Logical Data Type

- A special type of data that
  - can have one of only two possible values
    - true (displayed as 1) and false (displayed as 0)

- These values are produced by
  - special functions: true and false
    - x = true
      x =
      1
    - y = false
      y =
      0
  - relational and logical operators

Logical Data Type

- Logical and numerical values can be used in place of another
  - If a logical value is used in a place where a numerical value is expected
    - true values are converted to 1
    - false values are converted to 0
      - a = true;
        b = a *3
        b =
            3
      - x = false;
        y/x
        ans =
            Inf

Logical Data Type

- Logical and numerical values can be used in place of another
  - If a numerical value is used in a place where a logical value is expected
    - non-zero values are converted to true
    - zero values are converted to false
      - a = -5;
        a & true
        ans =
            1
      - x = false;
        a = -5;
        y = (a > 0) | x
        y =
            0

Relational Operators

- Relational operators are used to represent conditions
  - such as "space ≤ 0" and "result ≠ 25"
- They take two numerical (or string) operands
- They yield a logical result (true or false)

Relational Operators

- The general form is: *a1* op *a2*
  - *a1* and *a2* are arithmetic expressions, variables, or strings.
  - op is one of the following

Operator	Operation
==	Equal to
~=	Not equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to

Relational Operators

Operation	Result
3 < 4	true (1)
3 <= 4	true (1)
3 == 4	false (0)
3 ~= 4	true (1)
3 > 4	false (0)
4 >= 4	true (1)
'A' < 'B'	true (1)

Characters are evaluated in alphabetical order

Relational Operators

- Relational operators may be used
  - To compare a scalar value with an array
    - a = [1 10; -2 3];
      b = 3;
      c = (a >= b)
      c =
          0   1
          0   1
  - To compare two arrays
    - x = [1 10; -2 3];
      y = [4 5; 6 7];
      x < y
      ans =
          1   0
          1   1
    - x = [1 10; -2 3];
      y = [4 5; 6 7; 2 0];
      x < y
      ??? Error using ==> <
      Matrix dimensions must agree.

Relational Operators

- Don’t confuse equivalence (==) with assignment (=)
- Relational operations have lower priority than arithmetic operations
  - i.e., relational operators are evaluated after all arithmetic operators have been evaluated
  - You can use parentheses to be safe
    - 14 + 4 < 3 * 5

      ans =

      0
    - (14 + 4) < (3 * 5)

      ans =

      0

Relational Operators

- Be careful about roundoff errors during numeric comparisons (== and ~= operators)
  - a = 0;
    b = sin(pi);
    a == b
    ans =
    0 (since sin(pi) calculation yields 1.2246e-016)
- You may use "abs(a-b) < small number" instead of "a == b"
  - abs(a-b) < 1.0E-14
    ans =
    1
  - abs(a-b) < eps
    ans =
    1

Logical Operators

- More complex conditions can be represented
  - by combining relational operations using logical operators
    - “temperature ≠ 25” AND “humidity < 60 %”
    - “exam grade < 45” OR “attendance ≤ 75”
- They take one or two logical operands
- They yield a logical result (true or false)

Logical Operators

- The general form of a binary logic operation: *b1* op *b2*
- The general form of a unary logic operation: op *b1*
  - *b1* and *b2* are expressions or variables
  - op is one of the following

Operator	Operation
&	Logical AND
&&	Logical AND with shortcut evaluation
|	Logical OR
||	Logical OR with shortcut evaluation
xor	Logical exclusive OR
~	Logical NOT

Logical ANDs

b1	b2	b1 & b2	b1 && b2
0	0	0	0
0	1	0	0
1	0	0	0
1	1	1	1

- Differences between & and && operators
  - && only works between scalar values, while & works with either scalar and array values (arrays should be compatible)
  - && evaluates first *b1* and then *b2* only if *b1* is true, otherwise it returns false immediately without evaluating *b2,* while & operator evaluates both *b1* and *b2* before returning an answer

Logical ANDs

- Most of the time, it does not matter which AND operation is used.
  - b = 4; a = 8;
  - (b ~= 6) & (a > 4)
    ans =
    1
  - (b ~= 6) && (a > 4)
    ans =
    1
- If the comparison is between arrays, we have to use & operator
  - x = [true false; false true]; y = [false false; true true];
  - x & y
    ans =
    0 0
    0 1
  - x && y
    ??? Operands to the || and && operators must be convertible to logical scalar values.
- Sometimes it is important to use shortcut expressions
  - b = 0; a = 4;
  - (b ~= 0) & (a/b > 10)
    Warning: Divide by zero.
  - (b ~= 0) && (a/b > 10)
    ans =
    0

Logical ORs

b1	b2	b1 | b2	b1 || b2
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	1

- Differences between | and || operators
  - || only works between scalar values, while | works with either scalar and array values (arrays should be compatible)
  - || evaluates first *b1* and then *b2* only if *b1* is false, otherwise it returns true immediately without evaluating *b2,* while | operator evaluates both *b1* and *b2* before returning an answer

Logical XOR and NOT

b1	b2	xor(b1, b2)	~b1
0	0	0	1
0	1	1	1
1	0	1	0
1	1	0	0

Hierarchy of Operators

1) Parenthesis (starting from the innermost)
2) Exponentials (left to right)
3) Multiplications and divisions (left to right)
4) Additions and subtractions (left to right)
5) Relational operators (==, ~=, >, >=, <, <=) (left to right)
6) ~ operators
7) & and && operators (left to right)
8) |, ||, and xor operators (left to right)

Logical Functions

- Matlab includes a number of logical functions, which can be used with relational and logical operators

Function	**Purpose**
ischar(a)	Returns true if *a* is a character array
isempty(a)	Returns true if *a* is an empty array
isinf(a)	Returns true if the value of *a* is *Inf* (infinite)
isnan(a)	Returns true if the value of *a* is *NaN* (not a number)
isnumeric(a)	Returns true if *a* is a numeric array

Examples

- To count the occurrence of digits in a sentence (represented by array B)
  - (B(i) >= '0') & (B(i) <= '9')
- To count the occurrence of letters in a sentence (represented by array B)
  - ((B(i) >= 'a') & (B(i) <= 'z')) | …
    ((B(i) >= 'A') & (B(i) <= 'Z'))

Examples

- To determine if a quadratic equation has two distinct real roots
  - (b^2 – 4*a*c) > 0
- To determine if the specified year is a leap year
  - mod(year,4) == 0 & …
    (mod(year,100) ~= 0 | …
    mod(year,400) == 0)