Top-down Program Design, Relational and Logical Operators

Selim Aksoy
Bilkent University
Department of Computer Engineering
saksoy@cs.bilkent.edu.tr

Creating MATLAB Scripts

- Choose File>New>M-file from the menu
- Use the editor to write your program
- Document your program using comments that include
  - Short note about what your program does
  - Short note about how it works
  - Author information
  - Date information
  - Version information

Creating MATLAB Scripts

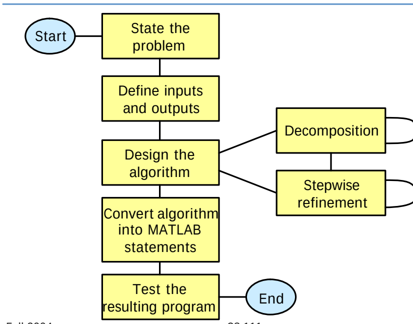
```
%  Script file: temp_conversion.m
%
%  Purpose:
%    To convert an input temperature from degrees Fahrenheit to
%    an output temperature in kelvins.
%
%  Record of revisions:
%      Date       Programmer          Description of change
%      ====       ==========          =====================
%    12/01/97    S. J. Chapman        Original code
%
% Define variables:
%   temp_f    -- Temperature in degrees Fahrenheit
%   temp_k    -- Temperature in kelvins

% Prompt the user for the input temperature.
temp_f = input('Enter the temperature in degrees Fahrenheit: ');

% Convert to kelvins.
temp_k = (5/9) * (temp_f - 32) + 273.15;

% Write out the result.
fprintf('%6.2f degrees Fahrenheit = %6.2f kelvins.\n', ...
        temp_f,temp_k);
```


Top-down Program Design



Pseudocode

- A hybrid mixture of MATLAB and English for defining algorithms
- Independent of any programming language so it can be easily converted to any programming language
- Example pseudocode:

  Prompt user to enter temperature in degrees Fahrenheit
  Read temperature in degrees Fahrenheit (temp_f)
  temp_k (in Kelvins) ← (5/9) * (temp_f – 32) + 273.15
  Write temperature in degree Kelvins

Top-down Program Design

- Problem: write a program that takes the radius and height (in meters) of a cylinder tank and the amount of water (in m^3) from the user and output the amount of extra space (in m^3) in the tank.
- Input:
  - radius and height
  - amount of water
- Output:
  - extra space

Top-down Program Design

- Design:
  1. Get radius of the tank base from the user
  2. Get the height of the tank from the user
  3. Get the amount of water
  4. Calculate the amount of extra space
  5. Write the result
- Step 4 is not clear enough, refine it:
  - Calculate the capacity of the tank (pi * radius^2 * h)
  - extra space ← capacity - water

Top-down Program Design

- Code:

```
r = input('Enter the radius of the tank base:');
h = input('Enter the height of the tank:');
water = input('Enter the amount of water:');
capacity = pi * r^2 * h;
space = capacity - water;
fprintf('There is %f m3 extra space in the
tank', space);
```


Top-down Program Design

- Testing:

  Enter the radius of the tank base:2
  Enter the height of the tank:5
  Enter the amount of water:10
  There is 52.831853 m3 extra space in the tank
- Continue testing:

  Enter the radius of the tank base:2
  Enter the height of the tank:5
  Enter the amount of water:100
  There is -37.168147 m3 extra space in the tank

Top-down Program Design

- Design: refine step 4 again
  - Calculate the capacity of the tank (pi * radius^2 * h)
  - extra space ← ((capacity – water) + abs(capacity – water))/2

Relational Operators

- Relational operators are used to represent conditions (such as “space ≤ 0” in the water tank example)
- Result of the condition is either true or false
- In MATLAB:
  - false is represented by 0
  - true is represented by 1 (non-zero)

Relational Operators

Operation	Result
3 < 4	1
3 <= 4	1
3 == 4	0
3 ~= 4	1
3 > 4	0
4 >= 4	1
‘A’ < ‘B’	1

Relational Operators

- Don't confuse equivalance (==) with assignment (=)
- Be careful about roundoff errors during numeric comparisons (you can represent "x == y" as "abs(x-y) < eps")
- Relational operations have lower priority than arithmetic operations (use parentheses to be safe, though)

Logical Operators

- More complex conditions can be represented by combining relational operations using logic operators
- Logical operators:

&	AND
|	OR
xor	Exclusive OR
~	NOT

Logical Operators

input		and	or	xor	not
a	**b**	**a & b**	**a | b**	**xor(a,b)**	**~a**
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

Operator Hierarchy

- Processing order of operations:
  - parenthesis (starting from the innermost)
  - exponentials (left to right)
  - multiplications and divisions (left to right)
  - additions and subtractions (left to right)
  - relational operators (left to right)
  - ~ operators
  - & operators (left to right)
  - | operators (left to right)