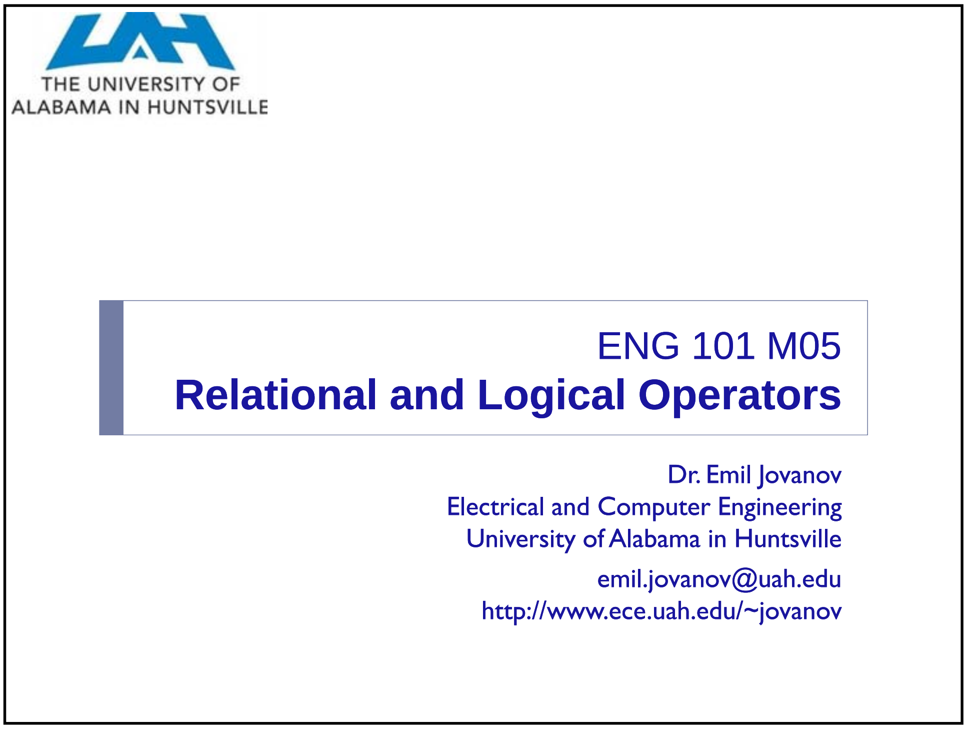
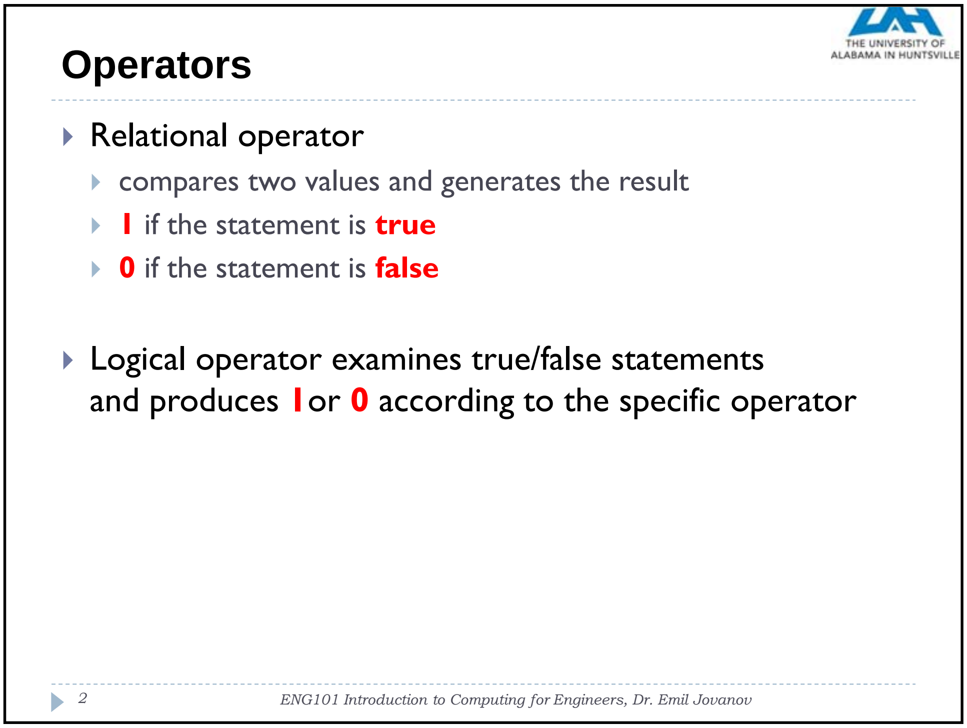


THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

ENG 101 M05
Relational and Logical Operators

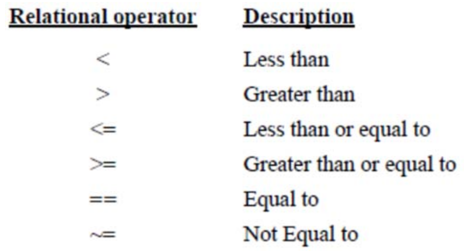
Dr. Emil Jovanov
Electrical and Computer Engineering
University of Alabama in Huntsville
emil.jovanov@uah.edu
http://www.ece.uah.edu/~jovanov



Operators

- Relational operator
  - compares two values and generates the result
  - **1** if the statement is **true**
  - **0** if the statement is **false**
- Logical operator examines true/false statements and produces **1** or **0** according to the specific operator

Relational operator:

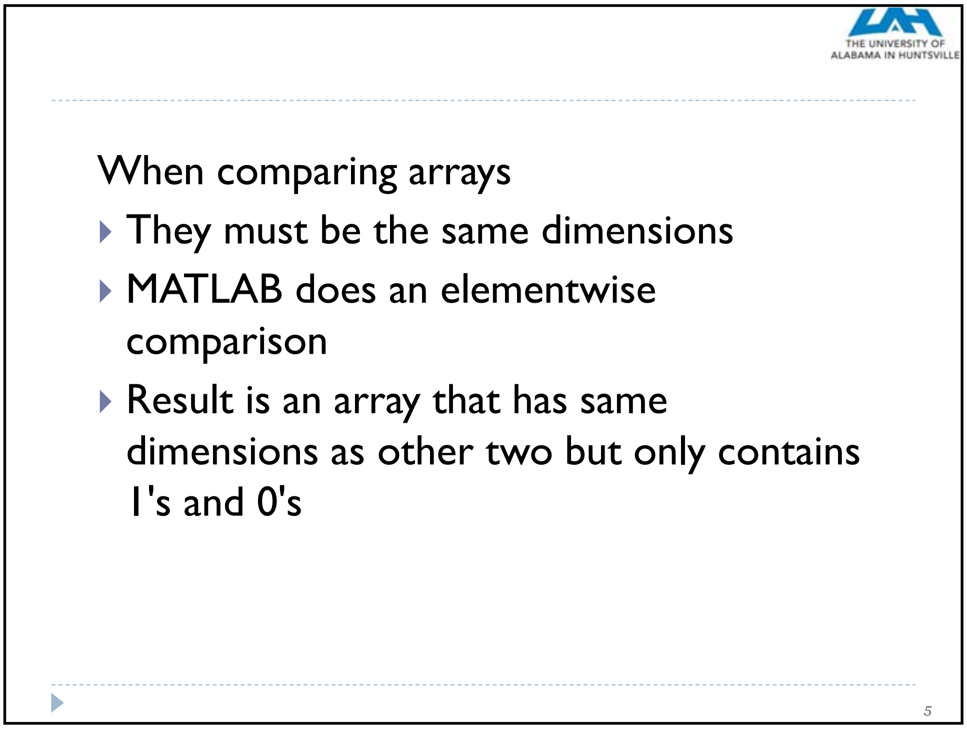


Relational operator	Description
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Equal to
~=	Not Equal to

- Can't put space between operators that have two characters
- "Not equal to" is "~=", not "!=" as in C or C++
- "Equal to" comparison is two equal signs (==), not one.
  - Remember, "=" means "assign to" or "put into"

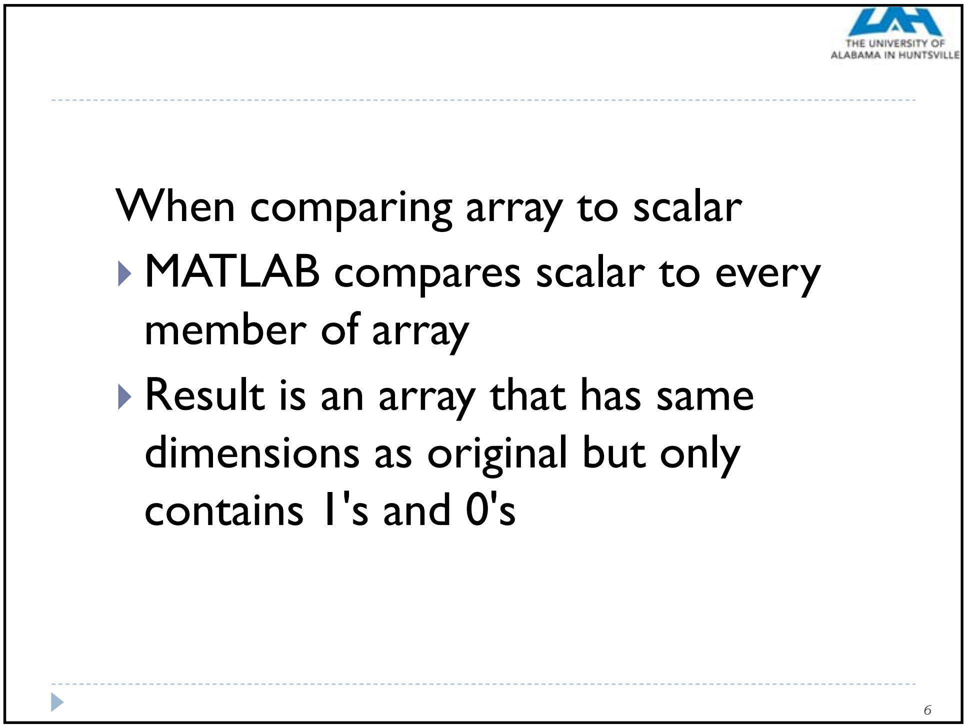
- Result of comparing with a relational operator is always "true" or "false"
  - If "true", MATLAB gives the comparison a value of one (1)
  - If "false", MATLAB gives the comparison a value of zero (0)

This may be different than convention in other programming languages. For example, C gives an expression that is false a value of zero, but it can give a true expression any value but zero, which you can't assume will be one



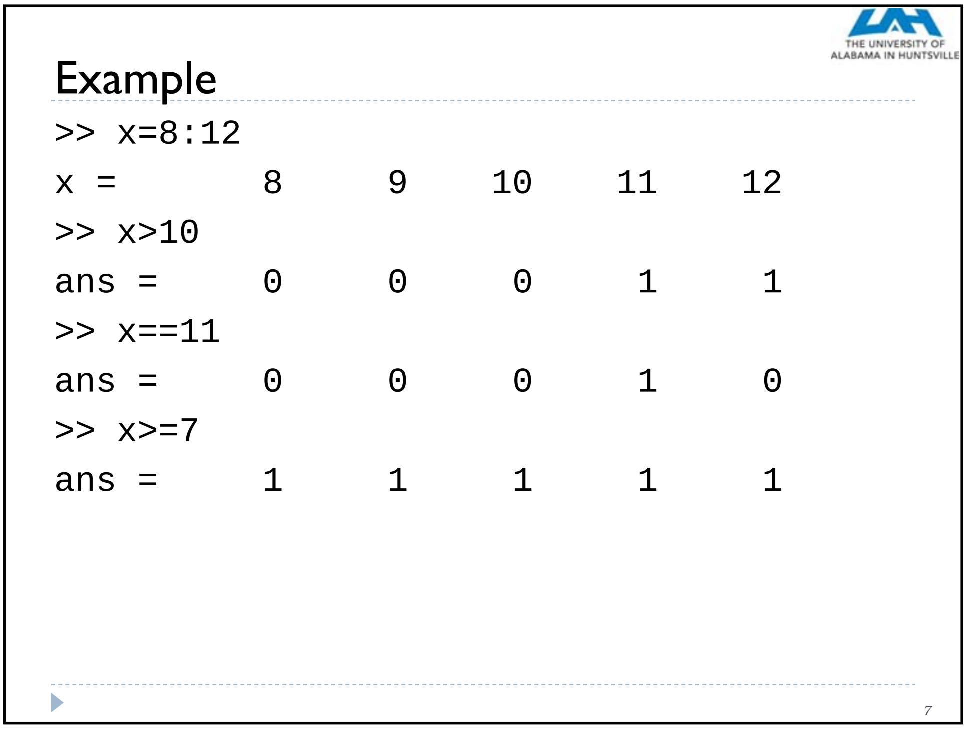
When comparing arrays

- They must be the same dimensions
- MATLAB does an elementwise comparison
- Result is an array that has same dimensions as other two but only contains 1's and 0's



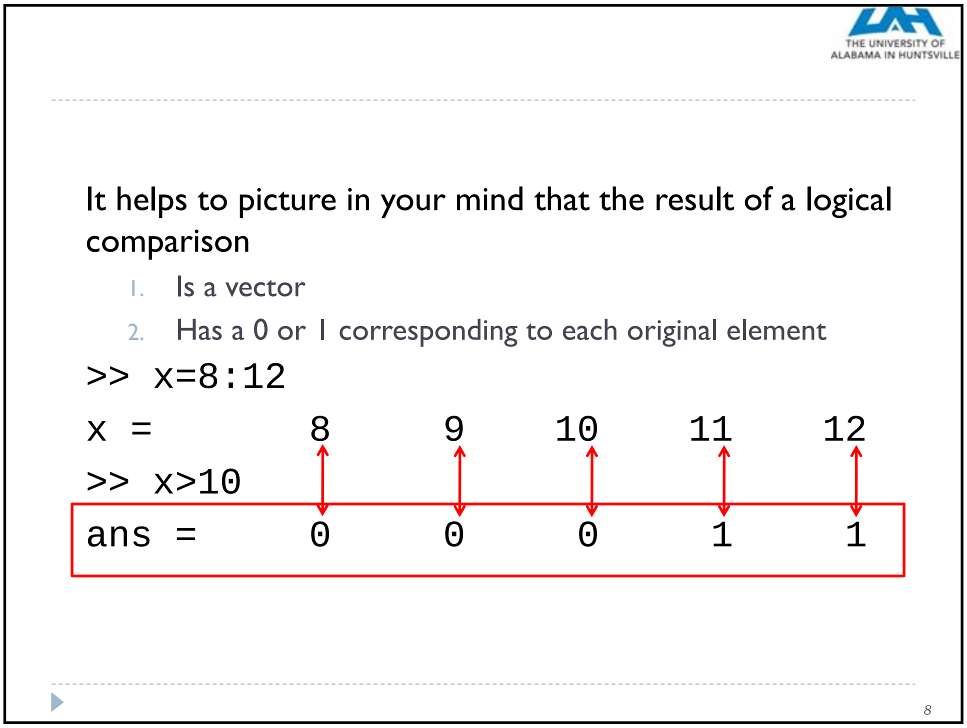
When comparing array to scalar

- MATLAB compares scalar to every member of array
- Result is an array that has same dimensions as original but only contains 1's and 0's



Example

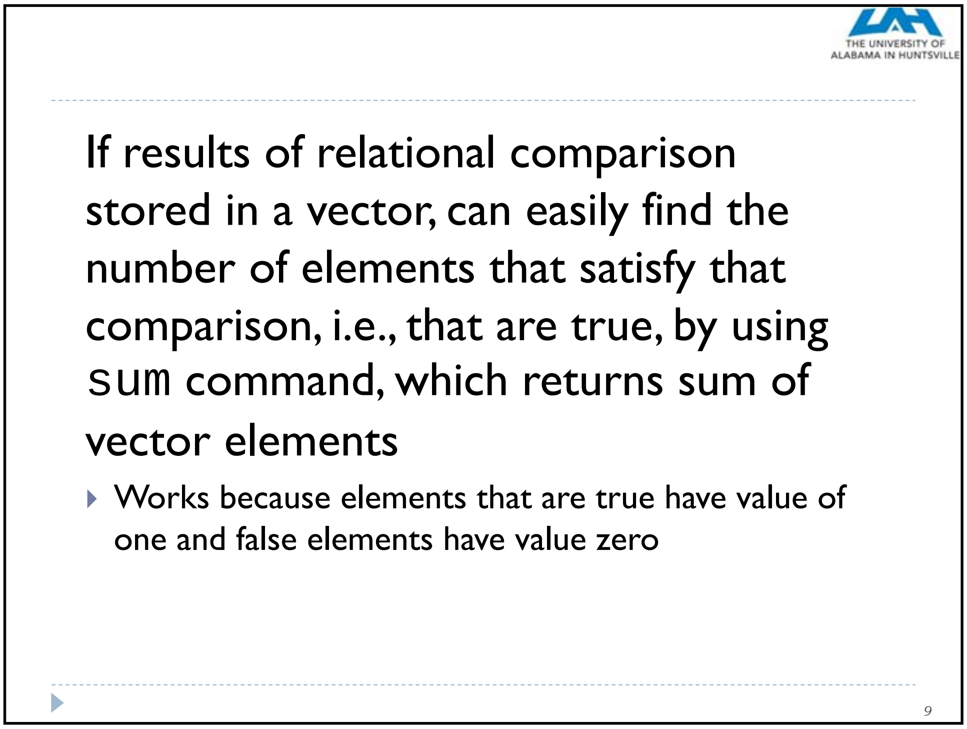
```
>> x=8:12
x =         8     9     10     11     12
>> x>10
ans =       0     0      0      1      1
>> x==11
ans =       0     0      0      1      0
>> x>=7
ans =       1     1      1      1      1
```



It helps to picture in your mind that the result of a logical comparison

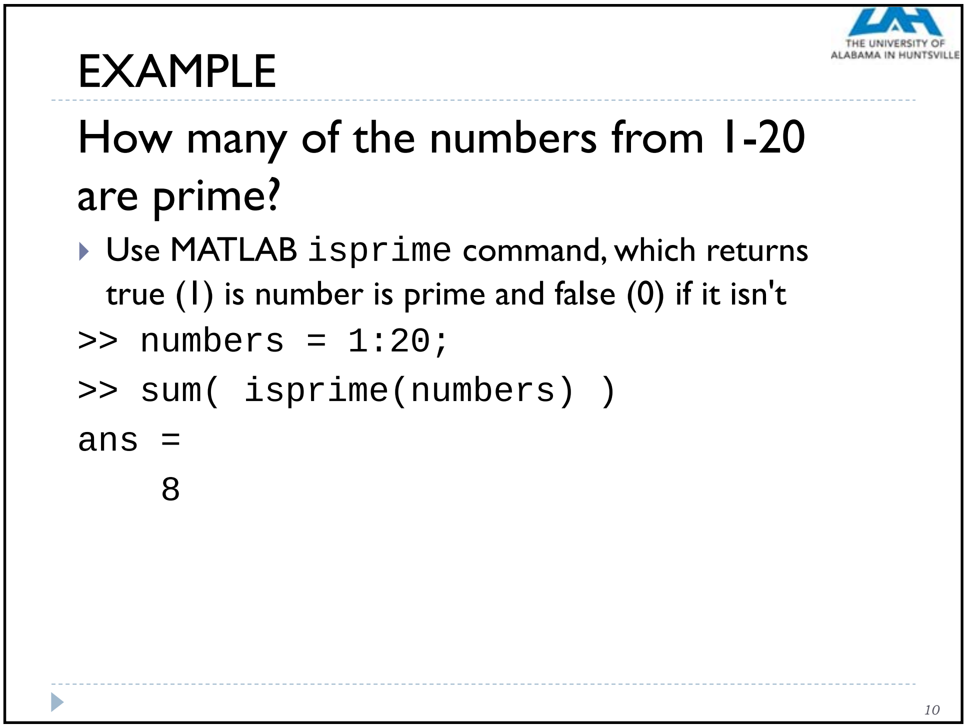
1. Is a vector
2. Has a 0 or 1 corresponding to each original element

```
>> x=8:12
x =         8     9     10     11     12
>> x>10
ans =       0     0      0      1      1
```



If results of relational comparison stored in a vector, can easily find the number of elements that satisfy that comparison, i.e., that are true, by using `sum` command, which returns sum of vector elements

- Works because elements that are true have value of one and false elements have value zero

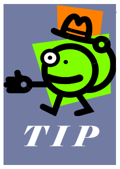


EXAMPLE

How many of the numbers from 1-20 are prime?

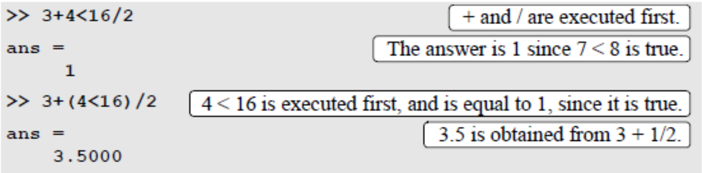
- Use MATLAB `isprime` command, which returns true (1) is number is prime and false (0) if it isn't

```
>> numbers = 1:20;
>> sum( isprime(numbers) )
ans =
    8
```



Can mix relational and arithmetic operations in one expression

- Arithmetic operations follow usual precedence and always have higher precedence than relational operations
- Relational operations all have equal precedence and evaluated left to right



```
>> 3+4<16/2
ans =
     1
>> 3+(4<16)/2
ans =
    3.5000
```

+ and / are executed first.

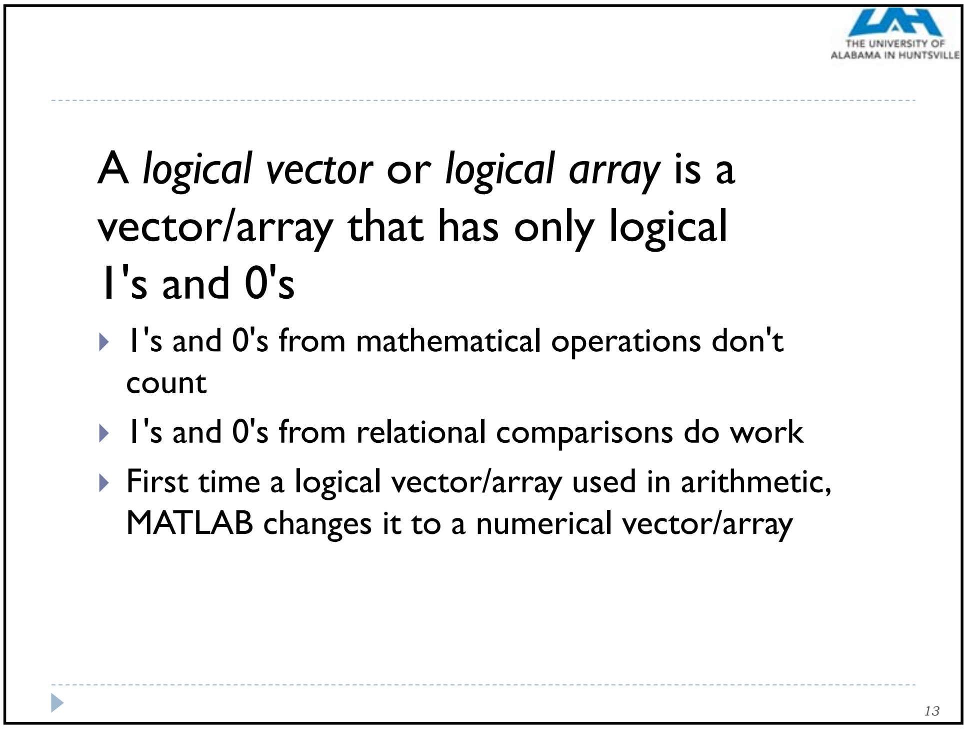
The answer is 1 since 7 < 8 is true.

4 < 16 is executed first, and is equal to 1, since it is true.

3.5 is obtained from 3 + 1/2.

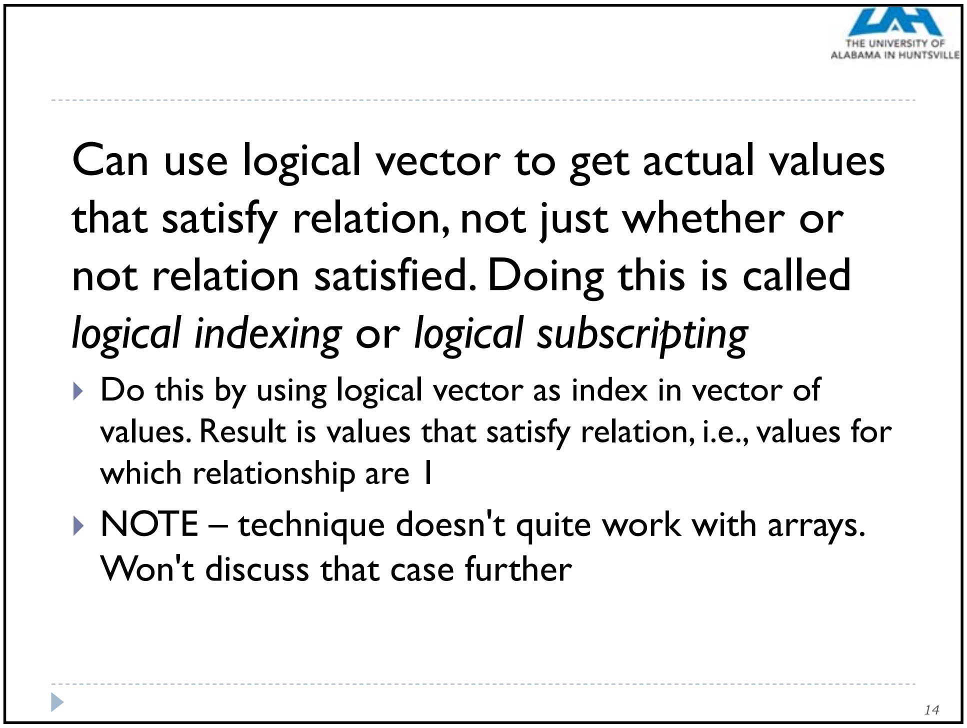
Pay attention to the order of operations

y=(6<10) + (7>8) + (5*3 == 60/4)



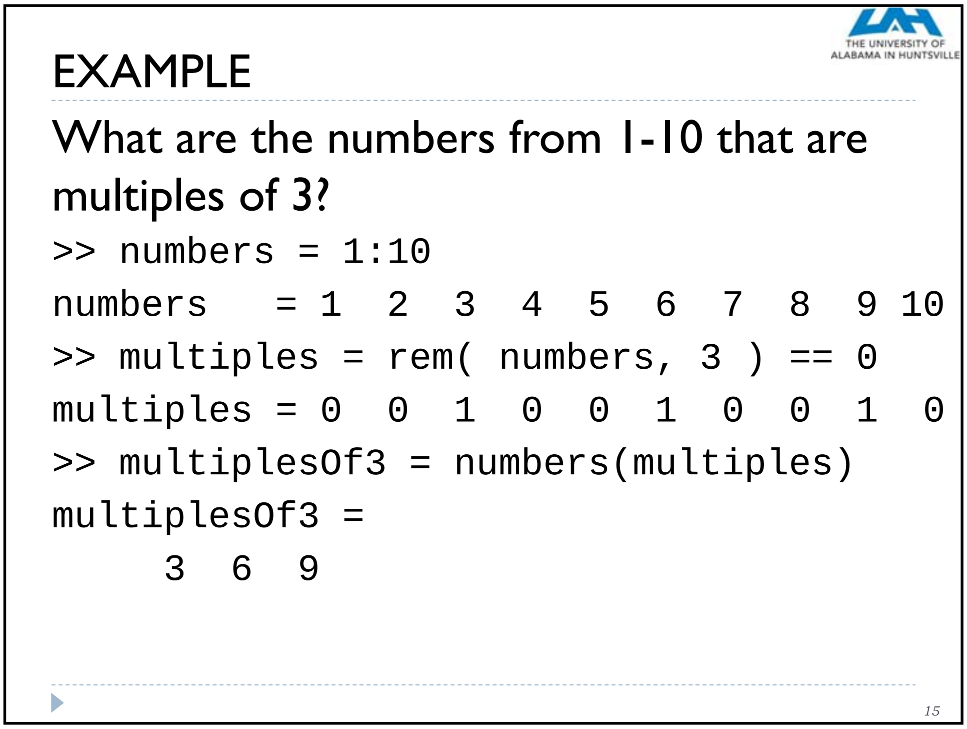
A *logical vector* or *logical array* is a vector/array that has only logical 1's and 0's

- 1's and 0's from mathematical operations don't count
- 1's and 0's from relational comparisons do work
- First time a logical vector/array used in arithmetic, MATLAB changes it to a numerical vector/array



Can use logical vector to get actual values that satisfy relation, not just whether or not relation satisfied. Doing this is called *logical indexing* or *logical subscripting*

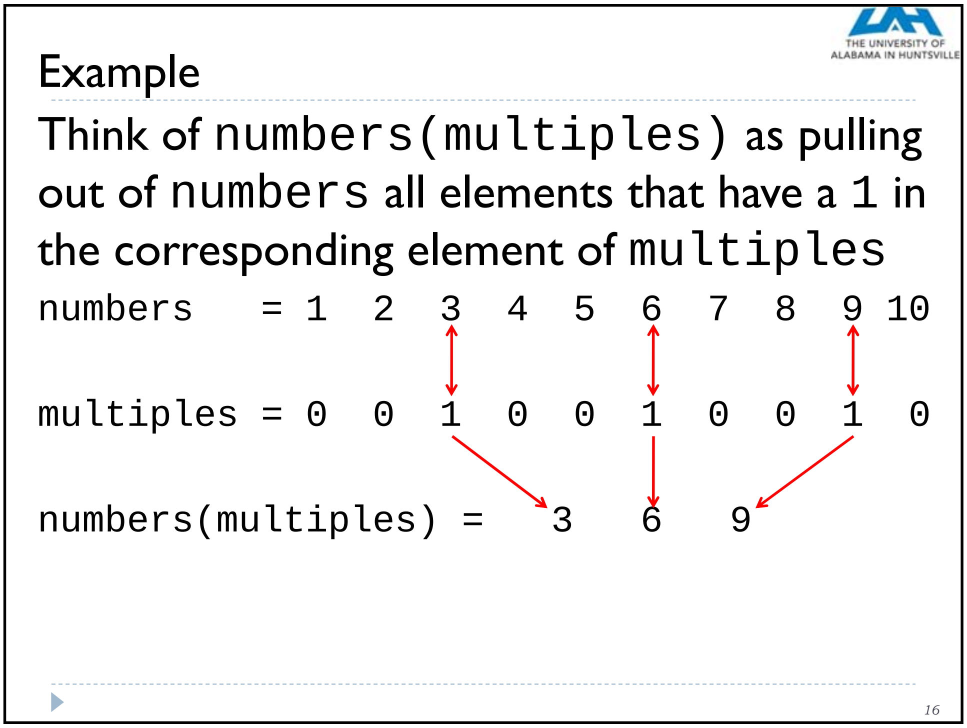
- Do this by using logical vector as index in vector of values. Result is values that satisfy relation, i.e., values for which relationship are 1
- NOTE – technique doesn't quite work with arrays. Won't discuss that case further



EXAMPLE

What are the numbers from 1-10 that are multiples of 3?

```
>> numbers = 1:10
numbers   = 1  2  3  4  5  6  7  8  9 10
>> multiples = rem( numbers, 3 ) == 0
multiples = 0  0  1  0  0  1  0  0  1  0
>> multiplesOf3 = numbers(multiples)
multiplesOf3 =
     3  6  9
```



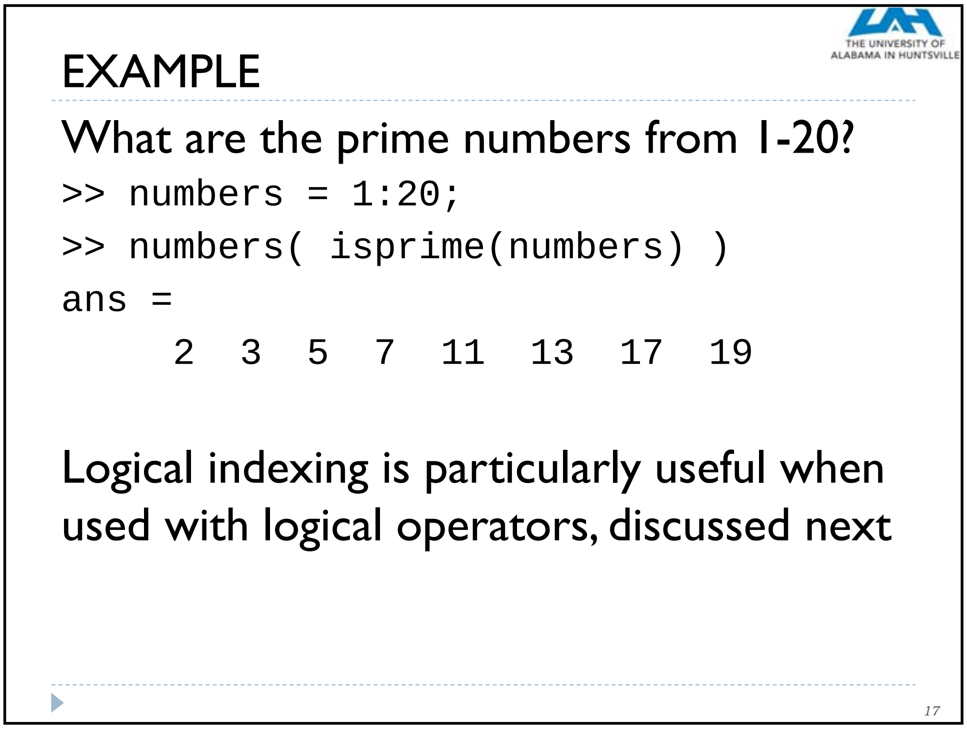
Example

Think of `numbers(multiples)` as pulling out of `numbers` all elements that have a `1` in the corresponding element of `multiples`

```
numbers   = 1  2  3  4  5  6  7  8  9 10

multiples = 0  0  1  0  0  1  0  0  1  0

numbers(multiples) =   3   6   9
```

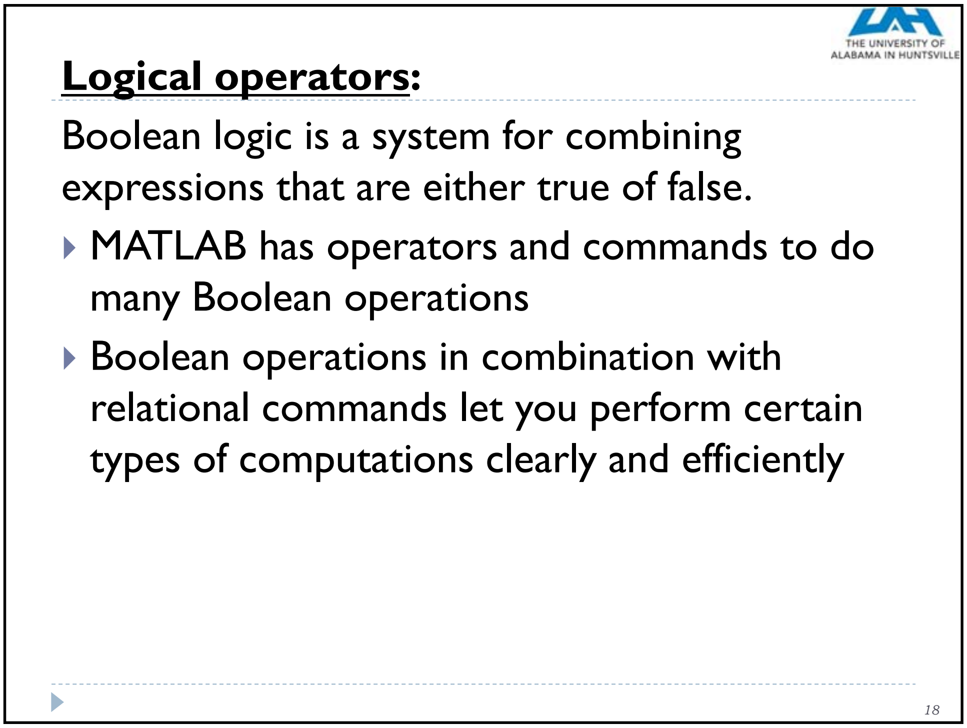


EXAMPLE

What are the prime numbers from 1-20?

```
>> numbers = 1:20;
>> numbers( isprime(numbers) )
ans =
     2   3   5   7   11   13   17   19
```

Logical indexing is particularly useful when used with logical operators, discussed next

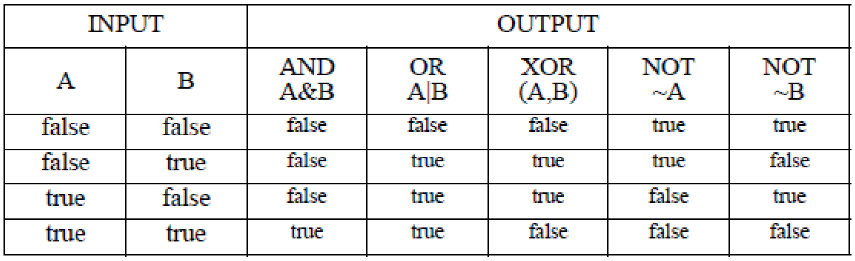


Logical operators:

Boolean logic is a system for combining expressions that are either true of false.

- MATLAB has operators and commands to do many Boolean operations
- Boolean operations in combination with relational commands let you perform certain types of computations clearly and efficiently

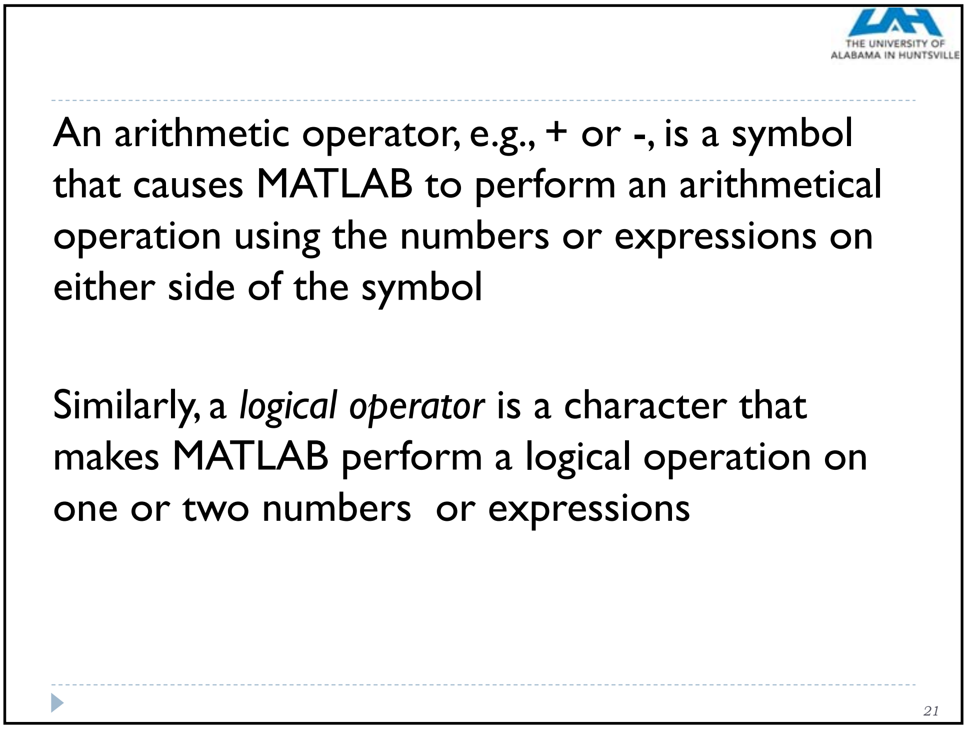
A *truth table* defines the laws of Boolean logic. It gives the output of a logical operation for every possible combination of inputs. The truth table relevant to MATLAB is



INPUT		OUTPUT				
A	B	AND A&B	OR A|B	XOR (A,B)	NOT ~A	NOT ~B
false	false	false	false	false	true	true
false	true	false	true	true	true	false
true	false	false	true	true	false	true
true	true	true	true	false	false	false

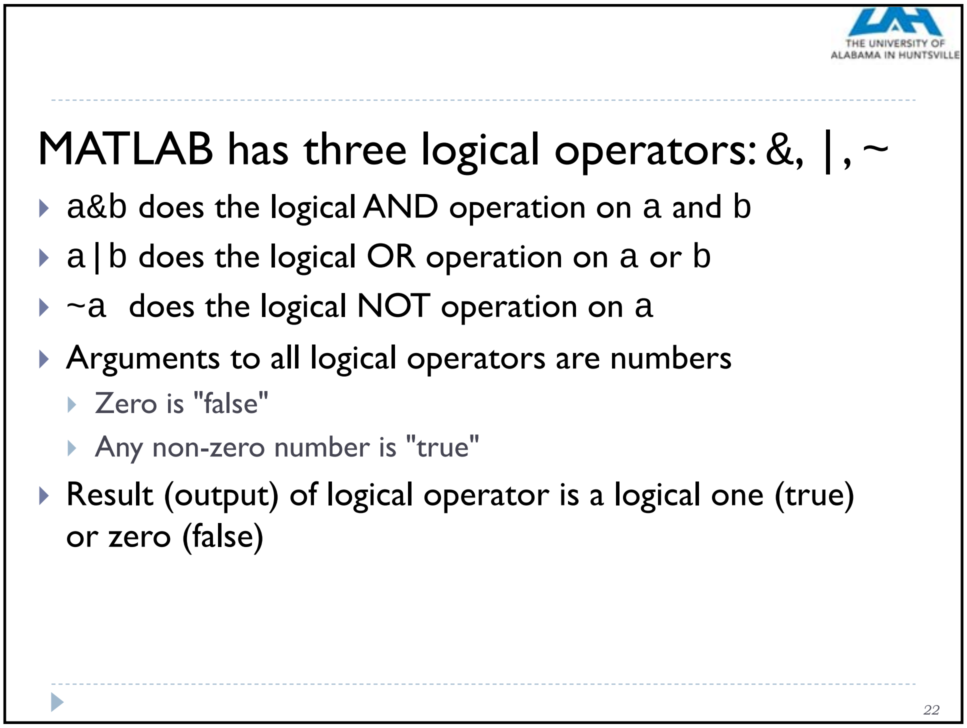
In words, the truth table says

- AND is true if both inputs are true, otherwise it is false
- OR is true if at least one input is true, otherwise it is false
- XOR (exclusive OR) is true if exactly one input is true, otherwise it is false
- NOT is true if the input is false, otherwise it is false



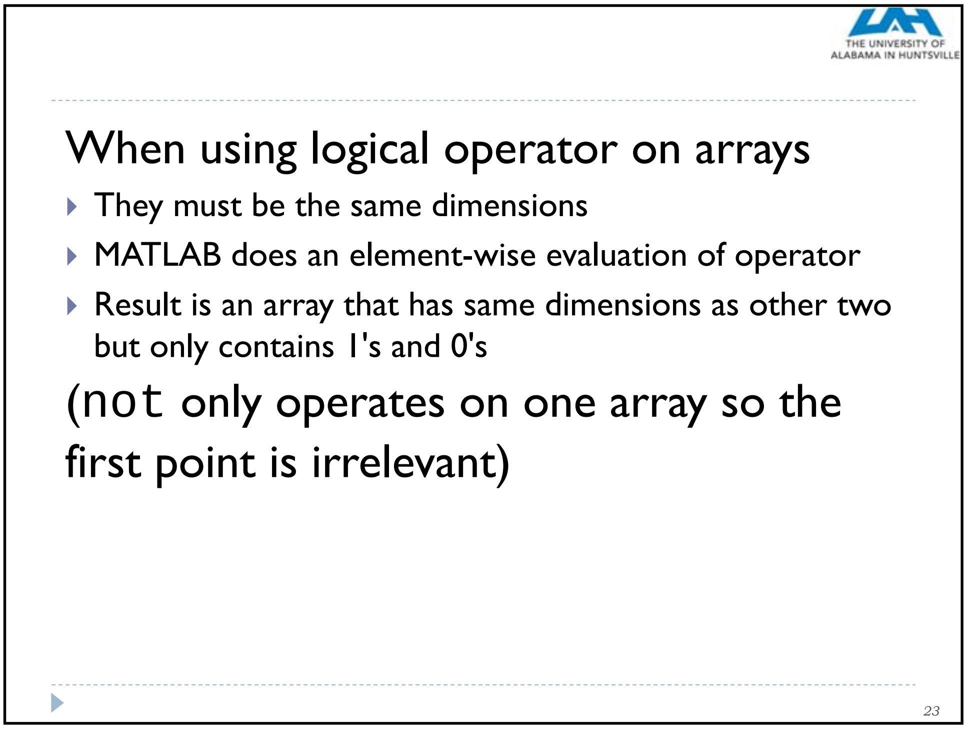
An arithmetic operator, e.g., + or -, is a symbol that causes MATLAB to perform an arithmetical operation using the numbers or expressions on either side of the symbol

Similarly, a *logical operator* is a character that makes MATLAB perform a logical operation on one or two numbers or expressions



MATLAB has three logical operators: &, |, ~

- a&b does the logical AND operation on a and b
- a|b does the logical OR operation on a or b
- ~a does the logical NOT operation on a
- Arguments to all logical operators are numbers
  - Zero is "false"
  - Any non-zero number is "true"
- Result (output) of logical operator is a logical one (true) or zero (false)

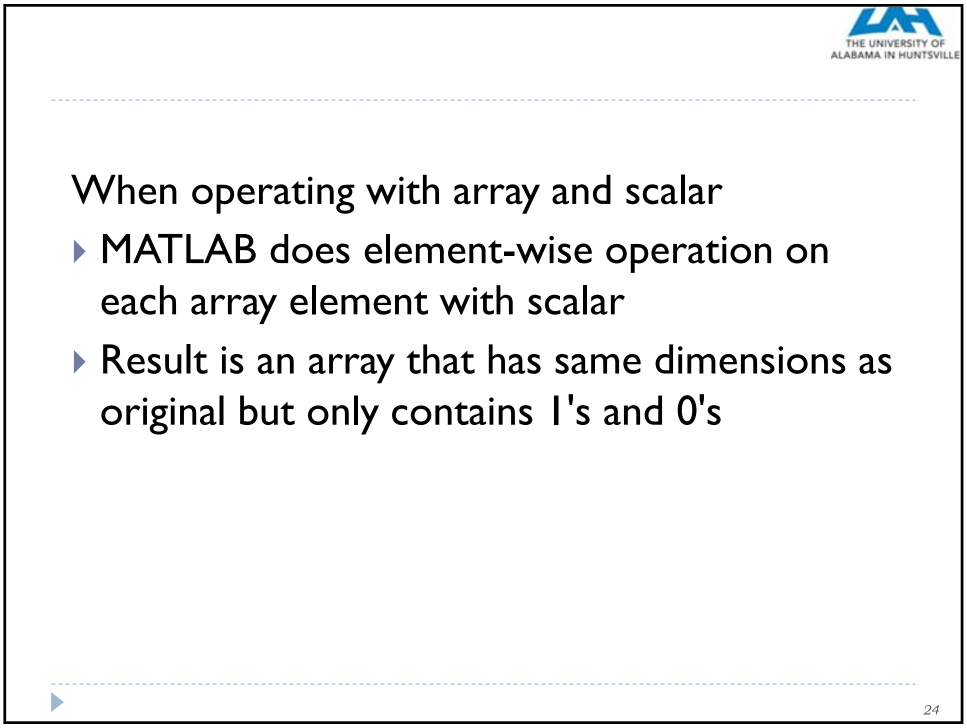


When using logical operator on arrays

- They must be the same dimensions
- MATLAB does an element-wise evaluation of operator
- Result is an array that has same dimensions as other two but only contains 1's and 0's

(`not` only operates on one array so the first point is irrelevant)

23

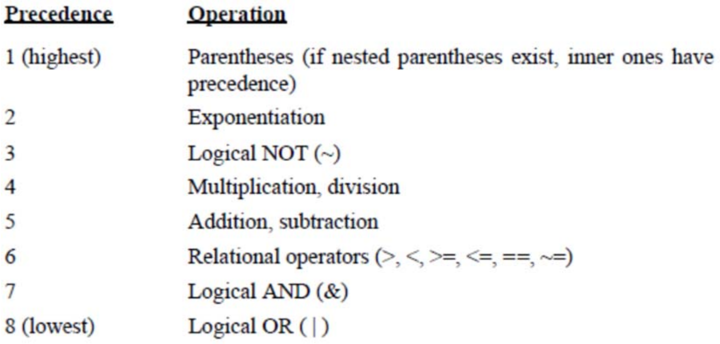


When operating with array and scalar

- MATLAB does element-wise operation on each array element with scalar
- Result is an array that has same dimensions as original but only contains 1's and 0's

24

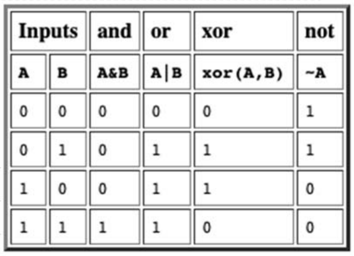
Can combine arithmetic, relational operators, and logical operators. Order of precedence is



Precedence	Operation
1 (highest)	Parentheses (if nested parentheses exist, inner ones have precedence)
2	Exponentiation
3	Logical NOT (~)
4	Multiplication, division
5	Addition, subtraction
6	Relational operators (>, <, >=, <=, ==, ~=)
7	Logical AND (&)
8 (lowest)	Logical OR (|)

Logical Operations (Boolean Functions)

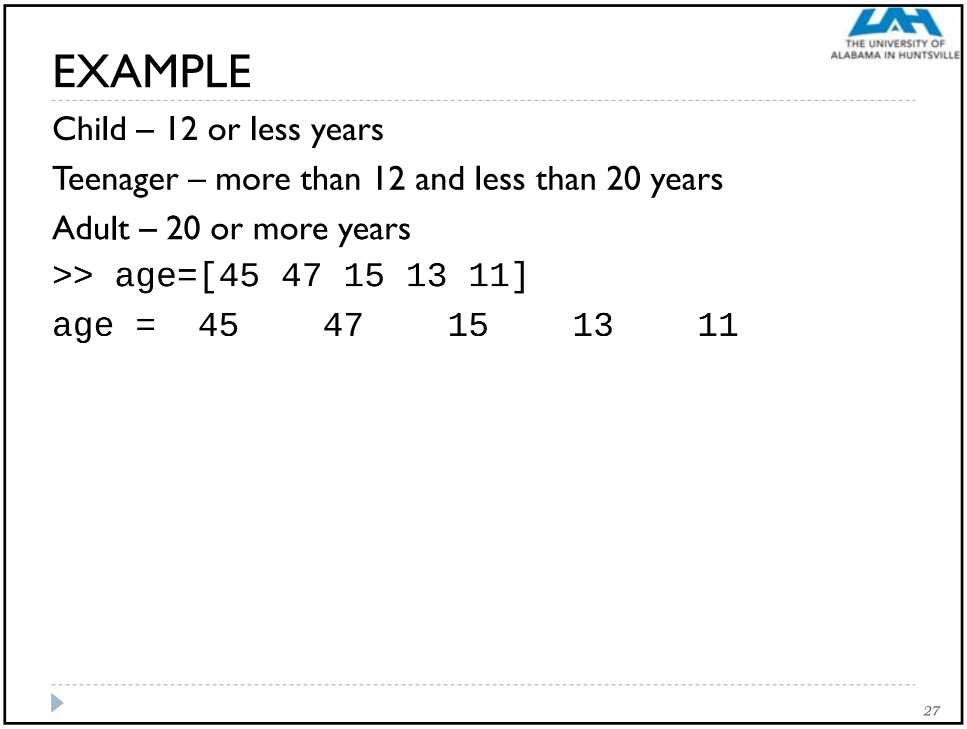
- Individual elements are
  - true – 1
  - false – 0



Inputs		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

&	Find logical AND
~	Find logical NOT
|	Find logical OR
xor	Find logical exclusive-OR
all	Determine if all array elements are nonzero or true
any	Determine if any array elements are nonzero
false	Logical 0 (false)
find	Find indices and values of nonzero elements
islogical	Determine if input is logical array
logical	Convert numeric values to logicals
true	Logical 1 (true)

- https://www.mathworks.com/help/matlab/logical-operations.html



THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

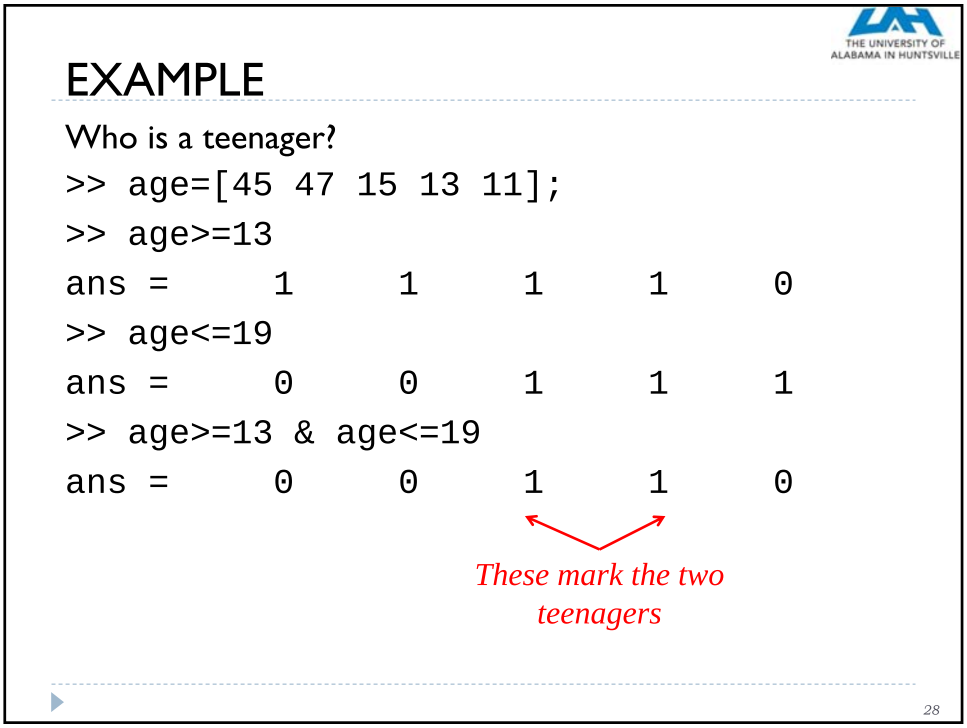
EXAMPLE

Child – 12 or less years

Teenager – more than 12 and less than 20 years

Adult – 20 or more years

```
>> age=[45 47 15 13 11]
age =   45     47     15     13     11
```



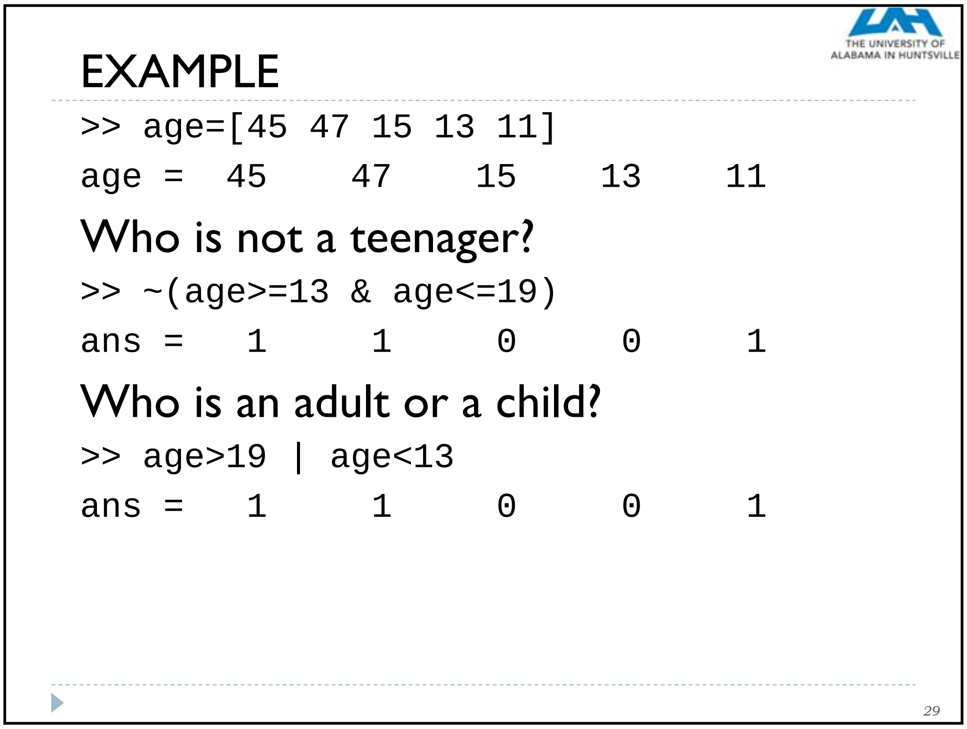
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

EXAMPLE

Who is a teenager?

```
>> age=[45 47 15 13 11];
>> age>=13
ans =      1      1      1      1      0
>> age<=19
ans =      0      0      1      1      1
>> age>=13 & age<=19
ans =      0      0      1      1      0
```

These mark the two teenagers



EXAMPLE

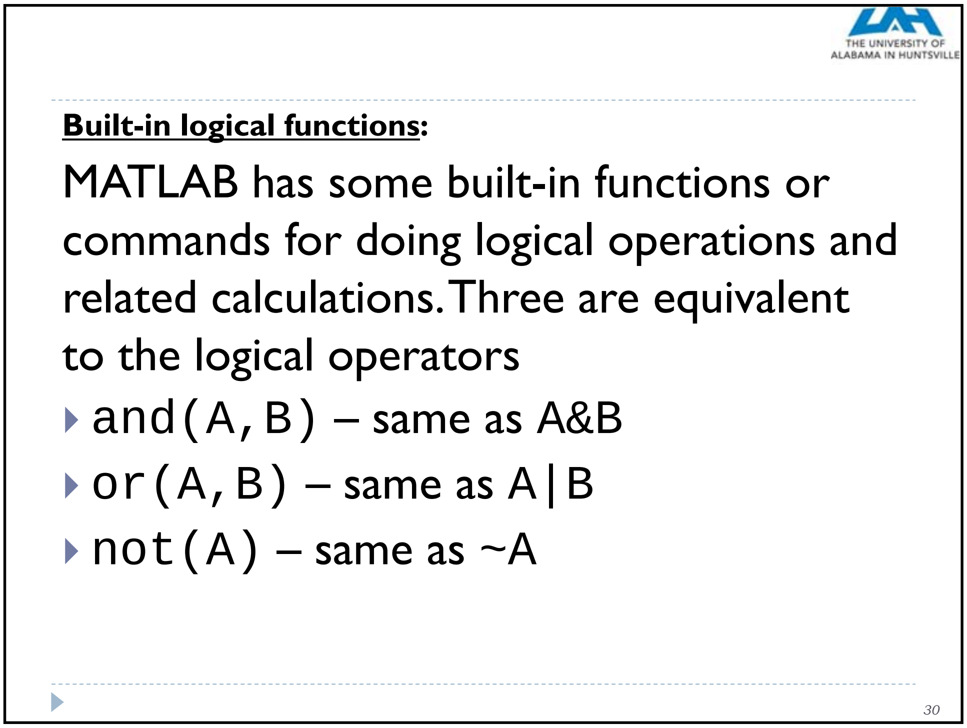
```
>> age=[45 47 15 13 11]
age =  45    47    15    13    11
```

Who is not a teenager?

```
>> ~(age>=13 & age<=19)
ans =   1     1     0     0     1
```

Who is an adult or a child?

```
>> age>19 | age<13
ans =   1     1     0     0     1
```

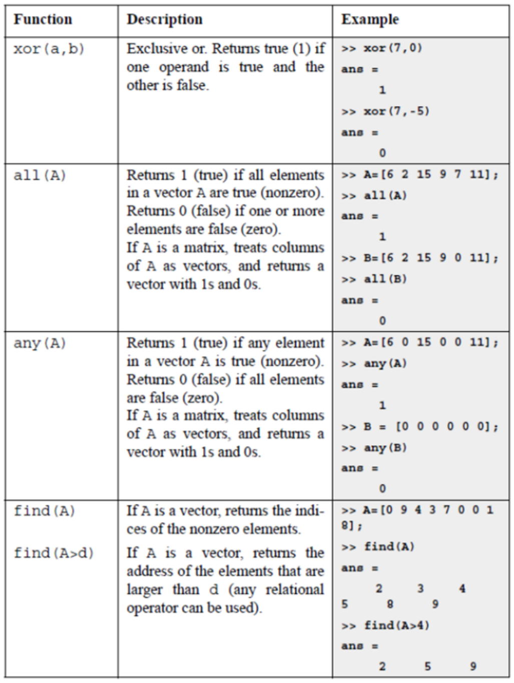


Built-in logical functions:

MATLAB has some built-in functions or commands for doing logical operations and related calculations. Three are equivalent to the logical operators

- `and(A,B)` – same as A&B
- `or(A,B)` – same as A|B
- `not(A)` – same as ~A

MATLAB also has other Boolean functions



Function	Description	Example
xor(a,b)	Exclusive or. Returns true (1) if one operand is true and the other is false.	>> xor(7,0) ans = 1 >> xor(7,-5) ans = 0
all(A)	Returns 1 (true) if all elements in a vector A are true (nonzero). Returns 0 (false) if one or more elements are false (zero). If A is a matrix, treats columns of A as vectors, and returns a vector with 1s and 0s.	>> A=[6 2 15 9 7 11]; >> all(A) ans = 1 >> B=[6 2 15 9 0 11]; >> all(B) ans = 0
any(A)	Returns 1 (true) if any element in a vector A is true (nonzero). Returns 0 (false) if all elements are false (zero). If A is a matrix, treats columns of A as vectors, and returns a vector with 1s and 0s.	>> A=[6 0 15 0 0 11]; >> any(A) ans = 1 >> B = [0 0 0 0 0 0]; >> any(B) ans = 0

find(A)
find(A>d)
	If A is a vector, returns the indices of the nonzero elements.
	If A is a vector, returns the address of the elements that are larger than d (any relational operator can be used).
		>> A=[0 9 4 3 7 0 0 1 8];
		>> find(A)
		ans =
		2 3 4 5 8 9
		>> find(A>4)
		ans =
		2 5 9

Matlab Control structures

- Deciding between alternatives
  - **if** statement
  - **switch** statement
- Repeating
  - **while** loops
  - **for** loops