

LING/C SC/PSYC 438/538

Lecture 9

Prof. Sandiway Fong

Today's Topics

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

Homework 6 Review: Part 1

- Well-known English spelling rule: **I before E except after C**
 - **Wikipedia:**
 - many words were spelled with ⟨ie⟩ and ⟨ei⟩ interchangeably (*Early Modern English period*)
 - the rhyme suggests that the correct order is ⟨ie⟩ unless the preceding letter is ⟨c⟩, in which case it may be ⟨ei⟩.
- Orthography:
 - *rece**ive*** (digraph *ie*)
 - *sci**ence*** (I before E **after C**: exception to the rule, type 1)
 - **look for CIE using index**
 - *for**ei**gn* (E before I **not after C**: exception to the rule type 2)
 - **look for non-C followed by EI**

Homework 6 Review: Part 1: Type 1

6letters.txt:

```
• perl -e 'open $fh, $ARGV[0]; for (split " ",<$fh>)
  {print "$_\n" if (index($_, "CIE") >= 0)}'
6letters.txt | wc -l
```

BOCCIE CIELED CIERGE DECCIE DICIER
FACIES FARCIE GAUCIE ICIEST LACIER
LOCIES OCCIES PACIER RACIER RICIER
SCIENT SPECIE

17

17 / 22971 = 0.074%
17 / 956 = 1.8%

Homework 6 Review: Part 1: Type 1

7letters.txt:

```
• perl -e 'open $fh, $ARGV[0]; for (split " ",<$fh>)  
  {print "$_\\n" if (index($_, "CIE") >= 0)}'  
7letters.txt
```

ANCIENT BACCIES BICCIES BOCCIES CEILING CIERGES DANCIER DARCIES DECCIES DICIEST
EPICIER FACIEND FANCIED FANCIER FANCIES FARCIED FARCIES FLEECIE GAUCIER GAWCIER
GLACIER JUICIER LACIEST LECCIES MERCIES MINCIER MOCCIES ORACIES PACIEST PICCIES
PONCIER PRICIER RACIEST RECCIED RECCIES RICIEST SAUCIER SCIENCE SOCIETY SPACIER
SPECIES SPICIER TALCIER ZINCIER

44

44 / 34506 = 0.13%

44 / 2363 = 1.9%

Homework 6 Review: Part 1: Type 2

6letters.txt:

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

`6letters.txt | wc -l`

- **237**

$237 / 22971 = 1.0\%$

$237 / 255 = 93\%$

7letters.txt:

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

`7letters.txt | wc -l`

- **334**

$334 / 34506 = 1.0\%$

$334 / 358 = 93\%$

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* }

special
meaning



```
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

```
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
4
5
6
7
8
9 }
10 print "\n"
```

```
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     foreach $char (split //, $word) {
4         unless ($vowel{$char}) {
5             print $char
6         }
7     }
8     print " "
9 }
10 print "\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: Part 3

\$ perl hw6p3.perl ppch1.txt	"Why, my dr, y mst knw, Mrs. Lng sys tht Nthrflid is tkn	_my_ fill in lv wth on of thm, and thrfr y mst vst hm as	cmrs. Indd y mst g, fr it will b impssbl fr _s_ t vst	"h, y d nt knw wht I sffr."
It is a trth unvrslly acknwldgd, tht a sngl mn in pssssn	by a yng mn of lrg frtn frm th nrth of Engld; tht h cm	sn as h cms."	hm, if y d nt."	"Bt I hp y will gt ovr it, and lv t s mny yng mn of fr
of a gd frtn mst b in wnt of a wf.	dwn on Mndy in a chs and fr t s th plc, and ws s mch	"s n occsn fr tht. Y and th grls my g--r y my snd	"Y ar ovr scrpls, srly. I dr sy Mr. Bngly will b vry	thsnd a yr cm int th nghbrhd."
Hwvr lttl knwn th flngs or vws of sch a mn my b on hs	dlghtd wth it tht h agrd wth Mr. Mrrs immdtly; tht h is	thm by thmslvs, which prhps will b stll bttr; fr as y ar	gld t s y; and I will snd a fw lns by y t assr hm of my	"t will b n us t us, if twnty sch shld cm, snc y will nt
frst entrng a nghbrhd, ths trth is s will fxd in th mnds	t tk psssn bfr Mchlms, and sm of hs srvnts ar t b	as hndsm as any of thm, Mr. Bngly mght lk y th bst of th	hrtly cnsnt t hs mrryng whchvr h chss of th grls--thgh	vst thm."
of th srndng fmils, tht h is cnsdrd as th rghtfl	in th hs by th end of nxt wk."	prty."	I mst thrw in a gd wrd fr my lttl Lzzy."	"Dpnd upn it, my dr, tht whn thr ar twnty, I will vst thm
prprty of sm on or othr of thr dghtrs.	"Wht is hs nm?"	"My dr, y flttr m. I crtnly _hv_ hd my shr of bty, bt	"dsr y will d n sch thng. Lzzy is nt a bt bttr thn th	all."
"My dr Mr. Bnnt," sd hs ldy t hm on dy, "hv y hrd tht	"Bngly."	I d nt prtnd t b anything extrrdnry nw. Whn a wmn hs fv	othrs: and I am sr sh is nt hlfs hndsm as Jn, nr hlfs	Mr. Bnnt ws s odd a mxtr of qck prts, srcstc hmr,
Nthrflid Prk is lt at lst?"	"s h mrrd or sngl?"	grwn-p dghtrs, sh oght t gv ovr thnkng of hr own bty."	gd-hmrds as Lyd. Bt y ar always gvng _hr_ th prfrnc."	rsrv, and cpcc, tht th exprnc of thr-nd-twnty yrs hd
Mr. Bnnt rpld tht h hd nt.	"h, sngl, my dr, t b sr! A sngl mn of lrg frtn; fr or	"n sch css, a wmn hs nt oftn mch bty t thnk of."	"Thy hv nn of thm mch t rcmmnd thm," rpld h: "thy	bn insffcnt t mk hs wf undrstnd hs chrctr. _Hr_ mnd
"Bt it is," rtrnd sh; "fr Mrs. Lng hs jst bn hr, and sh	fv thsnd a yr. Wht a fn thng fr or grls!"	"Bt, my dr, y mst indd g and s Mr. Bngly whn h cms	all slly and ignrnt lk othr grls; bt Lzzy hs smthng mr	ws lss dffclt t dvlp. Sh ws a wmn of mn undrstndng,
tld m all abt it."	"Hw s? hw cn it affct thm?"	int	of	lttl infrmtn, and uncrtn tmpr. Whn sh ws dscentnd, sh
Mr. Bnnt md n answr.	"My dr Mr. Bnnt," rpld hs wf, "hw cn y b s trsm? Y	"t is mr thn I engg fr, I assr y."	"Mr. Bnnt, hw cn y abs yr own chldrn in sch a wy? Y	fncd hrslf nrvs. Th bsns of hr lf ws t gt hr
"D nt y wnt t knw wh hs tkn it?" crd hs wf, impntly.	mst knw tht I am thnkng of hs mrryng on of thm."	"Bt cnsdr yr dghtrs. Only thnk wht an estblshmnt it	tk	dghtrs mrrd: its slc ws vstng and nws.
"_Y_ wnt t tll m, and I hv n objctn t hrng it."	"s tht hs dsgn in sttng hr?"	wld	dlght in vxng m. Y hv n cmpssn on my pr nrvs."	
This ws invttn engh.	"Dsgn? Nnsns, hw cn y tlk s! Bt it is vry lkly tht h	b fr on of thm. Sr Wllm and Ldy Lcs ar dtrmnd t g,	"Y mstk m, my dr. I hv a hgh rspct fr yr nrvs. Thy	
		mrly on tht acctnt; fr in gnrl, y knw, thy vst n nw	ar my old frnds. I hv hrd y mntn thm wth cnsdrtn	
			ths twnty yrs at lst."	

Homework 6 Review: extra

- *WAR AND PEACE* By Leo Tolstoy/Tolstoi (1867)

- using 1922 (L&A Maude) English translation

```
$ wc war_and_peace.txt
```

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

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

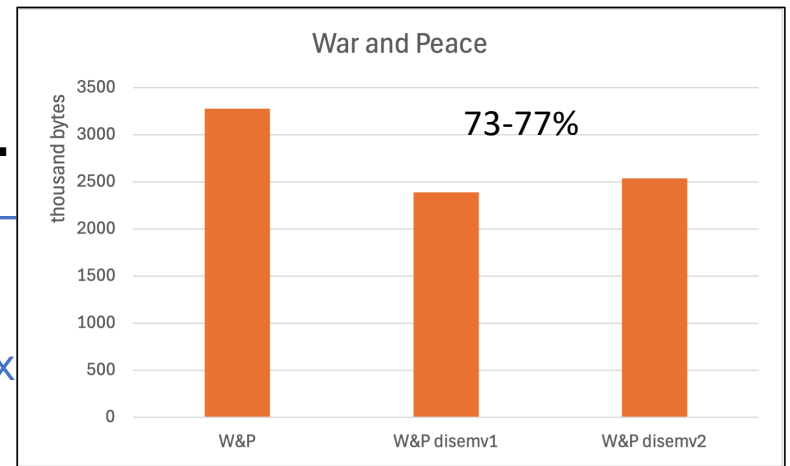
```
$ wc wr_nd_pc.txt
```

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

```
$ perl disemvowel.perl war_and_peace.txt
```

```
$ wc wr_and_pc.txt
```

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



Python behavior

Different behavior: How do we reconcile these?

python

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

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

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

```
>>> b
```

```
[-1]
```

```
>>> a
```

```
[-1]
```

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

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

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

```
>>> b
```

```
-1
```

```
>>> a
```

```
1
```

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:

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
```

Remember **bracketing** when initializing:

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

Make Rule 2

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

to replace the array name, `@array`, with the reference:

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

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

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

Careful! `$saref->[3]` and `$saref[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

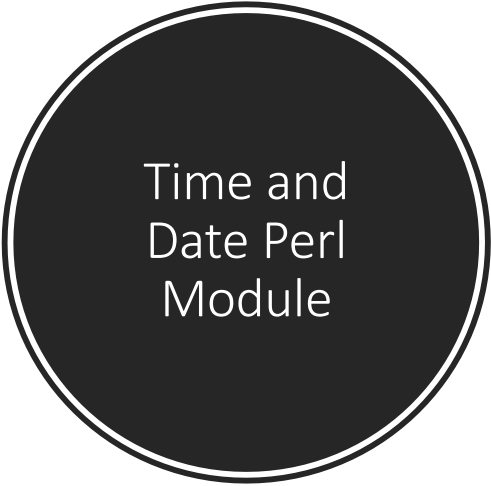
- Perl code:
 `$a = [1, 2, 3, 4, 5]; print $a+1, "\n";`
- What happens here?

```
$ 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



Time and
Date Perl
Module

```
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 22 2026  
9/22/2026 falls on a Tuesday  
9/20/2026 is today
```

CPAN

<https://www.cpan.org>



Comprehensive Perl Archive Network

OVER 20 YEARS OF PERL LIBRARIES AT YOUR FINGERTIPS

[Home](#) [Modules](#) [Ports](#) [Perl Source](#) [FAQ](#) [Mirrors](#)

Search:

Welcome to CPAN

The Comprehensive Perl Archive Network (CPAN) currently has [33,384,459 Perl modules](#) in 47,761 distributions, written by 14,747 authors.

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

Search CPAN via

- [metacpan.org](#)

Recent Uploads

- [Rex-LibSSH-0.004](#)
- [JQ-XS-2.03](#)
- [Net-LibSSH-0.004](#)
- [App-PDFLibrarian-6.3.3](#)
- [Physics-Balls-0.05](#)
- [MooX-Role-SEOTags-1.2.2](#)
- [ForgeOps-Tracker-0.7.1](#)
- [Astro-App-Satpass2-0.058_02](#)
- [Punk-0.53](#)
- [File-SOPS-0.004](#)
- [more...](#)

Getting Started

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

How to contribute

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

Perl Resources

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

macOS Date/Calc.pm

- Homebrew uses the prefix `/opt/homebrew` on Apple Silicon (Intel `/usr/local`)
 - `brew install perl cpanminus`
- Verify locations (PATH):
 - `$ which perl`
 - `/opt/homebrew/bin/perl`
 - `$ which cpanm`
 - `/opt/homebrew/bin/cpanm`
- Verify locations (PATH):
 - `~/.bash_profile` (bash)
 - `~/.zprofile` (zsh)

<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: ...)
 - `$ cpanm Date::Calc`
 - `--> 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`
 - `Building and testing Date-Calc-6.4 ... OK`
 - `Successfully installed Date-Calc-6.4`

CPAN SSL Certificate Error

```
export OPENSSL_DIR=$(brew --prefix openssl@3) export  
SSL_CERT_FILE=$(brew --prefix)/etc/ca-certificates/cert.pem export  
CURL_CA_BUNDLE=$(brew --prefix)/etc/ca-certificates/cert.pem
```

Date::Calc

meta::cpan

AboutSponsorgrep::cpan

Search results for "date::calc"

Date::Calc - Gregorian calendar date calculations 

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

[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'
23301520812602620
$ perl -le '$now = localtime; print $now'
Sun Sep 20 15:30:54 2026
$ perl -le '@now = localtime; print "@now"'
36 31 15 20 8 126 0 262 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)	=					
						1900			localtime(time);

Time and Date

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

```
$ python3
```

```
Python 3.13.1
```

```
>>> import datetime
```

```
>>> d = datetime.date(2026,9,22)
```

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

```
Tuesday
```

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

```
September 22 2026
```

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

```
datetime.datetime(2026, 9, 20, 15, 35, 41, 347150)
```

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)

Time and Date

```
>>> import locale
>>> loc = locale.setlocale(locale.LC_ALL)
>>> loc
'C/en_US.UTF-8/C/C/C/C'
>>> locale.setlocale(locale.LC_ALL, 'ja_JP.UTF-8')
'ja_JP.UTF-8'
>>> d = datetime.date(2026,9,22)
>>> print(d.strftime("%A"))
火曜日          (kayōbi)
>>> print(d.strftime("%B %-d %Y"))
9月 22 2026
```

TBH Perl looks a bit scary
perldoc.perl.org/perllocale

```
perl -e 'use POSIX
qw(locale_h); use locale; my
$new_locale = setlocale(LC_ALL,
"ja_JP.UTF-8"); ...'
```

Date::Calc			
English		==>	1
Français	(French)	==>	2
Deutsch	(German)	==>	3
Español	(Spanish)	==>	4
Português	(Portuguese)	==>	5
Nederlands	(Dutch)	==>	6
Italiano	(Italian)	==>	7
Norsk	(Norwegian)	==>	8
Svenska	(Swedish)	==>	9
Dansk	(Danish)	==>	10
suomi	(Finnish)	==>	11
Magyar	(Hungarian)	==>	12
polski	(Polish)	==>	13
Romaneste	(Romanian)	==>	14

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)

Look It Up

☒ 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
...