Monica Tuttle
SP21 - Python 2

Comprehensions

List Comprehension: Definition

- Concise and convenient notation for creating new lists
- Consists of a variable assigned to a one-line expression enclosed by brackets
- Iterates over each element like a *for* loop

Ex. 1

Creating a list of integers

Without comprehension

```
In [1]: list1 = []
In [2]: for item in range(0, 8):
            list1.append(item)
In [3]: list1
Out [3]: [1, 2, 3, 4, 5, 6, 7]
```

Replaced by comprehension

```
In [1]: list1 = [item for item in range(0, 8)]
In [2]: list1
Out[2]: [0, 1, 2, 3, 4, 5, 6, 7]
```

Dictionary Comprehension: Definition

- Concise and convenient notation for generating new dictionaries
- Consists of variable assigned to one-line expression enclosed by curly braces

Ex. 1

Swapping key-value pairs

```
In [1]: ages = {'Sarah': 23, 'Samuel': 50, 'Amanda': 35}
In [2]: ages2 = {age: name for name, age in ages.items()}
In [3]: ages2
Out [3]: {23: 'Sarah', 50: 'Samuel', 35: 'Amanda'}
```

Ex. 2

Making a new dictionary

```
grades = {'Kit': [97, 85, 93], 'Thayer': [82, 78, 97]}
grades2 = {keys: sum(values) / len(values) for keys, values in grades.items()}
print(grades2)
```

```
{'Kit': 91.66666666666667, 'Thayer': 85.66666666666667}
```

Ex. 2 – *Mapping: operations within an expression*

```
list3 = [item**3 for item in range(1, 7)]
print(list3)
```

```
[1, 8, 27, 64, 125, 216]
```

Ex. 3 – *Filtering: if & if/else clauses*

```
list4 = [item for item in range(1, 11) if item % 2 == 0]
print(list4)
```

```
[2, 4, 6, 8, 10]
```

```
picking_petals = ["he loves me" if num % 2 == 1 else "he loves me not" for num in range(1, 7)]
print(picking_petals)
```

```
['he loves me', 'he loves me not', 'he loves me', 'he loves me not', 'he loves me', 'he loves me not']
```

Ex. 4 – *Processing a list's elements*

```
names = ['geraldine', 'antoine', 'theo', 'jordan', 'taylor'] # list in origi
print("Before:", names)
names = [item.capitalize() for item in names] # list comprehension
print("After: ", names)
```

```
Before: ['geraldine', 'antoine', 'theo', 'jordan', 'taylor']
After:  ['Geraldine', 'Antoine', 'Theo', 'Jordan', 'Taylor']
```

Ex. 5 – *Nesting*

```
nested_conditional = [i for i in range(12) if i % 2 == 0 if i % 3 == 0]
print(list1)
```

```
[0, 6]
```

```
nested_for_loop = [[i*j for j in range(1, 11)] for i in range(8, 10)]
print(nested_for_loop)
```

```
[[8, 16, 24, 32, 40, 48, 56, 64, 72, 80], [9, 18, 27, 36, 45, 54, 63, 72, 81, 90]]
```

Sources:
Deitel, Paul, and Harvey Deitel. *Intro to Python for Computer Science and Data Science*. Pearson Education, Inc., 2020.
Chen, Sitian. "Comprehension." 2019. PDF File. https://github.com/stellaapril/Python2/blob/master/tutorialspecs.pdf