CSC110 Lecture 4: Function Design

Hisbaan Noorani

September 21, 2020

Contents

1 Ex 1: Function design practice

1. Consider the code nippets below. For each code snippet, complete the docstring and/or docstring examples. Use the type contracts, functions name, parameter names, and function body to help you.

```python
def check_lengths(strings: list, max_length: int) -> bool:
    """Return whether all the strings in the set are shorter than the provided max_length

    >>> check_lengths(['cat', 'no', 'maybe'], 4)
    False
    >>> check_lengths(['hello', 'my', 'name', 'is', 'Hisbaan'], 8)
    True
    """
    return max([len(x) for x in strings]) < max_length


def string_lengths(strings: list) -> dict:
    """Return a dictionary mapping a string to it's length

    >>> string_lengths(['aaa', 'david'])
    {'aaa': 3, 'david': 5}
    >>> string_lengths(['one', 'two', 'three'])
    {'one': 3, 'two': 3, 'three': 4}
    """
    return {s: len(s) for s in strings}
```

2. For each of the following descriptions, write a Python function to perform the task described, using the Function Design Recipe.

   (a) Given a `float` representing the price of a product and another `float` representing a tax rate (e.g., a 13% tax rate represented as the `float` value `0.13`), calculate the after-tax cost of the product.

```python
def calculate_item_cost(price: float, tax_rate: float) -> float:
    """Return the price of an object after a tax is applied

    >>> price_with_tax(10.0, 0.13)
    11.3
    """
    return price + (price * tax_rate)
```

(b) Given a dictionary mapping names of products (as strings) to prices (as floats), which represents a customer order at a store, and a tax rate, calculate the total after-tax cost of the products.

Hint: the following illustrates how to use a dictionary as the "collection" in a comprehension.

```
>>> mapping = {1: 'a', 2: 'b', 3: 'c'}
>>> [k for k in mapping]  # Variable k is assigned to each keys in mapping
[1, 2, 3]
>>> [mapping[k] for k in mapping]
['a', 'b', 'c']
```

```
def calculate_order_cost(prices: dict, tax_rate: float) -> float:
    """Return the total cost of all items in order after tax.

    >>> prices_after_tax({'milk': 3.99, 'eggs': 4.99, 'chips': 2.99}, 0.13)
    13.5261
    """
    return calculate_item_cost(sum(prices[item] for item in prices), tax_rate)
```

2 Ex 2: Function design practice, `math` edition

1. Use the Function Design Recipe to implement the following function: Given three side lengths of a triangle (as floats), calculate the angles in the triangle.

   Include an `import math` statement at the top of your Python file so that you can use definitions from math in your function implementation for this question.

   Hints:

   - The Cosine Law from trigonometry states that for a triangle with side lengths , , and , with angle opposite the side with length , the following equality holds:

   $$c^2 = a^2 + b^2 - 2ab\cos\theta$$

   - The sum of all angles in a triangle add up to 180 degrees, or Π
   - The `math` module has functions both for calculating the trignometric functions and the inverses: `sin` and `asin` (short for "arcsin"), `cos` and `acos`, etc.
   - It also has a variable `pi` you can access in your code as `math.pi`.

```
import math

def calculate_angles(side_a: float, side_b: float, side_c: float) -> list:
    """Return the angles of a triangle based on the given side lengths, a b and c.

    >>> calculate_angles(1, 1, math.sqrt(2))
    {45, 45, 90}
    """
    a_2 = side_a ** 2
```

```python
    b_2 = side_b ** 2
    c_2 = side_c ** 2

    angle_a = math.acos((b_2 + c_2 - a_2) / (2 * side_b * side_c))
    angle_b = math.acos((a_2 + c_2 - b_2) / (2 * side_a * side_c))
    angle_c = math.acos((a_2 + b_2 - c_2) / (2 * side_a * side_b))
    return [angle_a, angle_b, angle_c]
```

3 Additional Exercises

1. Given a set of strings, calculate the length of the longest string.

```python
def get_longest_string(strings: set) -> int:
    """Return the length of the longest string in the set.

    >>> get_longest_string({'hello', 'my', 'name', 'is', 'hisbaan'})
    7
    """
    return max({len(x) for x in strings})
```

2. You are renting a car to make a road trip accross Canada. The car rental company you plan to use charges a fee of $50 plus $15 per day you rent the car.

   Given the starting and ending dates of your trip (represented in Python as `datetime.date` values), calculate the total cost of renting a car. (**Note**: the start and end date are both counted in the rental cost.)

   For this function, you should read Section 2.4 Importing Modules for more inforamtion about the `datetime` module.