LECTURE 16: CLASS AND FUNCTIONS

Hung-Yu Wei

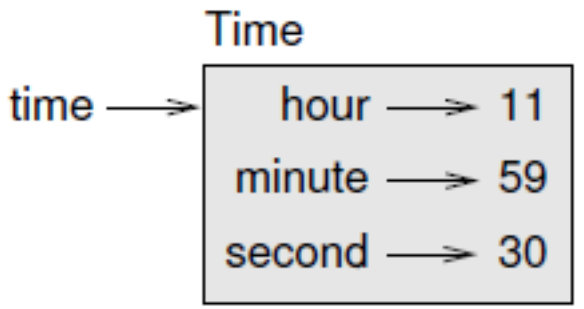
Time Class

- Class
  - *Time*
- Attributes
  - *Hour*
  - *Minute*
  - *Second*

```
class Time:
    """Represents the time of day.

    attributes: hour, minute, second
    """
```

```
time = Time()
time.hour = 11
time.minute = 59
time.second = 30
```



Time Class

- Function
  - *Print_time*
    - %.2d
    - Integer with 2 digits
  - *Is_after*
    - True of t1 is after t2

```
def print_time(t):
    """Prints a string representation of the time.

    t: Time object
    """
    print('%.2d:%.2d:%.2d' % (t.hour, t.minute, t.second))
```

```
def is_after(t1, t2):
    """Returns True if t1 is after t2; false otherwise."""
    return (t1.hour, t1.minute, t1.second) > (t2.hour, t2.minute, t2.second)
```

Design philosophy

- Prototype and patch
  - *且戰且走*
- Designed development

- Implement Time class with these 2 approaches

Pure functions

- Pure function
  - *Do not modify objects that are passed to the function*
- Modifiers
  - *Modify objects that passed as parameters*

- Suggestion
  - *Use pure functions (instead of modifier) as often as possible*
  - *Functional programming*

Pure function: add_time

```
def add_time(t1, t2):
    sum = Time()
    sum.hour = t1.hour + t2.hour
    sum.minute = t1.minute + t2.minute
    sum.second = t1.second + t2.second
    return sum
```

```
>>> start = Time()
>>> start.hour = 9
>>> start.minute = 45
>>> start.second =  0

>>> duration = Time()
>>> duration.hour = 1
>>> duration.minute = 35
>>> duration.second = 0

>>> done = add_time(start, duration)
>>> print_time(done)
10:80:00
```

Need to fix it

add_time: revised version

```
def add_time(t1, t2):
    sum = Time()
    sum.hour = t1.hour + t2.hour
    sum.minute = t1.minute + t2.minute
    sum.second = t1.second + t2.second

    if sum.second >= 60:
        sum.second -= 60
        sum.minute += 1

    if sum.minute >= 60:
        sum.minute -= 60
        sum.hour += 1

    return sum
```

Is there any problem?

Modifier: increment()

- Modifiers
  - *Modify objects that passed as parameters*

```
def increment(time, seconds):
    time.second += seconds

    if time.second >= 60:
        time.second -= 60
        time.minute += 1

    if time.minute >= 60:
        time.minute -= 60
        time.hour += 1
```

Conversion between time and integer

```
def time_to_int(time):
    minutes = time.hour * 60 + time.minute
    seconds = minutes * 60 + time.second
    return seconds
```

(hour, minute, second) ➔ seconds

```
def int_to_time(seconds):
    time = Time()
    minutes, time.second = divmod(seconds, 60)
    time.hour, time.minute = divmod(minutes, 60)
    return time
```

Rewrite add_time()

```
def add_time(t1, t2):
    seconds = time_to_int(t1) + time_to_int(t2)
    return int_to_time(seconds)
```

add_time: revised version 2

- Change it to integer (seconds)
- Change it back when it's done

```
def add_times(t1, t2):
    seconds = time_to_int(t1) + time_to_int(t2)
    return int_to_time(seconds)
```

Debugging

```
def valid_time(time):
    if time.hour < 0 or time.minute < 0 or time.second < 0:
        return False
    if time.minute >= 60 or time.second >= 60:
        return False
    return True
```

- Checking for invariant
  - *There are some conditions that should always be True (Check them)*
- Assert statement
  - *Raise an exception when it fails*
  - *Useful to check errors*
  - *Syntax*
    - Assert

```
def add_time(t1, t2):
    if not valid_time(t1) or not valid_time(t2):
        raise ValueError('invalid Time object in add_time')
    seconds = time_to_int(t1) + time_to_int(t2)
    return int_to_time(seconds)
```

```
def add_time(t1, t2):
    assert valid_time(t1) and valid_time(t2)
    seconds = time_to_int(t1) + time_to_int(t2)
    return int_to_time(seconds)
```

Reading

- Chapter 16 in textbook “Think Python”

- Summary: Time class
  - *Attributes (2 internal structures)*
    - Hour, minute, second
    - Integer (counting in seconds)
  - *Functions*
    - print_time()
    - is_after ()
    - add_time ()
    - time_to_integer ()
    - integer_to_time ()
    - valid_time()