UCLA College | Social Sciences Economics

Day 2: Data Types, Flow Control & Loops, Lists & Dictionaries

Paul Schumacher, MSc Quantitative Economics

Announcements

Attendance, Jupiter Check

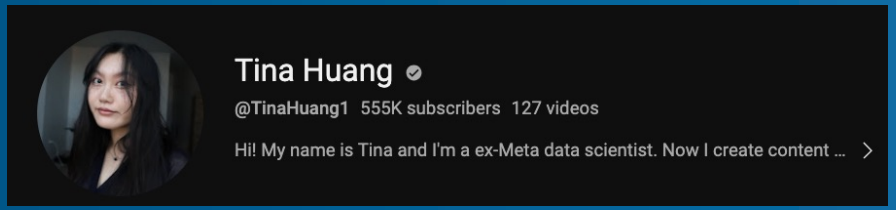
Link to “Special Lecture”? Today 6:30 – 7:30

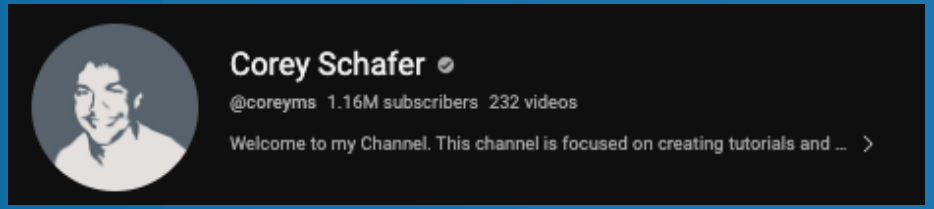
Midterm Review Session Tomorrow 2:00 – 3:30

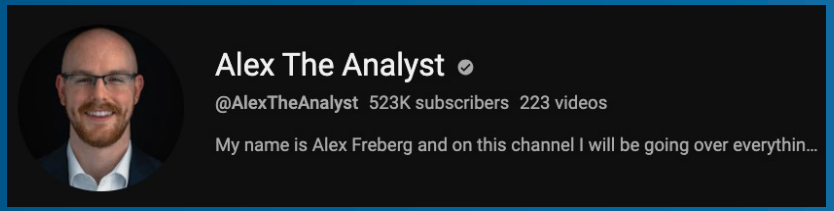
Quick Intro – 3 Students

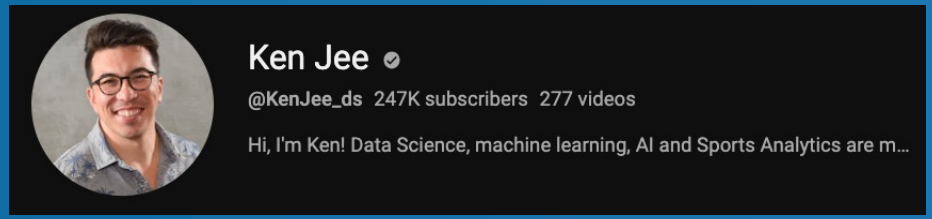
1. Name, School, Hobbies, …
2. What are your future plans (university, work, hobbies, travel)
3. Experience in Python? Why do you want to learn it? What do you wish to get out of this course?
4. Which apps do you use most on your phone?
5. What was the best/ worst thing that happened to you last month?

Learning outside the classroom!









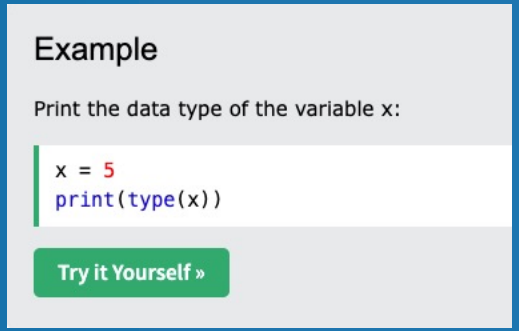
Learning outside the classroom!

- **Tina Huang → BR1**
- **Alex the Analyst → BR2**
- **Corey Schafer → BR3**
- **Ken Jee → BR4**

1. How many followers does he/she have? How many videos? How many views per video?
2. Name five topics he/she is covering in his/her videos!
3. What do you think about his/her thumbnails/ headlines?
4. Does he/she have any collaborations?
5. From a scale from 1 (*professional*) to 10 (*clickbait*), where would you rank him/her?

Class Recap: I. Data Types

Example	Data Type
`x = "Hello World"`	str
`x = 20`	int
`x = 20.5`	float
`x = 1j`	complex
`x = ["apple", "banana", "cherry"]`	list
`x = ("apple", "banana", "cherry")`	tuple
`x = range(6)`	range
`x = {"name" : "John", "age" : 36}`	dict



Link: https://www.w3schools.com/python/python_datatypes.asp

Class Recap: II. Arithmetic Operators

Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y

Link:
https://www.w3schools.com/python/gloss_python_arithmetic_operators.asp

Class Recap: III. String Extraction

`start <= x < stop` with `[start:stop]`

position from the back	-5	-4	-3	-2	-1
postion from the front	0	1	2	3	4
	a	**b**	**c**	**d**	**e**

```
s = 'abcde'

print(s[1:3])
# bc

print(s[:3])
# abc

print(s[1:])
# bcde
```

```
print(s[-4:-2])
# bc

print(s[:-2])
# abc

print(s[-4:])
# bcde
```

Link:
https://www.w3schools.com/python/gloss_python_arithmetic_operators.asp

Class Recap: IV. Functions you should know

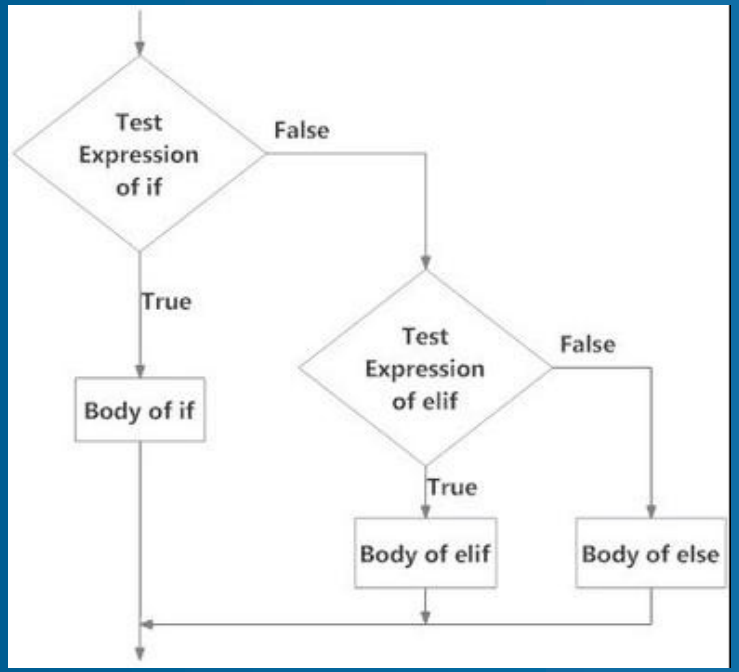
```
x = "Department of Economics"
```

number of elements	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
position in string	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
	D	**e**	**p**	**a**	**r**	**t**	**m**	**e**	**n**	**t**	**_**	**o**	**f**	**_**	**E**	**c**	**o**	**n**	**o**	**m**	**i**	**c**	**s**

- len(x) → total number of elements
- Find() → position in string
- upper() → capitalize everything
- Lower() → only lowercase letters
- input() → change/give input values for variables

Every answer is one Google search away!

Class Recap: V. Flow Control & Loops

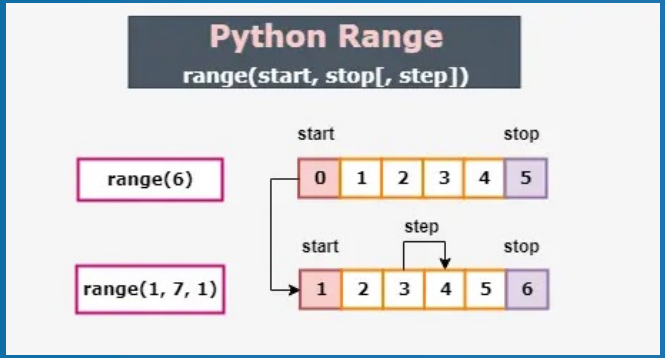



```
In [20]: x = 5
         if x > 9:
             x -= 2
         elif x < 9:
             x += 2
         else:
             x =x% 9
         print(x)

         7
```

Class Recap: V. Flow Control & Loops

```
for i in range(11):
    for j in range(11):
        if i%2 == 0 or j%2 == 0:
            continue
        else:
            print(i,j)
```



i	0	1	2	3	4	5	6	7	8	9	10
j	0	1	2	3	4	5	6	7	8	9	10

i		1		3		5		7		9	
j		1		3		5		7		9	

Pair every i with every j

Class Recap: V. Flow Control & Loops

```
y = range(101)
i = 0
total = 0
while i  < 101:
    total = total + y[i]
    i += 1
print(total)
```

```
5050
```

i is value for position in y

y	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
		51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100

Class Recap: V. Flow Control & Loops

```
x = 100
tot = 0
while x > 0:
    if tot > 200:
        break
    tot += x
    x = x //2
    print(x)
```

```
50
25
12
6
3
1
0
```

“as long as” $x > 0$

Del vs. pop vs. remove

```
fruits = ['apple', 'banana', 'cherry']
fruits.remove('apple')
print(fruits)
```

```
['banana', 'cherry']
```

Never used it in my life!!

```
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9]

# deleting the third item
del my_list[2]

# Output: [1, 2, 4, 5, 6, 7, 8, 9]
print(my_list)

# deleting items from 2nd to 4th
del my_list[1:4]

# Output: [1, 6, 7, 8, 9]
print(my_list)

# deleting all elements
del my_list[:]

# Output: []
print(my_list)
```

removes item/ slices at a **given index**

For both (del, pop):
The index of item is needed!!!
Not direct removal of value

```
fruits = ['apple', 'banana', 'cherry']
x = fruits.pop(2)
print(fruits)
```

```
['apple', 'banana']
```

pop() method removes the element at the **specified position**

same

= del fruits[2]

```
# create a list
prime_numbers = [2, 3, 5, 7, 9, 11]

# remove 9 from the list
prime_numbers.remove(9)

# Updated prime_numbers List
print('Updated List: ', prime_numbers)

# Output: Updated List:  [2, 3, 5, 7, 11]
```

For remove: no index of item is needed!!! → direct removal of value

Removes first "matching" item that is given in the brackets

Any Questions?