# Rectangle Exercise Solution

Hyungmo Gu

April 4, 2020

## Part 1

1. **Class Name:** Rectangle

   **One Line Summary:** A rectangle is defined by its top-left coordinates as well as its width and height.

```
Rectangle(10,20,300,400)

```

3. **Headers:**

   - translate_left(self, num):
   - translate_right(self, num):
   - translate_up(self, num):
   - translate_down(self, num):
   - is_equal(self, rect):
   - is_falling_within_another_rectangle(self, rect):
   - is_overlapping(self, rect):

```
class Rectangle:
    """A rectangle is defined by its top-left coordinates as well
as its width and height."""

    def translate_left(self, num):
        """Translate Rectangle to left by <num>
        @type self: Rectangle
        @type num: int
        @rtype: None
        >>> rect = Rectangle(10,20,300,400)
        >>> rect.translate_left(10)
        """
    def translate_right(self, num):
        """Translate Rectangle to right by <num>
```

```
            @type self: Rectangle
            @type num: int
            @rtype: None
            >>> rect = Rectangle(10,20,300,400)
            >>> rect.translate_right(10)
            """

        def translate_up(self, num):
            """Translate Rectangle to up by <num>
            @type self: Rectangle
            @type num: int
            @rtype: None
            >>> rect = Rectangle(10,20,300,400)
            >>> rect.translate_up(10)
            """

        def translate_down(self, num):
            """Translate Rectangle to down by <num>
            @type self: Rectangle
            @type num: int
            @rtype: None
            >>> rect = Rectangle(10,20,300,400)
            >>> rect.translate_down(10)
            """

        def is_equal(self, rect):
            """Return whether <rect> and <self> have the same
    coordinate and size
            @type self: Rectangle
            @type rect: Rectangle
            @rtype: bool
            >>> rect_1 = Rectangle(10,20,300,400)
            >>> rect_2 = Rectangle(10,20,300,400)
            >>> rect_3 = Rectangle(15,25,300,400)
            >>> rect_1.is_equal(rect_2)
            True
            >>> rect_1.is_equal(rect_3)
            False
            """

        def is_falling_within_another_rectangle(self, rect):
            """Return whether <self> is inside <rect>
            @type self: Rectangle
            @type rect: Rectangle
            @rtype: bool
            >>> rect_1 = Rectangle(10,20,300,400)
            >>> rect_2 = Rectangle(15,15,100,50)
            >>> rect_2.is_falling_within_another_rectangle(rect_1)
            True
            """

        def is_overlapping(self, rect):
            """Returns whether <self> has overlapping region with <
    rect>
```

```
66              @type self: Rectangle
67              @type rect: Rectangle
68              @rtype: bool
69              >>> rect_1 = Rectangle(10,20,300,400)
70              >>> rect_2 = Rectangle(0,0,300,400)
71              >>> rect_1.is_overlapping(rect_2)
72              True
73              """
74
```

**Notes:**

- What should be written for **@rtype** if nothing is returned?
- What should be written for **@type** if a parameter is of type class?
- What should be written for **@type** if a parameter is **self**?
- When writing an example, should class instantiation also be included like below?

```
1           def is_overlapping(self, rect):
2               """
3               ...
4               >>> rect_1 = Rectangle(10,20,300,400)
5               >>> rect_2 = Rectangle(0,0,300,400)
6               ...
7               """
8
```

# Part 2