02-iterations

November 4, 2022

1 Iterations

1.0.1 4/11/2022

[456]:

```
# Lists we're going to need

galaxy_names = ["NGC 5128", "TXS 0506+056", "NGC 1068", "GB6 J1040+0617", "TXS␣
  ↪2226-184"]
distances_mpc = [3.7, 1.75e3, 14.4, 1.51e4, 107.1]  # Mpc
luminosities = [1e40, 3e46, 4.9e38, 6.2e45, 5.5e41] # erg/s
```

1.1 Introducing range

[297]:

```
for i in range(5):
    print(i)
```

```
0
1
2
3
4
```

[295]:

```
for i in range(12,30,7): # start on 12, end on 29, steps of 7
    print(i)
```

```
12
19
26
```

1.1.1 Exercise - print name and distance of each galaxy in our list

[370]:

```
for i in range(len(galaxy_names)):
    print(f"Name: {galaxy_names[i]}; D = {distances_mpc[i]} Mpc")
```

```
Name: NGC 5128; D = 3.7 Mpc
Name: NGC 1068; D = 14.4 Mpc
Name: TXS 0506+056; D = 1750.0 Mpc
Name: GB6 J1040+0617; D = 15100.0 Mpc
Name: TXS 2226-184; D = 107.1 Mpc
```

1.1.2 More pythonic method - iterate directly over the list elements!

```
[360]: for pair in zip(distances_mpc, luminosities):
           print(pair)
```

```
(3.7, 1e+40)
(14.4, 4.9e+38)
(1750.0, 3e+46)
(15100.0, 6.2e+45)
(107.1, 5.5e+41)
```

```
[366]: for dist, lum in zip(distances_mpc, luminosities):
           print(dist, lum)
```

```
3.7 1e+40
14.4 4.9e+38
1750.0 3e+46
15100.0 6.2e+45
107.1 5.5e+41
```

1.1.3 Exercise - modify the printing code above to avoid using indices

```
[369]: for name, dist in zip(galaxy_names, distances_mpc):
           print(f"Name: {name}; D = {dist} Mpc;)
```

```
Name: NGC 5128; D = 3.7 Mpc; L = 5.5e+41 erg/s
Name: NGC 1068; D = 14.4 Mpc; L = 5.5e+41 erg/s
Name: TXS 0506+056; D = 1750.0 Mpc; L = 5.5e+41 erg/s
Name: GB6 J1040+0617; D = 15100.0 Mpc; L = 5.5e+41 erg/s
Name: TXS 2226-184; D = 107.1 Mpc; L = 5.5e+41 erg/s
```

1.1.4 And now a little cosmetic improvement using f-strings

```
[380]: for name, dist in zip(galaxy_names, distances_mpc):
           print(f"Name: {name:15}; D = {dist:10.1f} Mpc;")
```

```
Name: NGC 5128       ; D =        3.7 Mpc;
Name: NGC 1068       ; D =       14.4 Mpc;
Name: TXS 0506+056   ; D =     1750.0 Mpc;
```

```
Name: GB6 J1040+0617 ; D =    15100.0 Mpc;
Name: TXS 2226-184   ; D =      107.1 Mpc;
```

[382]:
```
for name, dist in zip(galaxy_names, distances_mpc):
    print(f"Name: {name:15}; D = {dist:.1e} Mpc;")  # extra points for␣
 ↪scientific notation
```

```
Name: NGC 5128       ; D = 3.7e+00 Mpc;
Name: NGC 1068       ; D = 1.4e+01 Mpc;
Name: TXS 0506+056   ; D = 1.8e+03 Mpc;
Name: GB6 J1040+0617 ; D = 1.5e+04 Mpc;
Name: TXS 2226-184   ; D = 1.1e+02 Mpc;
```

1.1.5 Simplifying counting with enumerate

[346]:
```
list(enumerate(galaxy_names))
```

[346]:
```
[(0, 'NGC 5128'),
 (1, 'NGC 1068'),
 (2, 'TXS 0506+056'),
 (3, 'GB6 J1040+0617'),
 (4, 'TXS 2226-184')]
```

[373]:
```
for i, name in enumerate(galaxy_names):
    print(f"Position: {i}; Name: {name}")
```

```
Position: 0; Name: NGC 5128
Position: 1; Name: NGC 1068
Position: 2; Name: TXS 0506+056
Position: 3; Name: GB6 J1040+0617
Position: 4; Name: TXS 2226-184
```

1.2 Creating lists

Exercise: convert distance list from Mpc to cm

[393]:
```
distances_cm = []
for d in distances_mpc:
    distances_cm.append(d * 3e24)

print(distances_cm)
```

```
[1.11e+25, 5.25e+27, 4.32e+25, 4.53e+28, 3.213e+26]
```

Exercise: select distances < 100 Mpc and convert them to cm

[391]:

```
# Exercise: convert distance list from Mpc to cm

short_distances_cm = []
for d in distances_mpc:
    if d < 100:
        short_distances_cm.append(d * 3e24)

print(short_distances_cm)
```

```
[1.11e+25, 4.32e+25]
```

1.2.1 Introducing list comprehension!

[395]:

```
distances_cm = [d * 3e24 for d in distances_mpc]

print(distances_cm)
```

```
[1.11e+25, 5.25e+27, 4.32e+25, 4.53e+28, 3.213e+26]
```

[397]:

```
# We can also select elements based on some criterium on the same one line:

short_distances_cm = [d * 3e24 for d in distances_mpc if d < 100.]
print(short_distances_cm)
```

```
[1.11e+25, 4.32e+25]
```

Exercise: get list of names based on distance critrion

[399]:

```
closeby_galaxy_names = [name for name, dist in zip(galaxy_names, distances_mpc) if dist < 100 ]

print(closeby_galaxy_names)
```

```
['NGC 5128', 'NGC 1068']
```

1.3 Counting

[403]:

```
# By building a list and checking its length:

print(len(closeby_galaxy_names))
```

```
2
```

[402]:

```
# Or better - if you don't need the list you don't have to create it

count = 0
```

```
for dist in distances_mpc:
    if dist < 100:
        count += 1
print(count)
```

```
2
```

1.4 Simultaneously iterating through multiple lists

[362]:

```
from math import pi

fluxes = []
for lum, d_mpc in zip(luminosities, distances_mpc):
    d_cm = d_mpc * 3e24
    fluxes.append(lum / (4 * pi * d_cm ** 2))

print(fluxes[1])
```

```
2.089386202070171e-14
```

[415]:

```
# Exercise - do the same using list comprehension!
fluxes = [lum / (4 * pi * (d_mpc * 3e24) ** 2) for lum, d_mpc in␣
 ↪zip(luminosities, distances_mpc)]
print(fluxes[3])
```

```
2.404282090867728e-13
```

1.5 Iterating thorugh tables with nested loops

1.5.1 Example - calculate a 2D table of fluxes based on the luminosities and distances

[416]:

```
from math import pi

flux_table = []
for lum in luminosities:
    flux_table.append([])
    for d_mpc in distances_mpc:
        d_cm = d_mpc * 3e24
        flux_table[-1].append(lum / (4 * pi * d_cm ** 2))

print(flux_table[3][3])
```

```
2.404282090867728e-13
```

1.5.2 Exercise - use list comprehension to rewrite the function in only one line!

[420]:

```
table = [[lum / (4 * pi * (d_mpc * 3e24) ** 2) for lum in luminosities] for␣
 ↪d_mpc in distances_mpc]

print(table[3][3])
```

```
2.404282090867728e-13
```

1.6 The `break` statement

[254]:

```
galaxy_names

i = 0

for i, name in enumerate(my_list):
    print(my_list[i])
    if (my_list[i] == 'Guru'):
        print('Found the name Guru')
        break
```

```
Siya
Tiya
Guru
Found the name Guru
After while-loop exit
```

Think, pair, share: breaks in nested loops

[268]:

```
# What's the output of the following code?

for i in range(4):
    for j in range(4):
        if j == 2:
            break
        print(f"{i} and {j}");
```

```
0 and 0
0 and 1
1 and 0
1 and 1
2 and 0
2 and 1
3 and 0
3 and 1
```

1.7 The continue statement

[261]:
```
for i in range(10):
    if not i % 2:
        continue
    print(f"{i} is odd")
```

```
1 is odd
3 is odd
5 is odd
7 is odd
9 is odd
```

Think, pair, share: breaks in nested loops

[275]:
```
# What's the output of the following code?

for i in range(4):
    if i < 2:
        continue
    for j in range(4):
        print(f"{i} and {j}");
```

```
2 and 0
2 and 1
2 and 2
2 and 3
3 and 0
3 and 1
3 and 2
3 and 3
```

1.8 From lists to dictionaries

1.8.1 Exercise: create dictionary mapping galaxy_name to luminosity

[424]:
```
galaxy_luminosities = {}

for name, lum in zip(galaxy_names, luminosities):
    galaxy_luminosities[name] = lum

print(galaxy_luminosities["TXS 0506+056"])
```

```
3e+46
```

A more pythonic way

```
[429]: galaxy_luminosities = {name:lum for name, lum in zip(galaxy_names,␣
        ↪luminosities)}

       print(galaxy_luminosities["TXS 0506+056"])
```

```
3e+46
```

An even more pythonic way

```
[430]: galaxy_luminosities = dict(zip(galaxy_names, luminosities))

       print(galaxy_luminosities["TXS 0506+056"])
```

```
3e+46
```

1.9 Iterate through dictionaries

```
[439]: for k in galaxy_luminosities:
           print(f"{k:15s} has {galaxy_luminosities[k]:.2e} erg/s ")
```

```
NGC 5128        has 1.00e+40 erg/s
TXS 0506+056    has 3.00e+46 erg/s
NGC 1068        has 4.90e+38 erg/s
GB6 J1040+0617  has 6.20e+45 erg/s
TXS 2226-184    has 5.50e+41 erg/s
```

More pythonic:

```
[440]: for k, v in galaxy_luminosities.items():
           print(f"{k:15s} has {v:.2e} erg/s ")
```

```
NGC 5128        has 1.00e+40 erg/s
TXS 0506+056    has 3.00e+46 erg/s
NGC 1068        has 4.90e+38 erg/s
GB6 J1040+0617  has 6.20e+45 erg/s
TXS 2226-184    has 5.50e+41 erg/s
```

Exercise: create a dictionary mapping galaxy names to their observed flux

```
[452]: d1 = {name : lum / (4 * pi * (d * 3e24) ** 2) for name, lum, d in␣
        ↪zip(galaxy_names,
                                                                                ␣
        ↪luminosities,
                                                                                ␣
        ↪distances_mpc) }
       print(d1["GB6 J1040+0617"])
```

```
2.404282090867728e-13
```

1.9.1 Last exercise

Create a nested dictionary in the form

```
galaxy_catalog = {galaxy_name1 : {'lum': luminosity1,
                                  'dist': distance1,
                                  'flux': flux1},
                  galaxy_name2 : {'lum': luminosity2,
                                  'dist': distance2,
                                  'flux': flux2},
                 #...
                 }
```

[ ]: