CSCI 141

Lecture 7:
Conditionals:
if, else, elif
Announcements

• Slides, schedule, assignments, readings, etc. are posted on the course webpage (not Canvas).

• You can find the course webpage via a link on the Syllabus page of Canvas

• Or just go here: https://facultyweb.cs.wwu.edu/~wehrwes/courses/csci141_19s/

  • Click Schedule to jump to the schedule where materials are posted.
Goals

• Know how to use an if statement to conditionally execute a block of code.

• Know how to use an if/else statement to choose which of two code blocks to execute.

• Understand how conditional statements can be nested to make decisions among more than two possibilities.

• Know how to use if/elif/else statements.
Last Time

• New type: bool

• New operators:
  • comparison <, >, <=, >=, ==, !=
  • logical not, and, or

• Operator precedence
Operator Precedence, Updated Again

Parentheses

Exponentiation (right-to-left)

Unary + and −

Multiplication and Division

Addition and Subtraction

Numerical comparisons <, >, <=, >=, ==, !=

not

and

or

All are evaluated left to right except for exponentiation.

Special case:

\[ 2^{-1} = 0.5 \]

Unspecial but surprising case:

\[ -2^2 = -4 \]

You can look up all the details: https://docs.python.org/3/reference/expressions.html#operator-precedence
Last Time: Boolean Expressions

What does the following statement print?

```python
print((3 == 5 or (3 != 5 and 5 != 7)) and 3 < 5)
```

A: True
B: False
C: true
D: false
Last Time: Boolean Expressions

What does the following statement print?

```python
print((3 == 5 or (3 != 5 and 5 != 7)) and 3 < 5)
print((3 == 5 or ( True and True )) and 3 < 5)
print((3 == 5 or True       ) and 3 < 5)
print(( False or True       ) and 3 < 5)
```

```python
print(True                         ) and 3 < 5)
print(                                ) and 3 < 5)
print(True                         and True)
print(                                )
```
Today’s Quiz

• 3 minutes
Today’s Quiz

• 3 minutes

• Working with a neighbor: do your answers agree? (3 minutes)
Today

• Last week: everything you already knew how to do using a calculator.

• Last lecture: representing and manipulating boolean (true/false) expressions and values.

Today: Making decisions based on the value of a boolean expression.

• Also: a new kind of statement!
Let’s talk about the weather

• You wish to write a software system that recommends what to wear/bring based on the current weather conditions.

• In a later version, you will hook your software up to automated weather sensors that read temperature, wind, and precipitation data in real time.

• For now, we’ll just ask the user.
Let’s talk about the weather

Suppose we have bool variables isRaining

Here’s the logic (pseudocode):

• if it is raining, tell the user to bring a raincoat

Here’s the Python code:

```python
if isRaining:
    print("You should wear a raincoat!")
```
The **if** statement

```python
if isRaining:
    print("You should wear a raincoat!")
```

**Notes:**
- In Python, the indentation is **required**.
- Indenting with tabs or spaces is acceptable.
- We’ll use the most common convention: indent 4 spaces beyond the line with the **if**
- Thonny follows this convention for you
Demo
Demo

• if statement with a condition that evaluates to True vs False

• statements after the indented code block

• multiple lines in the code block
What if it’s not raining?

What if we want to also print something in case it’s not raining?

```python
if isRaining:
    print("Wear a raincoat!")
if not isRaining:
    print("Don’t wear a raincoat!")
```

How many times did we check the value of isRaining?

Could we do any better?

Yes: it’s a common use case to want to choose between two paths of execution (two code blocks).
The if/else Statement

- **if keyword**: a boolean expression (the condition)
- **else keyword**: code block to be executed if the condition evaluates to False

```python
if isRaining:
    print("Wear a raincoat!")
else:
    print("Don’t wear a raincoat!")
```
What does the following program print?

```python
if 2 + 5 == 5
    print(2 + 5)
else:
    print("not equal")
```

A. 2 + 5  
B. 7  
C. 2 + 5 == 5  
D. not equal
What does the following program print?

```python
a = 5
if a >= 5 and a <= 5:
    print(a)
else:
    print("nope")
```

Is there a better way to write the condition?

A. 5  
B. a >= 5  
C. a <= 5  
D. nope
Aim for Simplicity

The program on the right does exactly the same thing, but is easier to read, and therefore is preferable.
Nested Conditionals

If/else lets you choose between two options.

What if there are more than two possibilities?

```python
# assume x and y are numbers
if x < y:
    print("x is less than y")
else:
    if x > y:
        print("x is greater than y")
    else:
        print("x and y must be equal")
```

Note: the conditions still have to be boolean expressions (i.e., they evaluate to True or False)

the inner if/else statement is the indented code block for the else clause of the outer if/else statement.