Logging into the CSIF

• You must use your University login name (what you type to the Central Authentication System)
• Try it from Eduroam; if that doesn’t work, get the Library VPN and use that
  • See the web page https://library.ucdavis.edu/vpn/ for how to do this
• Here is the command:
  • ssh your-cas-name@pcnn.cs.ucdavis.edu

where nn is a number between 01 and 43.
• To find the status of systems, look here:
  • http://iceman.cs.ucdavis.edu/nagios3/cgi-bin/status.cgi?hostgroup=all
Variable Names

• Composed of letters, digits, and underscore ("_")
  • amountOfMoney, amount_of_money, hello2, _xyzzy (all valid)
    • Note: avoid leading underscores, as many library functions and header files use names beginning with underscores
  • amount-Of-Money, amount/Of/Money, $amount (none valid)

• Cannot begin with a digit
  • go3work (valid), 2go2work (not valid)

• Capital and lower-case are different!
  • Amount, amount are 2 separate variables
Basic Types

• Integers
  • short, int, long
    • Really, short int and long int but the int is usually omitted
    • Guaranteed that the number of bits in each is short \leq int \leq long
  • signed, unsigned (signed can be omitted)
    • On an $n$-bit system, signed typically goes from $-2^{n-1}$ to $2^{n-1}-1$; for unsigned, 0 to $2^n-1$
    • Example: on a 64 bit system signed integers are in $[-2^{63}, 2^{63}-1]$; unsigned, $[0, 2^{64}-1]$

• Characters
  • char
    • Holds a character
    • Treated exactly as a very short integer
Basic Types

• Floating point
  • `float`, `double`
    • `double` can hold bigger numbers than `float`

• Examples
  • 34, –12, 0, 9999 are `ints`
  • 9836592047L is `long`
  • ‘5’, ‘X’, ‘\t’, are `chars`
  • 34.2, 99e-12 are `floats`
  • 3e50 is `double`
Type Casting

• To convert from one type to another, put the target type in parentheses

• Examples (all run on a 32-bit system)
  • (float) 3 is 3.0
  • (int) 3.25 is 3
  • (int) 3.9 is 3 (note truncation, not rounding)
  • (signed) –53.7 is –53
  • (unsigned) –53.7 is 4294967243 (= $2^{32} - 53$)
Arithmetic

• Addition: +
  • if operands are same type, type of result is type of operands
  • if operands are of different types, type of result is:
    • float + int gives float; int + double gives double; float + double gives double

• Subtraction: –
  • type of result follows same rules as +

• Multiplication: *
  • type of result follows same rules as +

• If results too large, result is truncated to maximum length of system (overflow)
Arithmetic

• Division: \\
  • type of result follows same rules as +
  • if result is too small, it will be treated as 0 (underflow)
  • division by 0 causes error (program crashes)

• Remainder, modulus: %
  • dividend is non-negative integer; divisor is positive integer
  • with anything else, the results may not be what you expect!

• Actual definition of remainder:
  • \( n \% p = r \) implies that \( n = ap + r \) for some integer \( a \)
  • So \( 5 \% -2 \) can be \( 5 = (-2) \times (-2) + 1 \) or \( 5 = (-3) \times (-2) + (-1) \)
Precedence and Associativity

• *, /, % have highest precedence, associate from left
  • $8 \times 5 / 4 = 10$, not 8

• +, – come next, also associate from left
  • $8 \times 5 + 3 = 43$, not 64

• Parentheses change order of evaluation
  • $8 \times (5 + 3) = 64$, not 40