ran close by her.
```

```
sddnly Wht Rbbt wth pnk ys rn cls by hr.
```

```
$ perl hw6c.perl If a sentence is unreadable
```

```
If a sntnc is unrdbl
```

```
$ perl -s hw6c.perl -del If a sentence is unreadable
```

```
f sntnc s nrdbl
```


Homework 6 Review: extra

- WAR AND PEACE By Leo Tolstoy/Tolstoi

```
$ wc war_and_peace.txt
```

```
65656 563290 3274138 war_and_peace.txt
```

```
$ perl -s disemvowel.perl -del war_and_peace.txt > wr_nd_pc.txt
```

```
$ wc wr_nd_pc.txt
```

```
65656 549620 2386014 wr_nd_pc.txt
```

```
$ perl disemvowel.perl war_and_peace.txt > wr_and_pc.txt
```

```
$ wc wr_and_pc.txt
```

```
65656 563290 2533881 wr_and_pc.txt
```

Python behavior

```
python
```

```
Python 3.9.16 | packaged by conda-  
forge
```

```
>>> a = [1]
```

```
>>> b = a
```

```
>>> b[0] = -1
```

```
>>> b
```

```
[-1]
```

```
>>> a
```

```
[-1]
```

```
>>> a = 1
```

```
>>> b = a
```

```
>>> b = -1
```

```
>>> b
```

```
-1
```

```
>>> a
```

```
1
```

```
>>>
```

Different behavior: How do we reconcile these?

Perl: more complex data structures

- Arrays and hashes may only contain scalars (*Python is way better here*)
- **Question:** How to accomplish nesting, i.e. put non-scalars inside?
- **Answer:** use references (called **pointers** in C), which happen to be scalars
- <http://perldoc.perl.org/perlreftut.html>

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

We have seen this before! Lecture 8: Part of speech example.
`%pos=(likes => ["n","v"], car=>["n"], smiled=>["v"]);`

Perl: References

- Two ways to make a reference:

Remember **bracketing** when initializing:
() List – used for both arrays and hashes
[] Reference to an array
{ } Reference to a hash

Make Rule 1

If you put a `\` in front of a variable, you get a reference to that variable.

```
1. $aref = \@array;      # $aref now holds a reference to @array
2. $href = \%hash;     # $href now holds a reference to %hash
3. $sref = \$scalar;   # $sref now holds a reference to $scalar
```

Make Rule 2

`[ITEMS]` makes a new, anonymous array, and returns a reference to that array. `{ ITEMS }` anonymous hash, and returns a reference to that hash.

```
1. $aref = [ 1, "foo", undef, 13 ];
2.      # $aref now holds a reference to an array
3.
4. $href = { APR => 4, AUG => 8 };
5.      # $href now holds a reference to a hash
```

Perl: References

- Example: array of arrays

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

Note: uses Make Rule 2: square brackets

- Let's figure out what the following mean:

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

de-referencing arrow

Arrow Rule

In between two **subscripts**, the arrow is optional.
Instead of `$a[1]->[2]`, we can write `$a[1][2]`;

Perl: References

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

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

o 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.

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

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

Perl: References

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

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

and then replace the hash name with the reference:

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

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

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

Careful! `$href->{'red'}` and `$href{'red'}` are different.

Perl: References

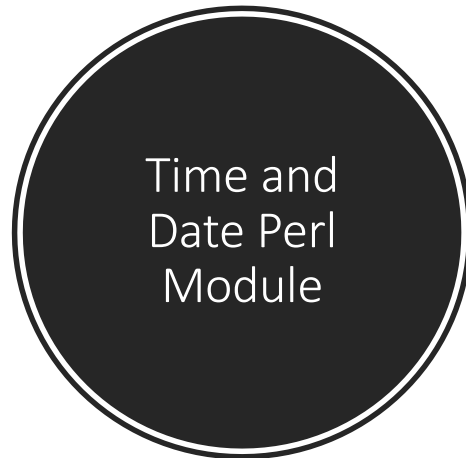
- Perl code:
 `$a = [1, 2, 3, 4, 5]; print $a+1, "\n";`
- What happens here?

```
$ perl -e '$a = [1, 2, 3, 4, 5]; print $a+1, "\n"'  
5033213049
```

```
$ perl -e '$a = [1, 2, 3, 4, 5]; print $a+1, "\n"'  
5628804217
```

```
$ perl -e '$a = [1, 2, 3, 4, 5]; print $a+1, "\n"'  
5276482681
```


Course website:
dow.perl



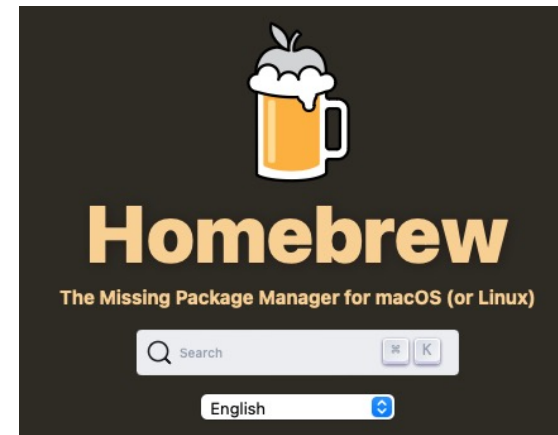
```
1 use Date::Calc qw(:all);  
2 die "usage: month day year\n" if $#ARGV != 2;  
3 ($month, $day, $year) = @ARGV;  
4  
5 $dow = Day_of_Week_to_Text(Day_of_Week($year, $month, $day));  
6 print "$month/$day/$year falls on a $dow\n";  
7  
8 ($year, $month, $day) = Today();  
9 print "$month/$day/$year is today\n";
```

```
$ perl dow.perl  
usage: month day year  
$ perl dow.perl 9 19 2023  
9/19/2023 falls on a Tuesday  
9/18/2023 is today  
$
```

Date/Calc.pm

- On macOS 13.2, homebrew uses the prefix `/opt/homebrew` instead of `/usr/local`
- Verify locations (PATH):
 - `$ which perl`
 - `/opt/homebrew/bin/perl`
 - `$ which cpan`
 - `/opt/homebrew/bin/cpan`
- Verify locations (PATH):
 - `~/.bash_profile` (macOS login)
 - `~/.bashrc` (new Terminal)

<https://brew.sh>




Date/Calc.pm


- Not installed or can't find it:
 - `$ perl dow.perl`
 - Can't locate `Date/Calc.pm` in `@INC` (you may need to install the `Date::Calc` module) (`@INC` contains:
 - `/Users/sandaway/perl5/lib/perl5/darwin-thread-multi-2level`
 - `/Users/sandaway/perl5/lib/perl5`
 - `/opt/homebrew/opt/perl/lib/perl5/site_perl/5.36/darwin-thread-multi-2level`
 - `/opt/homebrew/opt/perl/lib/perl5/site_perl/5.36`
 - `/opt/homebrew/opt/perl/lib/perl5/5.36/darwin-thread-multi-2level`
 - `/opt/homebrew/opt/perl/lib/perl5/5.36`
 - `/opt/homebrew/lib/perl5/site_perl/5.36`) at `dow.perl` line 1.
 - `BEGIN failed--compilation aborted at dow.perl line 1.`

Date/Calc.pm

nuke this?
rm -rf .cpan



- Not installed or can't find it:
 - `cpan Date::Calc`
 - Reading `'/Users/sandiway/.cpan/Metadata'`
 - Database was generated on Thu, 03 Nov 2022 16:54:00 GMT
 - ...
 - CPAN: HTTP::Tiny loaded ok (v0.080)
 - Trying with
 - `/usr/local/bin/wget -O`
 - `"/Users/sandiway/.cpan/sources/authors/01mailrc.txt.gz.tmp26612"`
 - to get
 - `https://cpan.org/authors/01mailrc.txt.gz`
 - `sh: /usr/local/bin/wget: No such file or directory`



shouldn't be going here if homebrew moved
from `/usr/local` to `/opt/homebrew`

cpan Date::Calc

- Install after nuking ~/.cpan:

```
$ cpan Date::Calc
```

```
CPAN.pm requires configuration, but most of it can be done automatically.
```

```
If you answer 'no' below, you will enter an interactive dialog for each
```

```
configuration option instead.
```

```
Would you like to configure as much as possible automatically? [yes]
```

```
...
```

```
Installing /Users/sandaway/perl5/lib/perl5/Date/Calc.pod
```

```
Installing /Users/sandaway/perl5/lib/perl5/Date/Calendar.pm
```

```
Installing /Users/sandaway/perl5/lib/perl5/Date/Calc.pm
```

www.cpan.org

CPAN Comprehensive Perl Archive Network
YOU CAN NEVER HAVE TOO MANY PERL MODULES

Home Modules Ports Perl Source FAQ Mirrors Search: **date::calc** Search

Welcome to CPAN

The Comprehensive Perl Archive Network (CPAN) currently has [213,861 Perl modules](#) in 44,449 distributions, written by 14,417 authors, mirrored on 1 servers.

The archive has been online since October 1995 and is constantly growing.

Search CPAN via

Recent Uploads

- [Text-Treesitter-0.11](#)
- [MIDI-RtMidi-FFI-0.049-TRIAL](#)
- [Mojo-IOLoop-ReadWriteProcess-0.34](#)
- [Catalyst-Plugin-HTML-Scrubber-0.04](#)
- [Connector-1.52](#)
- [List-Util-sglice-0.002](#)
- [List-Util-sglice-0.001](#)
- [Math-BigInt-1.999840](#)

Getting Started

- [Installing Perl Modules](#)
- [Learn Perl](#)

How to contribute

- [Read this](#)
- Visit <https://pause.perl.org/>

Resources

- [Perl Programming language](#)
- [Documentation](#)
- [Mailing Lists](#)
- [FAQ](#)
- [CPAN Repository](#)

- *Looking for date::calc*
- <https://www.cpan.org/modules/INSTALL.html>
 - *can use command cpan on the command line, see also alternatives*

cpan Date::Calc

meta::cpan About Sponsor grep::cpan

Search results for "date::calc"

Date::Calc - Gregorian calendar date calculations 23 ++

* "use Date::Calc qw(Days_in_Year Days_in_Month ...);" * "use Date::Calc qw(:all);" You can use them between the parentheses of the "qw()" operator, or you can use the "all" operator...

STBEY/Date-Calc-6.4 - Mar 07, 2015 - [Search in distribution](#)

- [lib/Date/Calc.pm](#)
- [lib/Date/Calc/Object.pm](#)
- [Date::Calc::PP - pure-Perl plug-in for Date::Calc](#)

[8 more results from Date-Calc »](#)

```
use Date::Calc qw(
  Days_in_Year
  Days_in_Month
  Weeks_in_Year
  leap_year
  check_date
  check_time
  check_business_date
  Day_of_Year
  Date_to_Days
  Day_of_Week
  Week_Number
  Week_of_Year
  Monday_of_Week
  Nth_Weekday_of_Month_Year
  Standard_to_Business
  Business_to_Standard
  Delta_Days
  Delta_DHMS
  Delta_YMD
  Delta_YMDHMS
  N_Delta_YMD
  N_Delta_YMDHMS
  Normalize_DHMS
  Add_Delta_Days
  Add_Delta_DHMS
  Add_Delta_YM
  Add_Delta_YMD
  Add_Delta_YMDHMS
  Add_N_Delta_YMD
  Add_N_Delta_YMDHMS
  System_Clock
  Today
  Now
  Today_and_Now
  This_Year
  Gmtime
```

```
use Date::Calc qw(:all);

Days_in_Year
  $days = Days_in_Year($year,$month);

Days_in_Month
  $days = Days_in_Month($year,$month);

Weeks_in_Year
  $weeks = Weeks_in_Year($year);

leap_year
  if (leap_year($year))

check_date
  if (check_date($year,$month,$day))

check_time
  if (check_time($hour,$min,$sec))

check_business_date
  if (check_business_date($year,$week,$dow))

Day_of_Year
  $doy = Day_of_Year($year,$month,$day);

Date_to_Days
  $days = Date_to_Days($year,$month,$day);

Day_of_Week
  $dow = Day_of_Week($year,$month,$day);

Week_Number
  $week = Week_Number($year,$month,$day);

Week_of_Year
  ($week,$year) = Week_of_Year($year,$month,$day);
  $week = Week_of_Year($year,$month,$day);
```

Time and Date

- Perl function time:

time

Returns the number of non-leap seconds since whatever time the system considers to be the epoch, and `localtime`. On most systems the epoch is 00:00:00 UTC, January 1, 1970; a prominent except

```
perl -le 'print time'  
1695061511
```

- To pause for n seconds, use `sleep n`

```
perl -le '$t1 = time; sleep 10; $t2 = time; print $t2-$t1'  
10
```


Time and Date

- Perl function localtime:

localtime EXPR

localtime

Converts a time as returned by the time function to a 9-element list with the time analyzed for the local time zone.

```
$ perl -le 'print localtime'
3401118812312600
$ perl -le '$now = localtime; print $now'
Mon Sep 18 11:40:55 2023
$ perl -le '@now = localtime; print "@now"'
17 41 11 18 8 123 1 260 0
```

Note: \$mon, \$wday (beginning on Sunday) and \$yday indexed from 0

```
#      0      1      2      3      4      5      6      7      8
my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) =
    localtime(time);
    1900
```

Time and Date

Reference: <https://docs.python.org/3/library/datetime.html#strftime-and-strptime-format-codes>

```
$ python
```

```
Python 3.9.16 | packaged by conda-forge
```

```
>>> import datetime
```

```
>>> d = datetime.date(2023,9,19)
```

```
>>> print(d.strftime("%A"))
```

```
Tuesday
```

```
>>> print(d.strftime("%B %-d %Y"))
```

```
September 19 2023
```

```
>>> datetime.datetime.now()
```

```
datetime.datetime(2023, 9, 18, 11, 31, 8, 445097)
```

Directive	Meaning	Example
<code>%a</code>	Weekday as locale's abbreviated name.	Sun, Mon, ..., Sat (en_US); So, Mo, ..., Sa (de_DE)
<code>%A</code>	Weekday as locale's full name.	Sunday, Monday, ..., Saturday (en_US); Sonntag, Montag, ..., Samstag (de_DE)
<code>%w</code>	Weekday as a decimal number, where 0 is Sunday and 6 is Saturday.	0, 1, ..., 6
<code>%d</code>	Day of the month as a zero-padded decimal number.	01, 02, ..., 31
<code>%b</code>	Month as locale's abbreviated name.	Jan, Feb, ..., Dec (en_US); Jan, Feb, ..., Dez (de_DE)
<code>%B</code>	Month as locale's full name.	January, February, ..., December (en_US); Januar, Februar, ..., Dezember (de_DE)

Ungraded Homework

- Install Perl Module `Lingua::EN::CMUDict` (*CMU Pronouncing dictionary*)
 - <https://metacpan.org/pod/Lingua::EN::CMUDict>

```
use Lingua::EN::CMUDict;  
my $obj = new Lingua::EN::CMUDict;  
print $obj->number_of_syllables("test");
```

Bug in code!
(P UW1 T IH0 N)

- write a program `cmudict.perl` that counts the number of syllables for `$ARGV[0]`

```
$ perl cmudict.perl Balkanization  
Balkanization: 5  
$ perl cmudict.perl Putin  
Putin: 1  
$ perl cmudict.perl Zelenskyy  
Not in cmudict
```

Ungraded Homework

- <http://www.speech.cs.cmu.edu/cgi-bin/cmudict>
- Data format: ARPAbet phoneme set
 - 0 — No stress
 - 1 — Primary stress
 - 2 — Secondary stress
 - Balkanization: 5 syllables
 - B A02 L – K AH0 – N IH0 – Z EY1 – SH AH0 N

● **Look up the pronunciation for a word or phrase in CMUdict (version 0.7b)**

Show Lexical Stress

● BALKANIZATION

● B A02 L K AH0 N IH0 Z EY1 SH AH0 N .

Ungraded Homework

- **\$obj->rhymes(word)**
- In the case of an array being returned, returns all rhymes to the given word. In a scalar context, returns a single rhyme.
- Balkanization rhymes
with implication
impregnation AVOCATION
volition TUITION
COARTICULATION
pollination AUCTION
CONSTELLATION
DISTORTION inspection
...