Learn / Courses / Python Toolbox

Course Outline

Exercise

List comprehension over iterables

You know that list comprehensions can be built over iterables. Given the following objects below, which of these can we build list comprehensions over?

```
doctor = ['house', 'cuddy', 'chase', 'thirteen', 'wilson']
range(50)
underwood = 'After all, we are nothing more or less than what we choose to reve
jean = '24601'
flash = ['jay garrick', 'barry allen', 'wally west', 'bart allen']
valjean = 24601
```

Instructions

50 XP

Possible answers

- (●) You can build list comprehensions over all the objects except the string of number characters `jean`.
- ( ) You can build list comprehensions over all the objects except the string lists `doctor` and `flash`.
- ( ) You can build list comprehensions over all the objects except `range(50)`.
- ( ) You can build list comprehensions over all the objects except the integer object `valjean`.

Submit Answer

Take Hint (-15 XP)

IPython Shell

```
In [1]:
```

Question:

List comprehension over iterables: You know that list comprehensions can be built over iterables. Given the following objects below, which of these can we build list comprehensions over?

Correct Answer:

You can build list comprehensions over all the objects except the integer object 'valjean'.

Answer Code:

```
# Define the objects as shown in the image
doctor = ['house', 'cuddy', 'chase', 'thirteen', 'wilson']
range_50 = range(50)
underwood = "After all, we are nothing more or less than what we choose to reveal."
jean = '24601'
flash = ['jay garrick', 'barry allen', 'wally west', 'bart allen']
valjean = 24601

# Checking list comprehension compatibility:
# Strings can be iterated but the integer object 'valjean' is not iterable.
result = {
    'doctor': [name.upper() for name in doctor],  # Works
    'range_50': [x * 2 for x in range_50],        # Works
    'underwood': [char for char in underwood],    # Works
    'jean': [char for char in jean],              # Works
    'flash': [hero.upper() for hero in flash],    # Works
    'valjean': None                               # valjean is not iterable
}

# Print results for validation
for key, value in result.items():
    print(f"{key}: {'Works' if value is not None else 'Does not work'}")
```

Explanation:

1. List comprehensions can only work on iterable objects. Strings, lists, and ranges are iterable,
   while integers are not.
2. Each object was tested using a simple list comprehension.
3. 'doctor', 'range_50', 'underwood', 'jean', and 'flash' support list comprehensions as they are iterable.

4. 'valjean', being an integer, does not support list comprehensions and returns None.