INTERMEDIATE PYTHON FOR DATA SCIENCE

while loop

if-elif-else

📄 `control.py`

                                        Goes through construct only once!
```
z = 6
if z % 2 == 0 :   True
    print("z is divisible by 2")   Executed
elif z % 3 == 0 :
    print("z is divisible by 3")
else :
    print("z is neither divisible by 2 nor by 3")

...     Moving on
```

While loop = repeated if statement

While

```
while condition :
    expression
```

- Numerically calculating model

- "repeating action until condition is met"

- Example

  - Error starts at 50

  - Divide error by 4 on every run

  - Continue until error no longer > 1

While

Error starts at 50
Divide error by 4 on every run
Continue until error no longer > 1

```
while condition :
    expression
```

📄 while_loop.py

```python
error = 50.0

while error > 1 :
    error = error / 4
    print(error)
```

While

```
while condition :
      expression
```

📄 **while_loop.py**

```
error = 50.0

      50.0
while error > 1 :  True
    error = error / 4
    print(error)
```

```
Output:
12.5
```

While

```
while condition :
    expression
```

📄 while_loop.py

```python
error = 50.0
        12.5
while error > 1 :  True
    error = error / 4
    print(error)
```

```
Output:
12.5
3.125
```

While

```
while condition :
    expression
```

📄 while_loop.py

```
error = 50.0
            3.125
while error > 1 :  True
    error = error / 4
    print(error)
```

```
Output:
12.5
3.125
0.78125
```

While

```
while condition :
    expression
```

while_loop.py

```
error = 50.0
        0.78125
while error > 1 :   False
    error = error / 4     not executed
    print(error)
```

```
Output:
12.5
3.125
0.78125
```

While

```
while condition :
    expression
```

📄 while_loop.py

```python
error = 50.0

while error > 1 :    always True
    # error = error / 4
    print(error)
```

```
Output:
50
50
50
50
50
50
50
...
```

DataCamp: session disconnected
Local system: Control + C

INTERMEDIATE PYTHON FOR DATA SCIENCE

Let's practice!

for loop

for loop

```
for var in seq :
    expression
```

*"for each **var** in **seq**, execute expression"*

fam

```python
fam = [1.73, 1.68, 1.71, 1.89]
print(fam)
```

Output:
[1.73, 1.68, 1.71, 1.89]

fam

📄 `family.py`

```python
fam = [1.73, 1.68, 1.71, 1.89]
print(fam[0])
print(fam[1])
print(fam[2])
print(fam[3])
```

Output:
1.73
1.68
1.71
1.89

for loop

```
for var in seq :
    expression
```

📄 `family.py`

```python
fam = [1.73, 1.68, 1.71, 1.89]

for height in fam :
    print(height)
```

for loop

```
for var in seq :
    expression
```

📄 `family.py`

```
fam = [1.73, 1.68, 1.71, 1.89]

for height in fam :
    print(height)
```

**first iteration
height = 1.73**

Output:
`1.73`

for loop

```
for var in seq :
    expression
```

📄 `family.py`

```python
fam = [1.73, 1.68, 1.71, 1.89]

for height in fam :
    print(height)
```

**second iteration
height = 1.68**

```
Output:
1.73
1.68
```

for loop

```
for var in seq :
    expression
```

📄 `family.py`

```python
fam = [1.73, 1.68, 1.71, 1.89]

for height in fam :
    print(height)
```

no access to indexes

```
Output:
1.73
1.68
1.71
1.89
```

for loop

```
for var in seq :
    expression
```

📄 `family.py`

```python
fam = [1.73, 1.68, 1.71, 1.89]

???
```

Output:
~~1.73~~ index 0: 1.73
~~1.68~~ index 1: 1.68
~~1.71~~ index 2: 1.71
~~1.89~~ index 3: 1.89

enumerate

```
for var in seq :
    expression
```

📄 `family.py`

```python
fam = [1.73, 1.68, 1.71, 1.89]

for index, height in enumerate(fam) :
    print("index " + str(index) + ": " + str(height))
```

Output:
```
index 0: 1.73
index 1: 1.68
index 2: 1.71
index 3: 1.89
```

Loop over string

```
for var in seq :
    expression
```

📄 strloop.py

```python
for c in "family" :
    print(c.capitalize())
```

```
Output:
F
A
M
I
L
Y
```

Let's practice!

Loop Data Structures Part 1

Dictionary

```
for var in seq :
        expression
```

📄 dictloop.py

```python
world = { "afghanistan":30.55,
          "albania":2.77,
          "algeria":39.21 }

for key, value in world :
    print(key + " -- " + str(value))
```

```
Output:
ValueError: too many values to unpack (expected 2)
```

Dictionary

```
for var in seq :
    expression
```

📄 dictloop.py

```python
world = { "afghanistan":30.55,
          "albania":2.77,
          "algeria":39.21 }

for key, value in world.items() :
    print(key + " -- " + str(value))
```

```
Output:
algeria -- 39.21
afghanistan -- 30.55
albania -- 2.77
```

Dictionary

```
for var in seq :
    expression
```

📄 dictloop.py

```python
world = { "afghanistan":30.55,
          "albania":2.77,
          "algeria":39.21 }
    key  value
for k, v in world.items() :
    print(k + " -- " + str(v))
```

```
Output:
algeria -- 39.21
afghanistan -- 30.55
albania -- 2.77
```

Numpy Arrays

```
for var in seq :
    expression
```

📄 nploop.py

```python
import numpy as np
np_height = np.array([1.73, 1.68, 1.71, 1.89, 1.79])
np_weight = np.array([65.4, 59.2, 63.6, 88.4, 68.7])
bmi = np_weight / np_height ** 2

for val in bmi :
    print(val)
```

Output:
```
21.852
20.975
21.750
24.747
21.441
```

2D Numpy Arrays

```
for var in seq :
    expression
```

📄 np2dloop.py

```python
import numpy as np
np_height = np.array([1.73, 1.68, 1.71, 1.89, 1.79])
np_weight = np.array([65.4, 59.2, 63.6, 88.4, 68.7])
meas = np.array([np_height, np_weight])

for val in meas :
    print(val)
```

```
Output:
[ 1.73  1.68  1.71  1.89  1.79]
[ 65.4  59.2  63.6  88.4  68.7]
```

2D Numpy Arrays

```
for var in seq :
    expression
```

📄 np2dloop.py

```python
import numpy as np
np_height = np.array([1.73, 1.68, 1.71, 1.89, 1.79])
np_weight = np.array([65.4, 59.2, 63.6, 88.4, 68.7])
meas = np.array([np_height, np_weight])

for val in np.nditer(meas) :
    print(val)
```

Output:
1.73
1.68
1.71
1.89
1.79

Output (cont):
65.4
59.2
63.6
88.4
68.7

Recap

- Dictionary

  - ```
    for key, val in my_dict.items() :
    ```

- Numpy array

  - ```
    for val in np.nditer(my_array) :
    ```

Let's practice!

INTERMEDIATE PYTHON FOR DATA SCIENCE

Loop Data Structures Part 2

brics

```
           country      capital    area   population
BR          Brazil     Brasilia   8.516       200.40
RU          Russia       Moscow  17.100       143.50
IN           India    New Delhi   3.286      1252.00
CH           China      Beijing   9.597      1357.00
SA    South Africa     Pretoria   1.221        52.98
```

📄 dfloop.py

```python
import pandas as pd
brics = pd.read_csv("brics.csv", index_col = 0)
```

for, first try

country   capital   area   population
BR   Brazil   Brasilia   8.516   200.40
RU   Russia   Moscow   17.100   143.50
IN   India   New Delhi   3.286   1252.00
CH   China   Beijing   9.597   1357.00
SA   South Africa   Pretoria   1.221   52.98

📄 dfloop.py

```
import pandas as pd
brics = pd.read_csv("brics.csv", index_col = 0)

for val in brics :
    print(val)
```

Output:
country
capital
area
population

iterrows

	country	capital	area	population
BR	Brazil	Brasilia	8.516	200.40
RU	Russia	Moscow	17.100	143.50
IN	India	New Delhi	3.286	1252.00
CH	China	Beijing	9.597	1357.00
SA	South Africa	Pretoria	1.221	52.98

📄 dfloop.py

```python
import pandas as pd
brics = pd.read_csv("brics.csv", index_col = 0)

for lab, row in brics.iterrows() :
    print(lab)
    print(row)
```

Output:
```
BR
country          Brazil
capital        Brasilia
area              8.516
population        200.4
Name: BR, dtype: object
...
```

Output (cont):
```
RU
country          Russia
capital          Moscow
area               17.1
population        143.5
Name: RU, dtype: object
IN ...
```

Selective print

```
          country      capital      area   population
BR         Brazil     Brasilia     8.516       200.40
RU         Russia       Moscow    17.100       143.50
IN          India    New Delhi     3.286      1252.00
CH          China      Beijing     9.597      1357.00
SA   South Africa     Pretoria     1.221        52.98
```

📄 dfloop.py

```python
import pandas as pd
brics = pd.read_csv("brics.csv", index_col = 0)

for lab, row in brics.iterrows() :
    print(lab + ": " + row["capital"])
```

Output:
```
BR: Brasilia
RU: Moscow
IN: New Delhi
CH: Beijing
SA: Pretoria
```

Add column

```
              country      capital      area  population
BR             Brazil     Brasilia     8.516      200.40
RU             Russia       Moscow    17.100      143.50
IN              India    New Delhi     3.286     1252.00
CH              China      Beijing     9.597     1357.00
SA       South Africa     Pretoria     1.221       52.98
```

📄 dfloop.py

```python
import pandas as pd
brics = pd.read_csv("brics.csv", index_col = 0)

for lab, row in brics.iterrows() :
    brics.loc[lab, "name_length"] = len(row["country"])
print(brics)
```

Creating Series on every iteration

Output:
```
              country      capital      area  population  name_length
BR             Brazil     Brasilia     8.516      200.40            6
RU             Russia       Moscow    17.100      143.50            6
IN              India    New Delhi     3.286     1252.00            5
CH              China      Beijing     9.597     1357.00            5
SA       South Africa     Pretoria     1.221       52.98           12
```

apply

```
                country      capital     area   population
BR               Brazil     Brasilia    8.516       200.40
RU               Russia       Moscow   17.100       143.50
IN                India    New Delhi    3.286      1252.00
CH                China      Beijing    9.597      1357.00
SA         South Africa     Pretoria    1.221        52.98
```

📄 dfloop.py

```python
import pandas as pd
brics = pd.read_csv("brics.csv", index_col = 0)

brics["name_length"] = brics["country"].apply(len)

print(brics)
```

len()

Output:

```
                country      capital     area   population   name_length
BR               Brazil     Brasilia    8.516       200.40             6
RU               Russia       Moscow   17.100       143.50             6
IN                India    New Delhi    3.286      1252.00             5
CH                China      Beijing    9.597      1357.00             5
SA         South Africa     Pretoria    1.221        52.98            12
```

INTERMEDIATE PYTHON FOR DATA SCIENCE

Let's practice!