

To be covered today

■Control structures

► if, while/until, for, foreach, do-while

■True and false values

■Loop control

► labels, next, last, redo

■Expression modifiers

► another way of doing control structures

■Program arguments

► aARGV array

■Perl defaults

► s_ variable (standard argument)

► <> filehandle (diamond operator)

Control structures

Definition

Any programming construct that alters the flow of a program

• selection

- if/else, unless
• iteration

- while, until, for, foreach

- last, next, redo

Performing selection

unless (condition) is same as if (!condition)

if (condition) evaluated in scalar context

{

These lines executed if condition is true.
}
elsif (condition2) — zero or more elsif clauses allowed

{

These lines executed if condition2 is true.
}
else — else Clause is optional
{

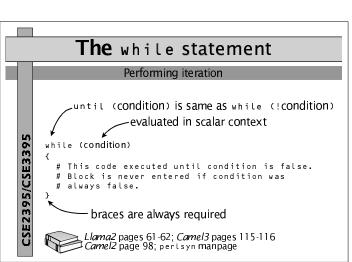
These lines executed if all
conditions are false.
}

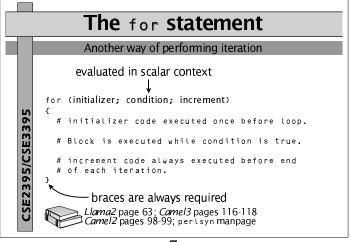
braces are always required, unlike in C

Llama2 pages 59-61; Camel3 page 114-115;
Camel2 page 98; pertsyn manpage

5

3





Boolean evaluation ■Boolean conditions evaluated in scalar context as strings ■ False values are "" (the empty string) undef (the undefined value) - converted to empty string when evaluated as string "0" (number 0 should be false) • empty arrays - in scalar context, returns number of elements (0) ■True values are ▶ everything else

CSE2395/CSE3395

Llama2 page 59; Camel3 pages 29-30 Camel2 page 46; perldata manpage

```
Example
                Printing a random line from input
     # Read in lines until EOF.
while (defined($input = <STDIN>))
        chomp $input;
CSE2395/CSE3395
        push alines, $input;
     # Get a random number of the right size.
     $pick = int(rand(scalar @lines));
     # Print line indexed by $pick.
print $lines[$pick], "\n";
```

9

The foreach statement Iterating over a list ✓ if omitted, default iterator is \$_ foreach \$var (list) - parentheses are always needed CSE2395/CSE3395 # \$var equals each element of list in turn. # list may be a literal or named array. -braces are always needed Llama2 pages 63-65; Camel3 pages 118-120 Camel2 pages 100-101; pertsyn manpage 10

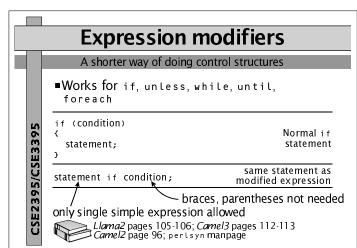
Example Adding a list of numbers # Read input until EOF (note list context). @data = <STDIN>; Iterate over every element in adata. # Use of adata could be eliminated with: CSE2395/CSE3395 # foreach $number (\langle STDIN \rangle)$ foreach \$number (@data) # This actually modifies elements of adata. chomp \$number; \$total += \$number; # Print result. print "Total is \$total\n";

11

The default argument The s variable With many functions taking an argument, naming this argument is optional Default value used in this case is special variable \$ CSE2395/CSE3395 ■print; ▶ same as print \$_; ■while (<STDIN>) ▶ same as while (defined(\$ = <STDIN>)) ► special case, only applies to filehandle in while Llama2 pages 64, 72-73; Camel3 pages 658, 682 Camel2 page 131; pertvar manpage

cat: duplicating standard input # Explicit version using \$line while (defined(\$line = <\STDIN>)) { print \$line; } # Implicit version using \$_ while (<\STDIN>) { print; } # Tiny version using expression modifier print while <\STDIN>;

13



______14

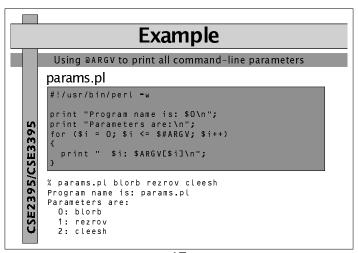
The do-while statement Performing a loop at least once unlike with a normal while expression modifier, which tests the condition first, the do-while statement always performs the loop body once before testing the condition do { # This code is always executed at least once. } while condition; until also allowed Llama2 page 62; Camel3 page 112, 701; Camel2 page 158 perlsyn, perlfunc manpages

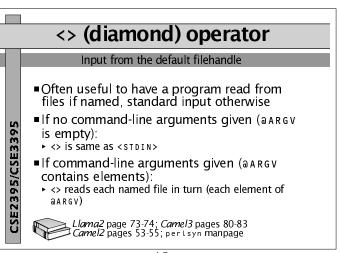
The BARGV array

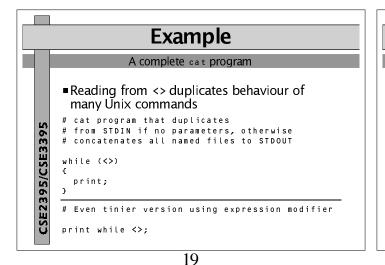
• BARGV contains command-line parameters given to script
• empty array if no parameters
• unlike C, no need for argc parameter because BARGV's size is known
• Script name is in special variable \$0
• different from C where argv[0] is program name and first parameter is in argv[1]

Llama2 page 73; Camel3** page 659
Camel2 pages 138, 136; pertvar manpage

15 16







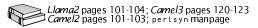
Loop control commands

Changing the way a loop is executed

- last
- exits the innermost containing loop
- ► like C's break statement

CSE2395/CSE3395

- jumps to end of innermost containing loop
- executes increment code of for loop, then tests loop
- continues from beginning of loop
- ▶ like C's continue statement



20

Loop control commands

Changing the way a loop is executed

CSE2395/CSE3395

- ▶ jumps to start of innermost containing loop
- does not test loop condition
 no equivalent in C without use of goto
- Last, next and redo can take a label
 - ▶ label attached to an enclosing block
- command now repeats/exits that loop instead of innermost containing loop
- ▶ not possible in C without use of goto
- ▶ label should be typed in all capitals

Example

Printing the header of an email message

```
# Mail messages come in two parts: the header
# and the body. The first blank line in the
      # message separates the header from the body.
     # Read each line of the email message.
CSE2395/CSE3395
     LINE: while (<>)
        # If line is blank (contains only a newline)
         # then exit the while loop.
        # Note use of expression modifier (if)
# and (in this case redundant) label LINE.
last LINE if $_ eq "\n";
         # Print header line.
        print;
```

21

22

Covered today

- ■Control structures
- ▶ if, while, for, foreach, do-while
- ■True and false values
- Loop control
- ► labels, next, last, redo
- ■Expression modifiers
- ▶ another way of doing control structures
- Program arguments
- ► a ARGV array

CSE2395/CSE3395

- Perl defaults
- \$_ variable (standard argument)

23

Going further

More things related to today's topic

- Loops, blocks and go to
- more scary things to do with control structures. including implementing C's switch
- Camel3 pages 123-127, 729-730; Camel2 pages 103-106, 178
- ■Perl special variables
- ► \$_, aARGV, \$>, %ENV, \$&, aINC: line noise with
- Camel3 pages 653-675; Camel2 pages 127-140

Next time

To be covered in Topic 5

- Hashes
- associative arraysarrays indexed by strings
- Hash variables and functions



CSE2395/CSE3395

Reading: Learning Perl 2nd edition chapter 5, pages 66-71 Programming Perl 3rd edition pages 76-78 Programming Perl 2nd edition pages 50-51 perl data manpage