# IT 240
# Shell Scripting for Administrators

## Chapter 2
## Scalar Data

Stan J. Senesy
IT Program/CCS
New Jersey Institute of Technology

# Scalar Data

- Perl treats all numbers internally as double-precision floating point values
- Distinct integer operations do not exist
- Literals are number values that are directly written into source; they are not calculated:
  - Floating point - 1.25
  - Integer - 2008

# Scalar Data

- Literals may use base values other than decimal:
  - Octal (leading 0) - 0377
  - Hex (leading 0x) - 0xff
  - Binary (leading 0b) - 0b10100011O
- You may use the underscore character to make long values easier to read

# Scalar Data

- Perl supports normal arithmetic operations:
  - Addition +
  - Subtraction -
  - Multiplication *
  - Division / (both integer and real)
  - Modulo %

# Scalar Data

- Strings are sequences of characters
- Single quotes may be used to identify the beginning and end of a string
- The backslash character may be used as an escape to a quote or additional backslash to be used in the string

# Scalar Data

- Double-quoted strings differ from their single quoted counterparts
- They allow the backslash to be used in conjunction with a control character (backslash escape)
- Table 2-1

# Scalar Data

- Perl supports a number of string operations:
  - Concatenation - dot operator
    - “string1” . “string2”
  - Repetion
    - “string” x 3

# Scalar Data

- Perl converts between numbers and strings as necessary
- When an operand requires a number, perl converts the string to it's proper numeric value
- If a string value is needed, the opposite conversion is performed

# Scalar Data

- There are a number of ways to tell perl to warn you if there are potential problems with your code:
  - *$perl -w my_program*
  - *#!/usr/bin/perl -w*
  - *#!/usr/bin/perl*
  - *use warnings;*

# Scalar Data

- Perl also supports a more detailed inspection of code:
  - *#!/usr/bin/perl*
  - *use diagnostics;*
  - *$perl -Mdiagnostics ./my_program*

# Scalar Data

- Variables hold one or more values
- Variable names begin with the $, and are followed by a letter or underscore and more digits/letters/underscores
- Scalar variables are always referenced with the leading $
- Names are case sensitive and should be descriptive

# Scalar Data

- The equal sign is used to assign values to a variable:
  - $fred = 17; #17
  - $barney = ‘hello’; #hello
  - $barney = $fred + 3; #20
  - $barney = $barney * 2; #40
  - $barney *= 2 #80

# Scalar Data

- Print may be used to output to the standard output (usually the terminal)
  - *print “Hello world \n”*
  - *print 2 * 2*
  - *print “The answer is “, 2 * 2, “.\n”;*

# Scalar Data

- Double quoted strings are subject to variable interpolation:
  - *$meal = “brontosaurus steak”;*
  - *$barney = “fred ate a $meal”;*
  - *$barney = ‘fred ate a ‘ . $meal;*
- Be careful of name evaluation:
  - *$what = “brontosaurus steak”;*
  - *$n = 3;*
  - *print “fred ate $n $whats.\n”;*

# Scalar Data

- Perl supports the standard rules for operator precedence as shown in table 2-2
- Precedence may be overridden by using the parentheses
- Perl also supports logical comparison operators that return a true/false value when comparing numbers
- Table 2-3

# Scalar Data

- The if/else structure is available in perl and follows the form:

```
if ( condition ) {
    statement(s)
} else {
    statement(s)
}
```

# Scalar Data

- Any Boolean test may be used for the conditional portion of the if statement
- In perl a numeric zero means false, everything else is true
- An empty strign is false, all others mean true
- The unary NOT operator (!) may be used to invert a truth character

# Scalar Data

- Perl typically reads data from the standard input (keyboard), also called <STDIN>
- When you use the keyword in place of a scalar value, perl reads the next complete line from the keyboard
- When reading a string, perl will place the return at the end of the string
- Use the chomp operator to remove the newline at the end of a string if desired

- Perl also supports the while control structure:

```
while (exit condition test) {
   statement(s)
}
```

# Scalar Data

- Perl will allow the use of a scalar variable before it is given a value:
  - If numeric, it will be assigned zero
  - If string, it will act as an empty string
  - This behavior is called undef and may be over-ridden by the programmer