COMP 110

for loops + range()

Looping Through Sequences

- We commonly use loops to iterate over every element in a sequence.

for … in … loops

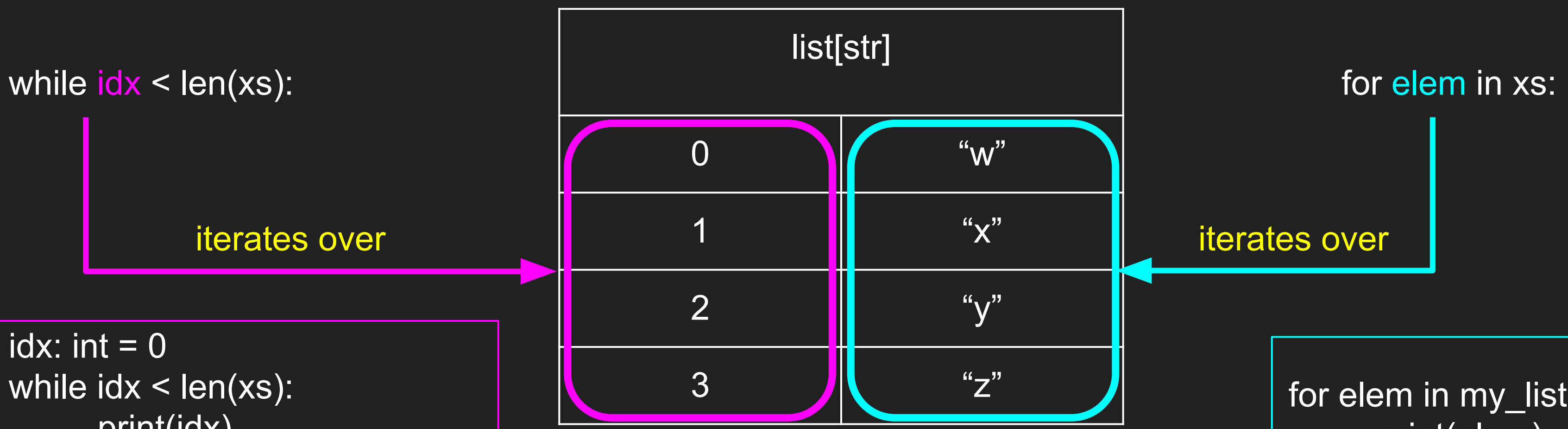
xs: list[str] = [“w”, “x”, “y”, “z”]

Print every element of xs

while

for … in …

```
xs: list[str] = [“w”, “x”, “y”, “z”]
```



```
idx: int = 0
while idx < len(xs):
        print(idx)
        idx += 1

Output:
                  0
                  1
                  2
                  3
```

```
for elem in my_list:
        print(elem)

Output:
            w
            x
            y
            z
```

for … in … loops in Memory

```python
"""Practice of for Loops"""

my_list: list[str] = ["hello", "world"]
new_list: list[str] = []
for elem in my_list:
    new_list.append(elem)
print(new_list)
```

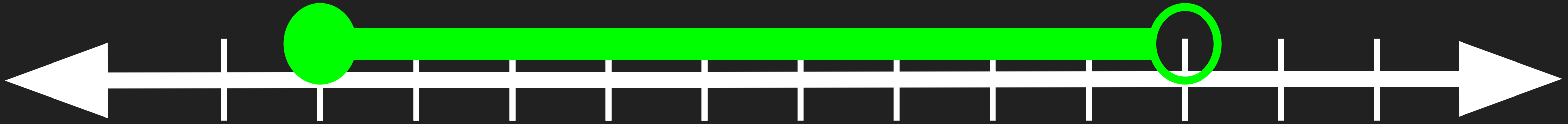
Writing for loops

- Let's implement a function named celebrate that we can call with 1 argument:
    - A pet_names: list[str] that stores the name of pets
- The return value of the function is None
- The function should use a for … in loop to print this string for every pet: “Good boy, <pet>!”
- Example:

```
>>> pets: list[str] = ["Louie", "Bo", "Bear"]
>>> celebrate(pets)
Good boy, Louie!
Good boy, Bo!
Good boy, Bear!
```

Why “while” loop over “for” loop?

Range

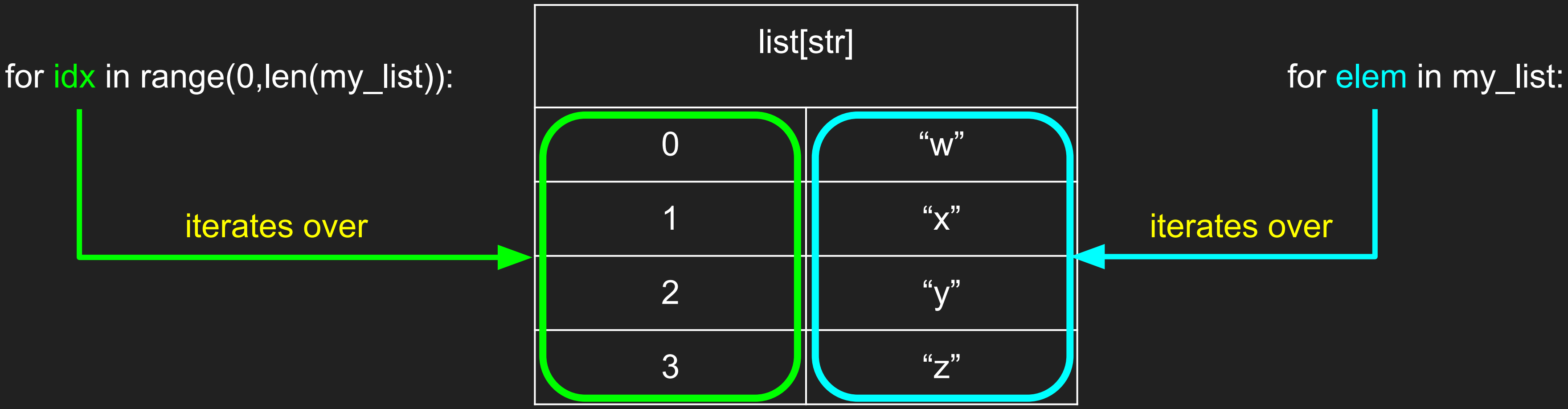


- A type of sequence you can loop over.
- Includes start point, does **not** include end point, and *steps* through every point in between
- Constructor: range(start, end, [step = 1])
- Examples:
  - range(1, 5) stops at numbers 1, 2, 3, 4
  - range(1, 6, 2) stops at numbers 1, 3, 5

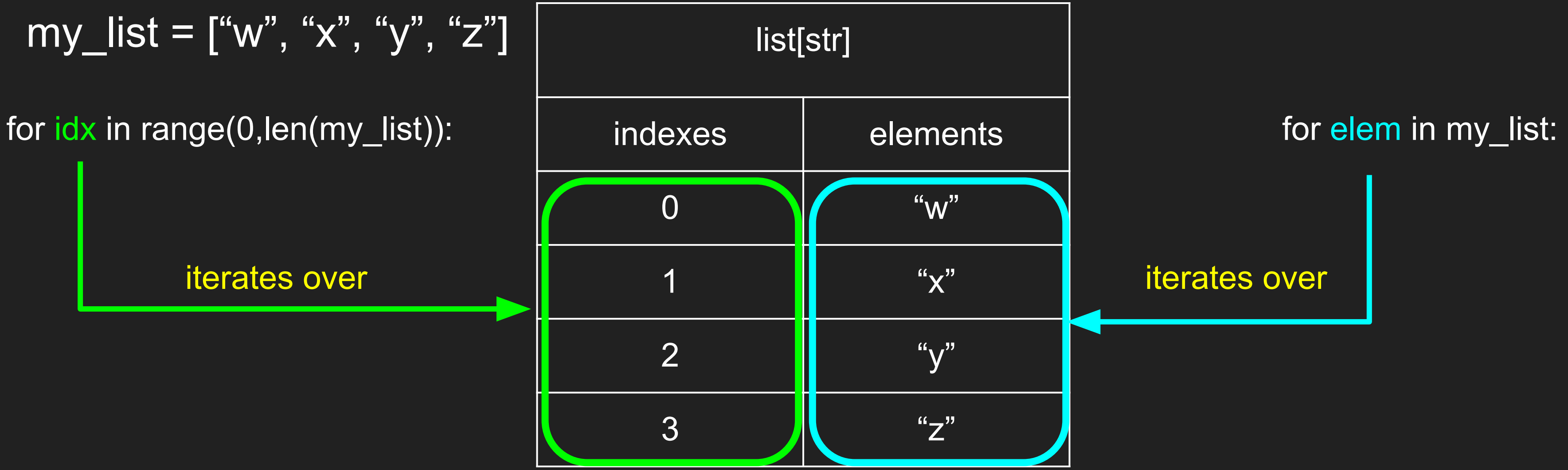
range() in Memory

On the heap, but don’t worry about it. :-)

```
my_list = ["w", "x", "y", "z"]
```


for idx in range(0,len(my_list)):
iterates over
list[str]
0
1
2
3
"w"
"x"
"y"
"z"
for elem in my_list:
iterates over

```
my_list = [“w”, “x”, “y”, “z”]
```



```
for idx in range(0,len(my_list)):
        print(idx)

Output:
                    0
                    1
                    2
                    3
```

```
for idx in range(0,len(my_list)):
        print(my_list[idx])

Output:
                    w
                    x
                    y
                    z
```

```
for elem in my_list:
        print(elem)

Output:
                    w
                    x
                    y
                    z
```

Using range() in a for … in … loop.

names: list[str] = [“Alyssa”, “Janet”, “Vrinda”]

Print every element’s index *and* value:

```
0: Alyssa

1: Janet

2: Vrinda
```

“for” Loops + Dictionaries

“for” loops iterate over the *keys* by default

```
for key in ice_cream:
    print(key)
```

```
for key in ice_cream:
    print(ice_cream[key])
```

Flavor	Num Orders
“chocolate”	12
“vanilla”	8
“strawberry”	5

Practice

- Let's implement a function named get_orders that we can call with 1 argument:
  - A orders: dict[str, int] that stores ice cream flavors as the keys and the number of orders as the values
- The return value of the function is None
- The function should print this string for every order in the dictionary: “<flavor> has <number> orders”
- Example:

```
>>> ice_cream: dict[str, int] = {"chocolate": 12, "vanilla": 8, "strawberry": 5}
>>> get_orders(ice_cream)
chocolate has 12 orders.
vanilla has 8 orders.
strawberry has 5 orders.
```