


Booleans & Conditionals

CSE 231, Bill Punch

Booleans


Boolean Expressions

- George Boole's (mid-1800's) mathematics of logical expressions
- Boolean expressions (conditions) have a value of True or False
- Conditions are the basis of choices in a computer, and, hence, are the basis of the appearance of intelligence in them.

 Michigan State University
CSE 231, Fall 2009 Conditionals & Booleans 3


What is True, and what is False

- true: any nonzero number or nonempty object. 1, 100, "hello", [a,b]
- false: a zero number or empty object. 0, "", []
- Special values called "True" and "False", which are just standins for 1 and 0. However, they print nicely (True or False)
- Also a special value, "None", less than everything and equal to nothing

 Michigan State University
CSE 231, Fall 2009 Conditionals & Booleans 4


Boolean expression

- Every boolean expression has the form:
 - expression booleanOperator expression
- The result of evaluating something like the above is also just true or false.
- However, remember what constitutes true or false in Python!

 Michigan State University
CSE 231, Fall 2009 Conditionals & Booleans 5

Relational Operators

- Less than: <
- Greater than: >
- Equal to: == (Not the same as =)
- Not equal to: !=
- Less than or equal to: <=
- Greater than or equal to: >=

 Michigan State University
CSE 231, Fall 2009 Conditionals & Booleans 6

Examples

- If the value of integer `myInt` is 5, then the value of expression `myInt < 7` is `True`
- If the value of char `myChar` is 'A', then the value of expression `myChar == 'Q'` is `False`



Chained comparisons

- In python, chained comparisons work just like you would expect in a mathematical expression:
- Given `myInt` has the value 5
 - `0 <= myInt <= 5`
 - `True`
 - `0 < myInt <= 5 > 10`
 - `False`



Pitfall

- Be careful of floating point equality comparisons, especially with zero, e.g. `myFloat==0`. Use the converse `“!=“` whenever possible.
- `Result = 2/2.0000000000000001`
- `Result == 1.0`
 - `True`



Compound Expressions

- Logically `0 < X < 3` is actually `(0 < X) and (X < 3)`
- Logical Operators (lower case)
 - `and`
 - `or`
 - `not`



Truth Tables

p	q	not p	p and q	p or q
True	True			
True	False			
False	True			
False	False			



Truth Tables

p	q	not p	p and q	p or q
True	True	False		
True	False	False		
False	True	True		
False	False	True		



Truth Tables

p	q	not p	p and q	p or q
True	True		True	
True	False		False	
False	True		False	
False	False		False	



Michigan State University
CSE 231, Fall 2009

Conditionals &
Booleans

13

Truth Tables

p	q	not p	p and q	p or q
True	True			True
True	False			True
False	True			True
False	False			False



Michigan State University
CSE 231, Fall 2009

Conditionals &
Booleans

14

Truth Tables

p	q	not p	p and q	p or q
True	True	False	True	True
True	False	False	False	True
False	True	True	False	True
False	False	True	False	False



Michigan State University
CSE 231, Fall 2009

Conditionals &
Booleans

15

Compound Evaluation

- Logically $0 < X < 3$ is actually $(0 < X)$ *and* $(X < 3)$
- Evaluate using X with a value of 5:
 $(0 < X)$ *and* $(X < 3)$
- Parenthesis first: (True) *and* (False)
- Final value: False

(Note: parenthesis are not necessary in this case.)



Michigan State University
CSE 231, Fall 2009

Conditionals &
Booleans

16

Precedence & Associativity

- Relational operators have precedence and associativity just like numerical operators.
- See full list in examples



Michigan State University
CSE 231, Fall 2009

Conditionals &
Booleans

17

booleans vs. relationals

- Relational operations always return True or False
- Booleans are different in that:
 - They can return values (that represent True or False)
 - They have "short circuiting"



Michigan State University
CSE 231, Fall 2009

Conditionals &
Booleans

18

Remember!

- 0, "", [] or other "empty" objects are equivalent to False
- anything else is equivalent to True



Returning values

- $2 < 3$ ■ True
- 2 or 3 ■ 2
- 2 and 3 ■ 3

Booleans return the first object (not necessarily True or False) that establishes the expression value (short circuit)



Character Expressions

- $C = 'z'$
- Values of the following?
- $C == 'z'$
 - $C > 'a'$



An Example

- $A = 3$
- $B = 8$
- Evaluate: $A != B$ or $A < -6$
- Precedence: $(A != B)$ or $(A < -6)$
- Evaluate: (True) or (False)
- Evaluate: True
- Shortcut: final value known after $(A != B)$



Example 4

Remember Assignments?

- Format: $lhs = rhs$
- Behavior:
 - *expression* in the rhs is evaluated producing a value
 - the value produced is placed in the location indicated on the lhs



Can do multiple assignments

- `x, y = 2, 3` # assigns `x=2` and `y=3`
- `print x, y` # prints 2 3



Michigan State University
CSE 231, Fall 2009

Conditionals &
Booleans

25

Swap

- Initial values: X is 2, Y is 3
- Behavior: swap values of X and Y
 - Note: `X=Y; Y=X;` doesn't work
 - introduce extra variable "temp"
 - `temp = X;` // save X's value in temp
 - `X=Y;` // assign Y's value to X
 - `Y=temp;` // assign temp's value to Y



Michigan State University
CSE 231, Fall 2009

Conditionals &
Booleans

26

Swap using multiple assignment

- `x, y = 2, 3`
- `print x, y` # prints 2 3
- `x, y = y, x`
- `print x, y` #prints 3 2



Michigan State University
CSE 231, Fall 2009

Conditionals &
Booleans

27

Chaining

- `x = y = 5`
- `print x, y` # prints 5 5



Michigan State University
CSE 231, Fall 2009

Conditionals &
Booleans

28

Compound Assignment

- `y += x` # equivalent to `y = y + x`
- Others
 - `-=, *=, /=, %=`



Michigan State University
CSE 231, Fall 2009

Conditionals &
Booleans

29

Selection

Selection

- Selection is how programs make choices, and it is the process of making choices that provides a lot of the power of computing

Michigan State University
CSE 231, Fall 2009 Conditionals & Booleans 31

Sequential Execution

```

    graph TD
      S1[Statement 1] --> S2[Statement 2]
      S2 -.-> Sn[Statement n]
  
```

Michigan State University
CSE 231, Fall 2009 Conditionals & Booleans 32

Selective Execution

```

    graph TD
      BE{boolean expression} -- true --> S1[statement 1]
      BE -- false --> S2[statement 2]
      S1 --> J(( ))
      S2 --> J
      J --> Exit(( ))
  
```

Michigan State University
CSE 231, Fall 2009 Conditionals & Booleans 33

Compound Statements

- Compound statements involve a set of statements being used as a group
- Most compound statements have:
 - a header, ending with a “:”
 - a “suite” of statements to be executed
- if, for, while are examples of compound statements

Michigan State University
CSE 231, Fall 2009 Conditionals & Booleans 34

More Compound Statement

```

    optional expression
    |
    key expression :
    |
    keyword  statement
            |
            statement
            |
            statement
    
```

Michigan State University
CSE 231, Fall 2009 Conditionals & Booleans 35

Python if statement, round 1

```

if boolean expression :
    suite
  
```

- evaluate the boolean (True or False)
- if True, execute all statements in the suite


Michigan State University
CSE 231, Fall 2009 Conditionals & Booleans 36

Example #6A

This example illustrates a simple selection statement.


Behavior

- Divide two integers, but check for division by zero before performing the division.
- If division by zero
 - print the error
 - exit the program
- print the result of the division


 Michigan State University
 CSE 231, Fall 2009
 Conditionals & Booleans
38

Warning about indentation

- Elements of the “suite” must all be indented the same number of spaces/tabs
- Python only recognizes suites when they are indented the same “distance”
- You must be careful to get the indentation right to get suites right.


 Michigan State University
 CSE 231, Fall 2009
 Conditionals & Booleans
39


Python Selection, Round 2

```

if boolean expression:
    suite1
else:
    suite2
  
```

The process is:

- evaluate the boolean
- if True, run suite1
- if False, run suite2


 Michigan State University
 CSE 231, Fall 2009
 Conditionals & Booleans
40


Example #6B

This example illustrates a simple selection statement.

Python Selection, Round 3

```

if boolean expression1:
    suite1
elif boolean expression2:
    suite2
(as many elif's as you want)
else:
    suiteLast
  
```


 Michigan State University
 CSE 231, Fall 2009
 Conditionals & Booleans
42

if, elif, else, the process

- evaluate boolean expressions until:
 - the boolean expression returns True
 - none of the boolean expressions return True
- if a boolean returns True, run the corresponding suite. Skip the rest of the if
- if no boolean returns True, run the else suite, the default suite

A graphic for a worksheet, featuring a dark blue horizontal bar with the word 'Worksheet' in white. To the left of the bar are several overlapping, semi-transparent squares in shades of blue and grey, creating a layered effect.

Worksheet