LECTURE 28

OBJECT-ORIENTED PROGRAMMING 4

PROTOCOLS

MCS 260 Fall 2021

David Dumas

REMINDERS

- Work on Project 3. Do not delay!
- We'll talk about Project 3 a bit today.

GOALS

- Introduce the **sequence protocol** for Python classes
- Work on an example
- Discuss other Python protocols

All of this is Python-specific. Some other languages use the term *interface* for a similar concept, though the details differ quite a bit.

WHAT IS A SEQUENCE?

In Python, a sequence is an ordered container supporting access to its items by 0-based index.

Things you can do with a sequence `seq`:

- `len(seq)`
- `seq[3]`
- `seq[3] = val` (only available if mutable)
- `for item in seq:`

CUSTOM SEQUENCE

If you create a class with the following methods, it can be used as a sequence:

- **`__len__()`**
  - The return value determines `len(obj)`
- **`__getitem__(idx)`**
  - The return value determines `obj[idx]`

These methods form the **sequence protocol**.

ITEM ASSIGNMENT

To make a sequence mutable, add one more method:

- **`__setitem__(idx,val)`**
  - Will be called when `obj[idx]=val` is executed

code	becomes
`obj[1]`	`obj.__getitem__(1)`
`obj[1]=60`	`obj.__setitem__(1,60)`
`len(obj)`	`obj.__len__()`

```
for x in obj:
  # stuff
```

```
for i in range(len(obj)):
    x = obj[i]
    # stuff
```

GEOMETRIC SEQUENCE

A **geometric sequence** (or *geometric progression*) is a sequence of numbers where the ratio between neighboring terms is constant.

Infinite example: $1, 2, 4, 8, 16, 32, 64, \ldots$

Finite example: $5, 15, 45, 135, 405$

Non-example: $6, 8, 10, 12, 14$

GEOMETRIC SEQUENCE CLASS

Let's make a class `FGS` to represent a **f**inite **g**eometric **s**equence.

We'll keep track of `start`, `ratio`, and `length`.

Indexing will be used to request an item from the sequence, which will be computed when needed but not stored.

ITEM ASSIGNMENT

Let's support item assignment with `__setitem__`.

Adopt these conventions:

- Assigning index 0 changes `start`
- Assigning any other index keeps `start` the same but adjusts `ratio`

OTHER PROTOCOLS

- Iterator — creates an iterable
- Mapping — creates a dict-like type

Still more can be found in the collections.abc module, which contains classes you can subclass when implementing the protocols.

REFERENCES

- In *Downey*:
  - Sequence, iterator, and other protocols are not discussed in the text.
  - Chapter 17 discusses the basics of object-oriented programming in Python.
- Object-oriented programming is also discussed in Section 6.5 of Brookshear & Brylow.

REVISION HISTORY

- 2021-10-27 Initial publication