Conditional Execution

CORE-UA 109.01, Joanna Klukowska
adapted from slides for CSCI-UA.002 by D. Engle, C. Kapp and J. Versoza

let's start with an example

```
name = input("Hello! What is your name? ")

num_of_languages = input("How many programming languages do you know? ")

num_of_languages = int( num_of_languages )

if  num_of_languages == 0 :
    print (name + ", you are about to learn your first programming language!")
else:
    print (name + ", I think you are in a wrong class! Talk to Joanna about this.")
```


let's start with an example

```
name = input("Hello! What is your name? ")

num_of_languages = input("How many programming languages do you know? ")

num_of_languages = int( num_of_languages )

if  num_of_languages == 0 :
    print (name + ", you are about to learn your first programming language!")
else:
    print (name + ", I think you are in a wrong class! Talk to Joanna about this.")
```

- Do think that all of the lines of code are executed by the computer when we run the program? Explain your answer.

let's start with an example

```
name = input("Hello! What is your name? ")

num_of_languages = input("How many programming languages do you know? ")

num_of_languages = int( num_of_languages )

if  num_of_languages == 0 :
    print (name + ", you are about to learn your first programming language!")
else:
    print (name + ", I think you are in a wrong class! Talk to Joanna about this.")
```

- Do think that all of the lines of code are executed by the computer when we run the program? Explain your answer.
- Notice that the `print` instructions after the `if ...` and `else ...` lines are indented. What happens when you try to remove the indent?

`if ... elif ... else` statement

if something is true, do this

```
if some_condition_to_check :
    code to execute
    when the condition turns out
    to be true

continue_with_the_rest_of_the_code
```

- the `if` statement allows us to tell Python to execute the code only is some condition turns out to be true
- **example**

```
today = input("what day is today" )

if today == "Thursday":
    print("Go to the lab at Tisch, room LC19")

print("Have a nice day!")
```


this or that

```
if some_condition_to_check :
    code to execute when the condition
    turns out to be true
else:
    code to execute when the condition
    turns out to be false

continue_with_the_rest_of_the_code
```

- the `if ... else ...` statement allows us to tell Python to execute one block of code if some condition turns out to be true and a different one if that condition turns out to be false
- **example**

```
today = input("what day is today" )

if today == "Thursday":
    print("Go to the lab at Tisch, room LC19")
else:
    print("Go to the lecture at 7 East 12th Street, room 125")

print("Have a nice day!")
```


this or that or that ... or ...

```
if condition_1_to_check :
    code to execute when the condition_1 is true
elif condition_2_to_check :
    code to execute when the condition_2 is true
elif ...
    ...
else:
    code to execute when all of the conditions turn out to be false

continue_with_the_rest_of_the_code
```

- the `if ... elif ... else ...` statement allows us to tell Python to execute a different block of code depending of a collection of different conditions.

this or that or that ... or ...

```
if condition_1_to_check :
    code to execute when the condition_1 is true
elif condition_2_to_check :
    code to execute when the condition_2 is true
elif ...
    ...
else:
    code to execute when all of the conditions turn out to be false

continue_with_the_rest_of_the_code
```

- **example**

```
today = input("what day is today" )

if today == "Thursday":
    print("Go to the lab at Tisch, room LC19")
elif today == "Monday" or today == "Wednesday" :
    print("Go to the lecture at 7 East 12th Street, room 125")
else:
    print("We do not have a class today")
print("Have a nice day!")
```


Boolean expresssions
(or what can be used as a condition)

Boolean values

- Boolean values can be either true or false - yes, there are only two possible values
- **In Python, these values are represented by the reserved words, `True` and `False` (notice that the initial letter is uppercase)**
- comparisons evaluate to Boolean values
  - `10 < 15` evaluates to `True`
  - `"Asia" > "Europe"` evaluates to `False`
  - `(12 + 3) == 15` evaluates to `True`

= vs. ==

- = (equals) is the **assignment operator**
  - assigns the value of the thing on the right to the variable on the left
  - sometimes called binding
  - example: `a = "foo"` gives value of string `"foo"` to the variable `a`
- == (double equals) is the **comparison equality operator**
  - tests if the thing on the left is equal to the thing on the right
  - also called logical equivalence
  - example: `"foo" == "bar"` evaluates to `False`

= VS. ==

- = (equals) is the **assignment operator**
  - assigns the value of the thing on the right to the variable on the left
  - sometimes called binding
  - example: `a = "foo"` gives value of string `"foo"` to the variable `a`
- == (double equals) is the **comparison equality operator**
  - tests if the thing on the left is equal to the thing on the right
  - also called logical equivalence
  - example: `"foo" == "bar"` evaluates to `False`

exercise Try to figure out the value of the following comparisons.

```
'One' == 'one'

'1.0' == 1.0

'one' == 1

1 == 1.0

"one" == 'one'
```


= VS. ==

- = (equals) is the **assignment operator**
  - assigns the value of the thing on the right to the variable on the left
  - sometimes called binding
  - example: `a = "foo"` gives value of string `"foo"` to the variable `a`
- == (double equals) is the **comparison equality operator**
  - tests if the thing on the left is equal to the thing on the right
  - also called logical equivalence
  - example: `"foo" == "bar"` evaluates to `False`

exercise Try to figure out the value of the following comparisons.

```
'One' == 'one'                  False

'1.0' == 1.0                    False

'one' == 1                      False

1 == 1.0                        True

"one" == 'one'                  True
```


dozens of doughnuts 🍩 🍩

What is the output of this program when the user enters different values in response to the prompt?

```
answer = input("you have 12 doughnuts, would you like another dozen? \n")

if answer == 'yes':
    print('you have 24 doughnuts')
else:
    print('you have 12 doughnuts')
```

program user exchange:

```
you have 12 doughnuts, would you like another dozen?
yes
```


dozens of doughnuts 🍩 🍩

What is the output of this program when the user enters different values in response to the prompt?

```
answer = input("you have 12 doughnuts, would you like another dozen? \n")

if answer == 'yes':
    print('you have 24 doughnuts')
else:
    print('you have 12 doughnuts')
```

program user exchange:

```
you have 12 doughnuts, would you like another dozen?
yes

you have 24 doughnuts
```

dozens of doughnuts 🍩 🍩

What is the output of this program when the user enters different values in response to the prompt?

```
answer = input("you have 12 doughnuts, would you like another dozen? \n")

if answer == 'yes':
    print('you have 24 doughnuts')
else:
    print('you have 12 doughnuts')
```

program user exchange:

```
you have 12 doughnuts, would you like another dozen?
yes

you have 24 doughnuts
```

```
you have 12 doughnuts, would you like another dozen?
no
```


dozens of doughnuts 🍩 🍩

What is the output of this program when the user enters different values in response to the prompt?

```
answer = input("you have 12 doughnuts, would you like another dozen? \n")

if answer == 'yes':
    print('you have 24 doughnuts')
else:
    print('you have 12 doughnuts')
```

program user exchange:

```
you have 12 doughnuts, would you like another dozen?
yes

you have 24 doughnuts
```

```
you have 12 doughnuts, would you like another dozen?
no

you have 12 doughnuts
```


dozens of doughnuts 🍩 🍩

What is the output of this program when the user enters different values in response to the prompt?

```
answer = input("you have 12 doughnuts, would you like another dozen? \n")

if answer == 'yes':
    print('you have 24 doughnuts')
else:
    print('you have 12 doughnuts')
```

program user exchange:

```
you have 12 doughnuts, would you like another dozen?
Yes
```


dozens of doughnuts 🍩 🍩

What is the output of this program when the user enters different values in response to the prompt?

```
answer = input("you have 12 doughnuts, would you like another dozen? \n")

if answer == 'yes':
    print('you have 24 doughnuts')
else:
    print('you have 12 doughnuts')
```

program user exchange:

```
you have 12 doughnuts, would you like another dozen?
Yes

you have 12 doughnuts
```

dozens of doughnuts 🍩 🍩

What is the output of this program when the user enters different values in response to the prompt?

```
answer = input("you have 12 doughnuts, would you like another dozen? \n")

if answer == 'yes':
    print('you have 24 doughnuts')
else:
    print('you have 12 doughnuts')
```

program user exchange:

```
you have 12 doughnuts, would you like another dozen?
Yes

you have 12 doughnuts
```

```
you have 12 doughnuts, would you like another dozen?
give me two dozens, please
```


dozens of doughnuts 🍩 🍩

What is the output of this program when the user enters different values in response to the prompt?

```
answer = input("you have 12 doughnuts, would you like another dozen? \n")

if answer == 'yes':
    print('you have 24 doughnuts')
else:
    print('you have 12 doughnuts')
```

program user exchange:

```
you have 12 doughnuts, would you like another dozen?
Yes

you have 12 doughnuts
```

```
you have 12 doughnuts, would you like another dozen?
give me two dozens, please

you have 12 doughnuts
```


math quiz: exponents

What is the output of this program when the user enters the correct answer?

```
answer = input('what is 2 to the 4th power?\n')

if answer == 2 ** 4:
    print('yup, you got it!')
else:
    print('sorry, you got that wrong')
```


math quiz: exponents

What is the output of this program when the user enters the correct answer?

```
answer = input('what is 2 to the 4th power?\n')

if answer == 2 ** 4:
    print('yup, you got it!')
else:
    print('sorry, you got that wrong')
```

- this program has a logical error:
- it always compares a string (containing the correct or incorrect answer) to a number
- the Boolean expression (`answer == 2 ** 4`) **always** evaluates to `False`
- the output of this program is always:

  `sorry, you got that wrong`

comparison operators and logical operators

comparison operators

- there are six comparison operators:
  - == equals (can be called logical equivalence or equality operator)
  - != not equal
  - > greater than
  - < less than
  - >= greater than / equal
  - <= less than / equal
- those operators always return a Boolean value: `True` or `False`
- they work with numbers,
- they also work with strings,

comparison examples

- `5.5 < 13` returns `True`
- `78 != n` returns either `True` or `False` depending on the value of `n` (assuming `n` has a value)
- `"apple" < "pear"` returns `True` - the order is based on the alphabetical ordering of characters,
  - well, not really
  - it actually uses ASCII ordering
- `"Apple" == "apple"` returns `False` - the upper case letters come *before* the lower case letters in ASCII

logical operators

- `and` - takes two operands, one on each side to return `True`, both sides of the operator must be `True`
- `or` - takes two operands, one on each side to return `True`, at least one side of the operator must be `True`
- `not` - only takes one operand to the right to return `True`, the original value on the right must evaluate to `False` two nots cancel eachother out (fun!)

truth tables

- the truth table tells us the value of the logical expression given all different posibilities of the operands

```
p | q | p and q
---------------
f | f |   f
f | t |   f
t | f |   f
t | t |   t
```

```
p | q | p and q
---------------
f | f |   f
f | t |   t
t | f |   t
t | t |   t
```

```
p | not p
-----------
t |  f
f |  t
```


programming challenges

Positive, negative or zero

- Write a program that prompts the user for a number.
- The program should then determine if the number entered is positive negative or zero.
- The program should them print appropriate message, for example
  `Your number is negative`

- Change your above program so instead of numbers it works with temperatures.
  - if the value entered is below freezing, the program should print
    `it is below freezing`
  - if the value entered is above freezing, the program should print
    `it is still warm`
  - if the value entered is exactly at 32 degrees, the program should print
    `it is just border-line, hope it is going up!`

Calculating bonuses

- You're the manager of a large, distributed sales force
- You want to create an easy to use tool that will allow your sales staff to do the following:
  - Input their monthly sales amount
  - Determine if they made their monthly quota of $10,000
  - If they made their quota, they are eligible for a bonus of 50% of whatever they sold above the $10,000
  - If they made their quota, they should receive a "Good Job!" message
  - At the end of the program you should print out how much their bonus will be.