

Relational and Logical Operations

Contents

- ① Another Way to Display Text
- ② Relational and Logical Operators
- ③ Examples
- ④ Exercises

Another Way to Display Text

FPRINTF: Alternate Displaying Function

Combine literal text with numeric data.

- Number of digits to display

```
fprintf('There are %d days in a year.\n', 365)
```

- Complex number

```
z = exp(1i*pi/4);  
fprintf('%f+%fi\n', real(z), imag(z));
```

FPRINTF: Formatting Operator

```
%[field width][precision][conversion character]
```

e.g. %12.5f.

- %: marks the beginning of a formatting operator
- [field width]: maximum number of characters to print; optional
- [precision] number of digits to the right of the decimal point; optional
- [conversion character]

| | |
|----|------------------------------|
| %d | integer |
| %f | fixed-point notation |
| %e | exponential notation |
| %g | the more compact of %f or %e |
| %s | string array |
| %x | hexadecimal |

Relational and Logical Operators

Relational Operators

How are two numbers X and Y related?

- $[X > Y]$ Is X greater than Y?
- $[X < Y]$ Is X less than Y?
- $[X \geq Y]$ Is X greater than or equal to Y?
- $[X \leq Y]$ Is X less than or equal to Y?
- $[X == Y]$ Is X equal to Y?
- $[X \neq Y]$ Is X not equal to Y?

The symbols used between X and Y are called the **relational operators**.

Logical Variables and Logical Operators

- A relational statement evaluates to either **True(1)** or **False(0)**; these are called **logical variables** or **boolean variables**.
- As arithmetic operators (+, -, *, /) put together two numbers and produce other numbers, **logical operators** combine two logical variables to produce other logical variables.
- **Logical Operators:** *and, or, not, xor*

Logical Operator: $\&\&$ (AND)

Let A and B be two logical variables. The $\&\&$ operation is completely defined by the following truth table:

| A | B | $A \ \&\& \ B$ |
|-----|-----|----------------|
| F | F | F |
| F | T | F |
| T | F | F |
| T | T | T |

Note that $A \ \&\& \ B$ is true if and only if both A and B are true.

Logical Operator: \vee (OR)

Let A and B be two logical variables. The \vee operation is completely defined by the following truth table:

| A | B | $A \vee B$ |
|-----|-----|------------|
| F | F | F |
| F | T | T |
| T | F | T |
| T | T | T |

Note that $A \vee B$ is false if and only if both A and B are false.

Logical Operator: `xor` (exclusive or)

This is a special variant of the `||` operator.

| A | B | <code>xor (A, B)</code> |
|---|---|-------------------------|
| F | F | F |
| F | T | T |
| T | F | T |
| T | T | F |

Note that `xor (A, B)` is true if only one of A or B is true.

Logical Operator: \sim (NOT)

This is a negation operator.

| A | $\sim A$ |
|---|----------|
| F | T |
| T | F |

Combination of Logical Operations

Let A and B be logical variables. Then $\sim (A \ \&\& \ B)$ and $\sim A \ || \ \sim B$ are equivalent:

| A | B | $A \ \&\& \ B$ | $\sim (A \ \&\& \ B)$ |
|-----|-----|----------------|-----------------------|
| F | F | | |
| F | T | | |
| T | F | | |
| T | T | | |

| A | B | $\sim A$ | $\sim B$ | $\sim A \ \ \sim B$ |
|-----|-----|----------|----------|------------------------|
| F | F | | | |
| F | T | | | |
| T | F | | | |
| T | T | | | |

Examples

Quadratics Revisited

Consider a monic quadratic function $q(x) = x^2 + bx + c$ on a close interval $[L, R]$.

- Critical point: $x_c = -b/2$
- If $x_c \in (L, R)$, $q(x)$ attains the (global) minimum at x_c ; otherwise, the minimum occurs at one of the endpoints $x = L$ or $x = R$.

Question

Write a program which determines whether the critical point of $q(x)$ falls on the interval.

Initialization

```
b = input('Enter b: ');  
c = input('Enter c: ');  
L = input('Enter L: ');  
R = input('Enter R (L<R): ');  
clc  
fprintf('Function: x^2 + bx + c, b = %5.2f, c = %5.2f\n', b, c)  
fprintf('Interval: [L, R], L = %5.2f, R = %5.2f\n', L, R)  
xc = -b/2;
```


Main Fragment

```
if L < xc && xc < R
    fprintf('Interior critical point at x_c = %5.2f\n', xc)
else
    disp('Either xc <= L or xc >= R')
end
```

Main Fragment – another way

```
if xc <= L || xc >= R
    disp('Either xc <= L or xc >= R')
else
    fprintf('Interior critical point at x_c = %5.2f\n', xc)
end
```

Main Fragment – yet another way

```
if ~(xc <= L || xc >= R)
    fprintf('Interior critical point at x_c = %5.2f\n', xc)
else
    disp('Either xc <= L or xc >= R')
end
```

The simplest `if` statement?

So far, we have seen

- `if-else` statement
- `if-elseif-else` statement

The simplest `if` statement is of the form

```
if [condition]
  [statements to run]
end
```

Input Errors

If a user mistakenly provides L that is larger than R , fix it silently by swapping L and R .

```
if L > R
    tmp = L;
    L = R;
    R = tmp;
end
```

I will show you how to send an error message and halt a program later.

Exercises

Exercise 1: Simple Minimization Problem

Question

Write a program that finds $x_{\min} \in [L, R]$ at which $q(x)$ is minimized and the minimum value $q(x_{\min})$.

- This can be done with `if-elseif-else`

Exercise 2: Leap Year

Question

Write a script which determines whether a given year is a leap year or not. A year is a leap year if

- it is a multiple of 4;
- it is not a multiple of 100;
- it is a multiple of 400.

Useful: `mod` function.

Pseudocode

```
if [YEAR] is not divisible by 4
    it is a common year
elseif [YEAR] is not divisible by 100
    it is a leap year
elseif [YEAR] is not divisible by 400
    it is a common year
else
    it is a leap year
end
```

Exercise 3: Angle Finder

Question

Let x and y be given, not both zero. Determine the angle $\theta \in (-\pi, \pi]$ between the positive x -axis and the line segment connecting the origin to (x, y) .

Four quadrants:

- 1st or 4th ($x \geq 0$): $\theta = \tan^{-1}(y/x)$
- 2nd ($x < 0, y \geq 0$): $\theta = \tan^{-1}(y/x) + \pi$
- 3rd ($x < 0, y < 0$): $\theta = \tan^{-1}(y/x) - \pi$

Useful: `atan` (inverse tangent function)

Extended Inverse Tangent

```
if x > 0
    theta = atan(y/x)
elseif y >= 0
    theta = atan(y/x) + pi
else
    theta = atan(y/x) - pi
end
```

- MATLAB provides a function that exactly does this: `atan2(x, y)`.
- **Further Exploration:** What would you do if you are asked to find the angle $\theta \in [0, 2\pi)$, with `atan` alone or with `atan2`?