



# Tutoring Section 6

---

Conditionals and Iteration

# Logistics

---

- Autograder is working again! Grades will come in soon.
  - Feedback Form:
    - Form: <https://tinyurl.com/feedbackD8Kevin>
  - Again, Tutor Office Hours (exclusively open for you all)
    - **Tuesday: 10:30-11:00am & 1:00-1:30pm**
    - *Please let me know if you are attending*
- All resources can be found on [kevin-miao.com](http://kevin-miao.com)
-

# Association vs Causation

---

After grading the homework, I wanted to discuss these questions:

- What is the difference between an **association** and a **causation**?  
association - some sort of relationship  
causation - something causes the other directionality
  - What are **confounding factors**?  
Variables that can basically confuse the  
clear relationship
  - When do we need to establish a **randomized control experiment**?  
when we want to eliminate confounding  
factors.
-

# Today

---

- Mini Review
  - *Conditionals*
  - *Iteration*
  - *What to do or not to do?*
- Worksheet

# Conditionals

---

- **Goal:** Depending on a certain value, we want to use different lines of code.

- **Structure:**

```
if <boolean>:  
    do something  
elif <boolean>:  
    do something else  
else:  
    do something different
```

- **Example:**

```
if color == 'Blue':  
    print("Go Bears!")  
else:  
    print("I hate this color")
```

---

# Iteration

---

- **Goal:** We want to do same thing for each item in a list/array.

- **Structure:**  variable name: arbitrary

```
for something in list/array:  
    do something
```

- **Example:**

```
for book in ["Book1", "Book2", "Book3"]:  
    print(book)
```

- Another implementation is to do the same thing x number of times:

```
x = 0
```

```
for i in np.arange(100):  
    x = x + 1
```

---

# Common mistakes/issues

- Is this correct?

```
for name in ['David', 'Swupnil']:  
    return name
```

return only for functions!

- Has to be list/array

```
for i in len(100):  
    print("Repeat 100 times")
```

① len only works on strings/ lists  
② needs to be list

- Are these two the same?

```
if x % 2 == 0:  
    print('Divisible by 2')  
else:  
    print('Divisible by 4')
```

if  
else

```
if x % 2 == 0:  
    print('Divisible by 2')  
print('Not divisible by 2')
```

not divisible by 2  
← always evaluated

# Worksheet

Link: <https://tinyurl.com/d8tutweek6>



# Q1

---

## Practice Problems

**Question 1.** Examine the function, then answer the questions below. It has been written with a purposely vague name and arguments!

```
def mystery_function(x):  
    if (x > 0):  
        return "Positive"  
    elif (x < 0):  
        return "Negative"  
    else:  
        return "Neither"
```

`print("Hello")` ← never  
evaluated

**What does each of the following return?**

1. `mystery_function(10)`  
"positive"
2. `mystery_function(-1)`  
"negative"
3. `mystery_function(0)`  
"neither"

# Q2.1

---

**Question 2.** The for loop statement below stores the length of each name in `names` in a new array called `lengths`.

```
lengths = make_array()
names = make_array('Bob', 'Sarah', 'Michael', 'Sam')
```

```
for name in names:
    lengths = np.append(lengths, len(name))
```

**2.1** For each iteration below, fill in the value of `name` as well as what `lengths` looks like.

Iteration 1: `name` = "Bob", `lengths` = [3]  
Iteration 2: `name` = "Sarah", `lengths` = [3, 5]  
Iteration 3: `name` = "Michael", `lengths` = [3, 5, 7]  
Iteration 4: `name` = "Sam", `lengths` = [3, 5, 7, 3]

---

## Q2.2-2.3

---

**2.2** Now, let's say that instead of storing lengths, we want to store the name as long as the length of the name is greater than 4. Fill in the following for loop statement such that `longer` contains these names.

```
longer = make_array()
```

```
for name in names:  
    if len(name) > 4:  
        longer = np.append(longer, name)
```

**2.3** What names would `longer` contain after the for loop executes?

`"Sarah"` and `"Michael"`

---

# Q2.4

---

**2.4** Finally, look at this last for loop below. What values does `i` take on throughout? How is `i` used as compared to the way `name` is used in the previous for loops?

```
counter = 0
for i in np.arange(1000):
    counter = counter + 1
```

• for — in np.arange(1000)

Ⓘ `i` is an integer looping from [0...999]

Ⓙ we don't care about the value of `i` specifically!

---

# Q3.1

**Question 3.** Suppose you have an array called `salaries`, containing the salary information of 5 individuals. You would like to determine what percentage of the total salaries each individual's salary comprises. You want to output an array, `proportion` where the  $i$ th element of `proportion` corresponds to what percentage of the total salary `salary.item(i)` is.

For example, if `salaries` was equal to an array `[1, 2, 3, 1, 3]`, then `proportion.item(0)` would be 0.1.

**3.1** Your friend writes some code, but it doesn't work! Find the error that your friend made. What would the code output if executed as is? How would you fix it?

```
salaries = make_array(25, 50, 100, 25, 100)
total = sum(salaries)
```

```
for salary in salaries:
    proportion = make_array()
    percentage = salary/total
    proportion = np.append(proportion, percentage)
```

*T*, [ 0.333 ... ]

*should be there!*  
*# you'd be making an empty array the whole time.*

## Q3.2

---

**3.2** You fix the error described above, but in doing so, break something else. Again, find the error in the code below. What would the code output if executed as is? How would you fix it?

```
salaries = make_array(25, 50, 100, 25, 100)
total = sum(salaries)
proportion = make_array()
```

```
for salary in salaries:
    percentage = salary/total
    np.append(proportion, percentage)
```

proportion = []

↑

not saving it

└─▶

ERROR would  
return

→

proportion =

# End of Section

---

- Please complete the anonymous Feedback form so I can improve my teaching:
    - <https://tinyurl.com/feedbackD8Kevin>
  - Solutions and notes will be posted as soon as possible.
  - Email me if you have any questions: [kevinmiao@berkeley.edu](mailto:kevinmiao@berkeley.edu)
-