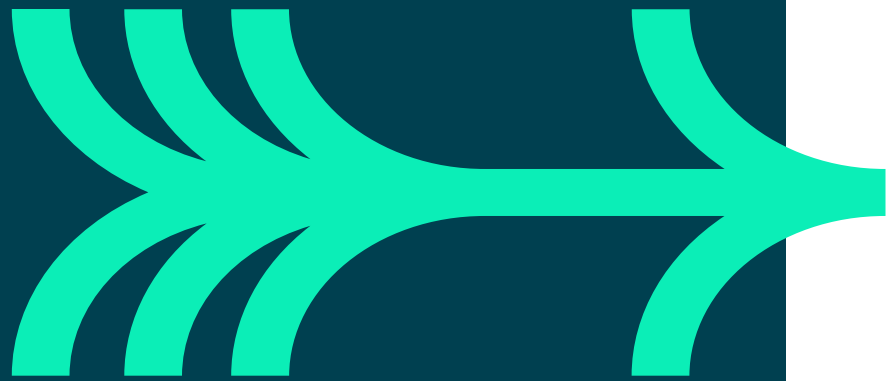
Advanced collections



The Topic: What?

•Interesting things you can do with collections

- Filtering Data
- Copying Data

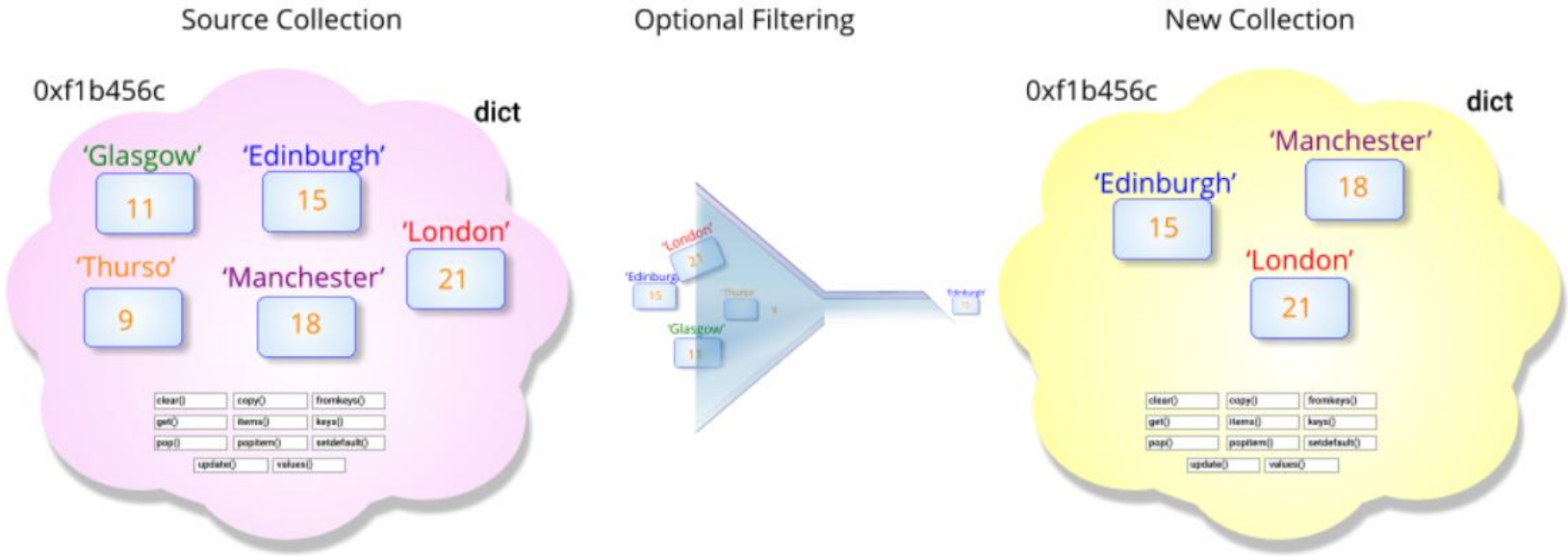
Applications: Why?

•To expand on your knowledge of collections and how they can be used.

Expectations: Who?

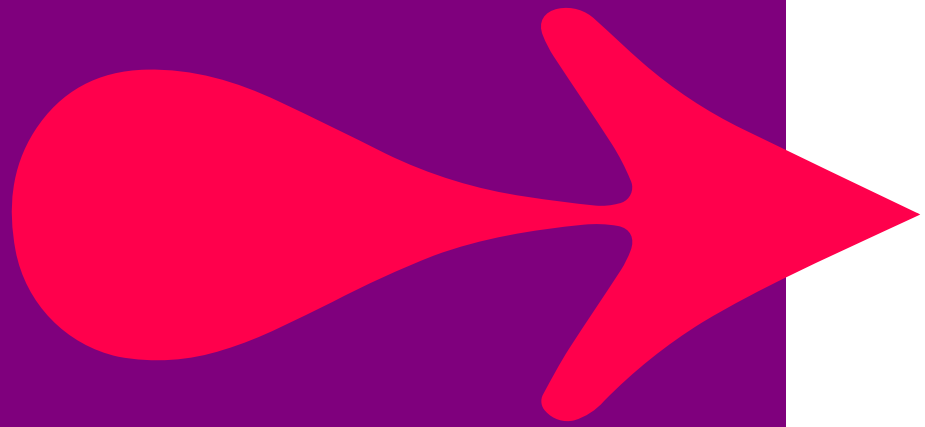
•Learners are expected to have covered tuples, lists, dictionaries, and sets in Python previously.

Filtering


Source Collection
Optional Filtering
New Collection
0xf1b456c
dict
'Glasgow'
11
'Edinburgh'
15
'Thurso'
9
'Manchester'
18
'London'
21
clear()
copy()
fromkeys()
get()
items()
keys()
pop()
popitem()
setdefault()
update()
values()
0xf1b456c
dict
'Manchester'
18
'Edinburgh'
15
'London'
21

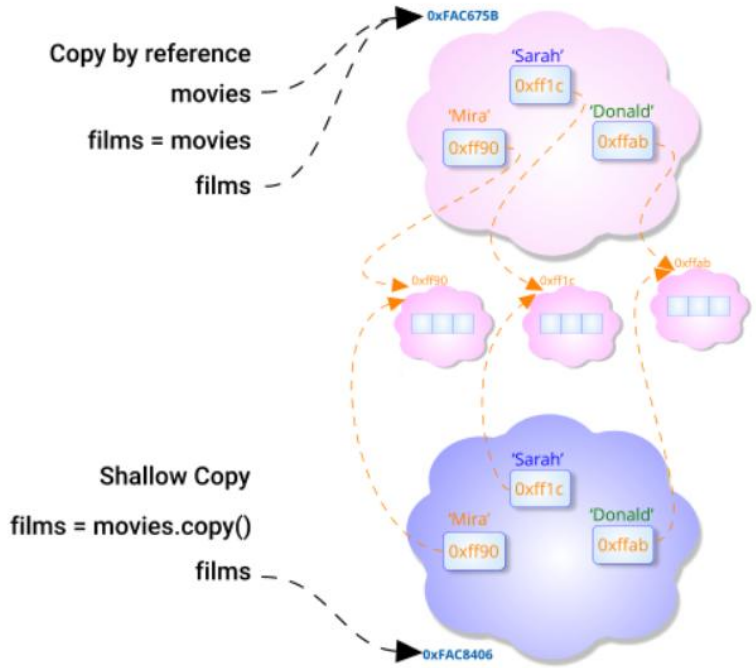
Filtering

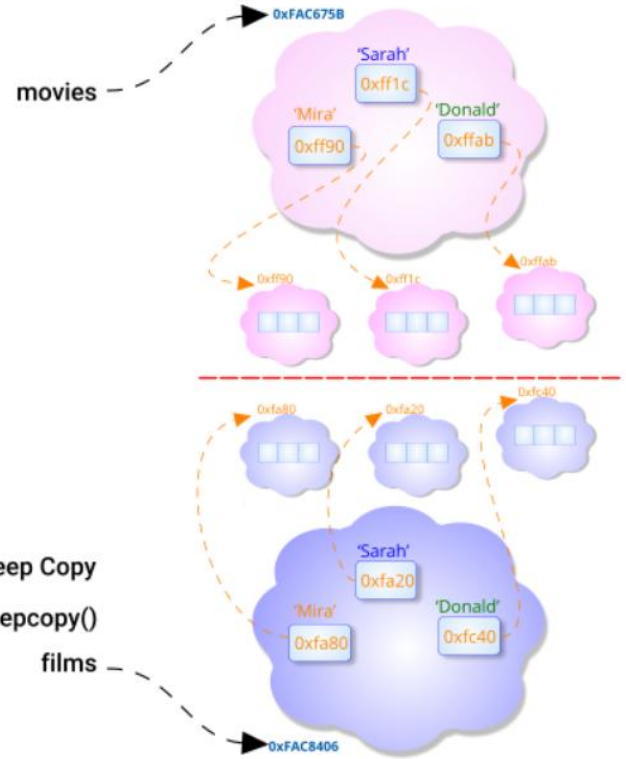
Trainer demonstration



demo_collections_comprehensions.py

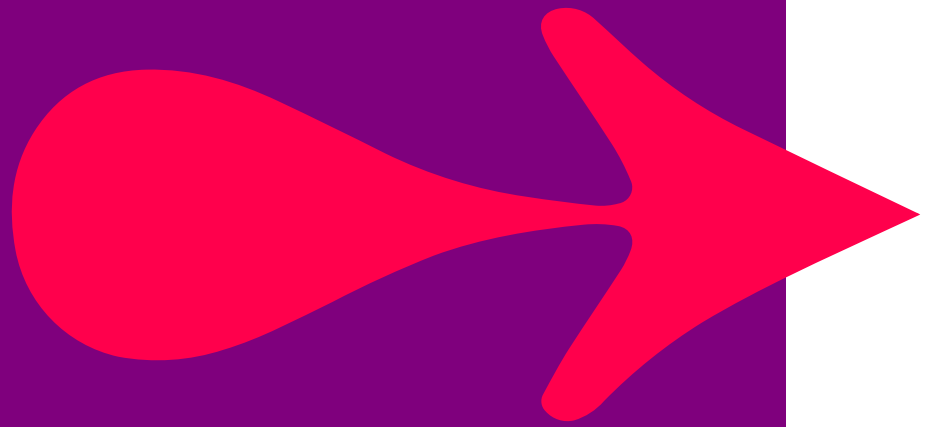
Copying


Copy by reference
movies
films = movies
films
0xFAC675B
'Sarah'
0xff1c
'Mira'
0xff90
'Donald'
0xffab
Shallow Copy
films = movies.copy()
films
0xFAC8406


movies
0xFAC675B
'Sarah'
0xff1c
'Mira'
0xff90
'Donald'
0xffab
Deep Copy
films = copy.deepcopy()
films
0xfa80
0xfa20
0xfc40
0xFAC8406

Copying

Trainer demonstration



demo_collections_copying.py

Learning check

5-10 mins

Quiz!

1. Can you think of a better way to write the following pattern, which is 15 chars long?

```
>>> [ num for num in range(0, 10) if num % 2 == 0 and num % 4 == 0]
```

2. Can you think of a better way to write the following pattern, which is 15 chars long?

```
>>> gumbys = ['eric', 'michael', 'terry', 'john', 'terry', 'graham']
>>> { name for name in gumbys if len(name) <= 5 }
```

3. Can you think of a better way to write the following pattern, which is 15 chars long?

```
>>> movies = {'Cruella': '2021', 'Mulan': '2020', 'Mary Poppins Returns': '2018'}
>>> films = movies.copy()
>>> movies['Mulan'] = 1998
>>> films == movies
```

Solutions

Advanced Collections quiz

1. What type of comprehension is this, and what would it return?

```
>>> [ num for num in range(0, 10) if num % 2 == 0 and num % 4 == 0]
```

Answer: [0, 4, 8] # List Comprehension

2. How many Gumbies (might have to google that!) would be displayed?

```
>>> gumbies = ['eric', 'michael', 'terry', 'john', 'terry', 'graham']
>>> { name for name in gumbies if len(name) <= 5 }
```

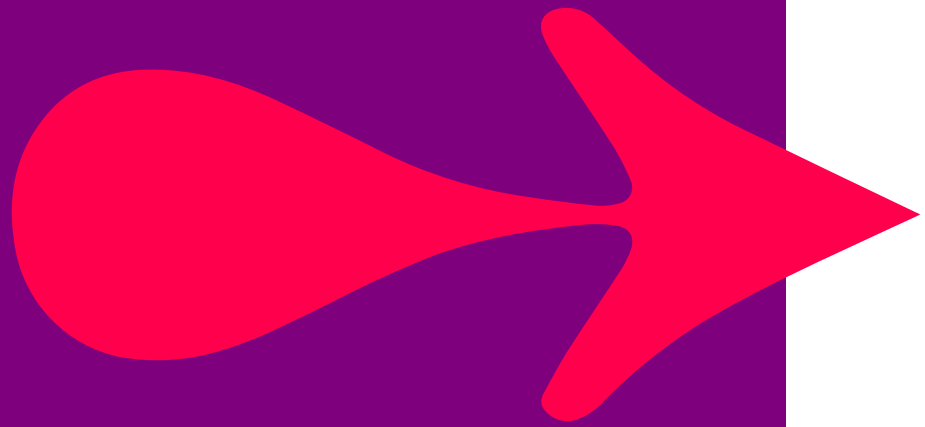
Answer: {'eric', 'terry', 'john'} # 3 Displayed

3. Would this sequence of commands return True or False

```
>>> movies = {'Cruella': '2021', 'Mulan': '2020', 'Mary Poppins Returns': '2018'}
>>> films = movies.copy()
>>> movies['Mulan'] = 1998
>>> films == movies
```

Answer: False # Shallow Copy. Movies was changed after the copy.

Labs



1. You should be familiar with the built-in range() function that generates a sequence of integers from a start point to a stop point with an optional step increment.

range(stop) # default start=0, step=1

range(start, stop[, step])

Unfortunately, it only works with integers. In this exercise, we will create our own simple version of the built-in range() function but for floating point numbers. Create a new script 'C:\labs\gen.py' with a generator function called frange() which accepts at least two parameters (start, stop) and an optional parameter with default (step=0.25)). Be wary of the possibility of a step = zero being passed in!

frange(start, stop[, step=0.25])

Test with the following calls in main():

print(list(frange(1.1, 3)))

print(list(frange(1, 3, 0.33)))

print(list(frange(1, 3, 1))) # Should print [1.0, 2.0].

print(list(frange(3, 1))) # Should print an empty list.

print(list(frange(1, 3, 0))) # Should print an empty list.

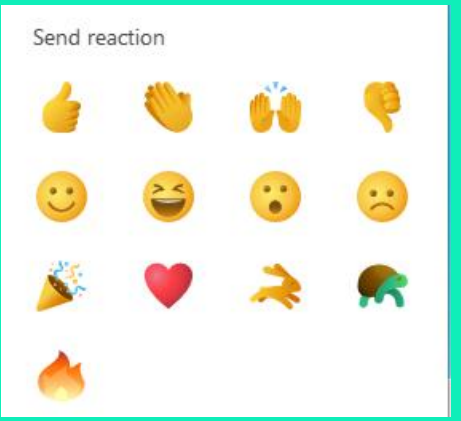
print(list(frange(-1, -0.5, 0.1)))

print(frange(1, 2)) # Should print <generator object frange at 0x....>

for num in frange(3.142, 12):

Stretch Exercises 2-3

END OF SECTION



Interesting things you can do with collections

- **Filtering Data**
- **Copying Data**

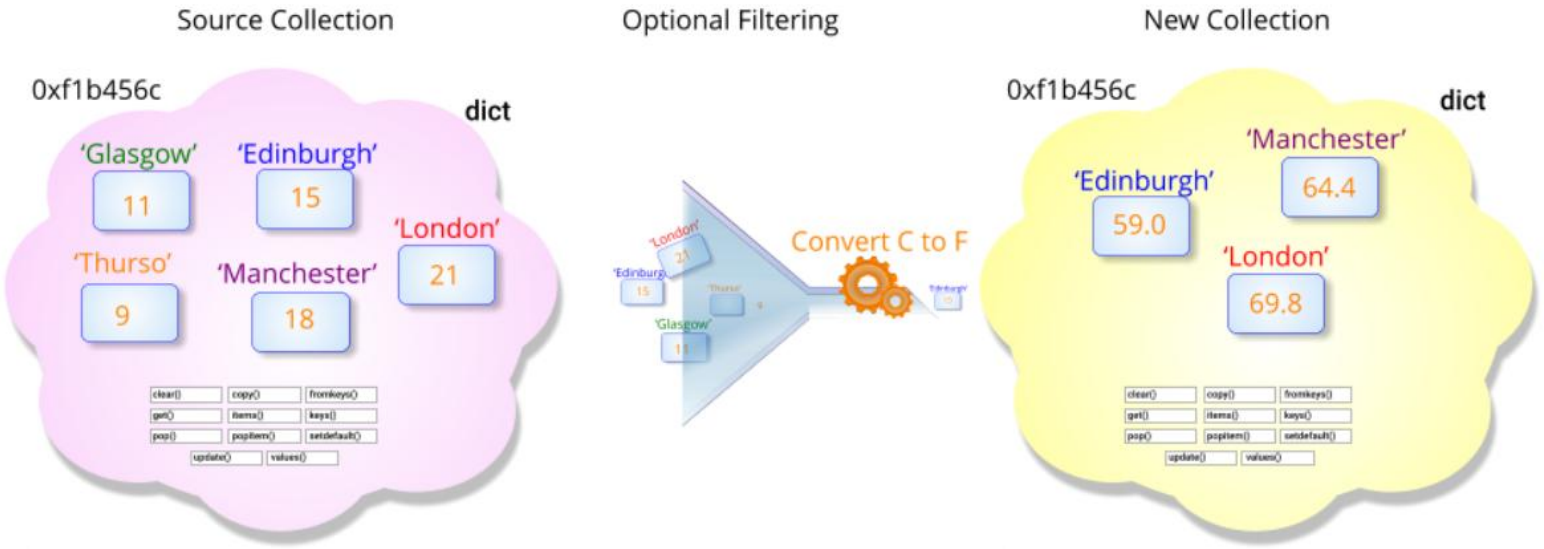
- **To expand on your knowledge of collections and how they can be used.**

Extension materials

Filtering+

Filtering+

- demo_collections_comprehensions_plus_c2f.py

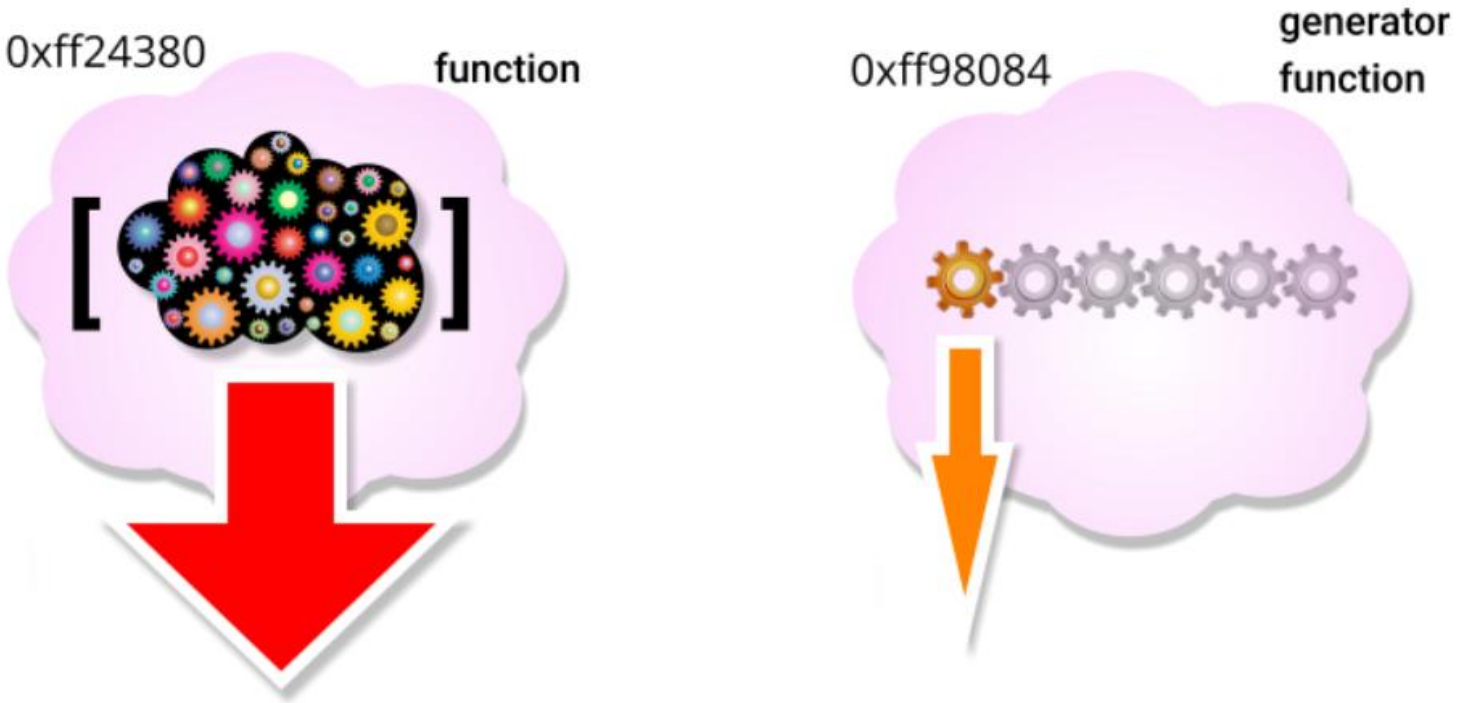


Extension materials

Generating

Filtering+

- demo_collections_generator_functions.py



REMINDER: TAKE A BREAK!

10.30 - 10.40
11.40 – 11.50
12.50 – 13.30
14.30 – 14.40
15.40 – 15.50