[Session 4] Pygame Exercise

Note

You don't need to solve everything, because it is not mandatory and graded, just for practice! We will provide a solution after the real-time class. But, if you have some problem and ask us, then we will give you some help or hints. **Good luck!**

To make a game what you want, it is important to exercise and implement some common features of game. So please try the following exercises! Also, we will upload sample solution of some (EX) problems, as soon as possible.

Exercise 1

This problem set is for implementation of movement.

(1.1) Basic movement

Make a `Rect` object and **move this via keyboard** (up/down/left/right).

- Make the motion smoothly as much as possible
- Your Rect must not go beyond the window.

(1.2) Jump

Make a `Rect` object and **move this via keyboard** (left/right/spacebar).

- Make the motion smoothly as much as possible
- Your Rect must not go beyond the window.
- There should be **"gravity"**. It should fall onto the bottom side of window, eventually.
- Note that, falling is accelerated over time (jump, too).

(1.3) Wall

Use the solution of (1.1), and add several **walls**.

- Like the edge of window, we cannot pass there.
- we can use collision handling, we covered in the pre-class lecture.
- For multiple walls, how can we deal with these efficiently? (class? or something others?)

(1.4) Stair

Use the solution of (1.2) and (1.3), and add several **stairs**.

- We can jump and step on there.
- Although we can be on the stair, we cannot pass there.

(1.EX) Bouncy Ball

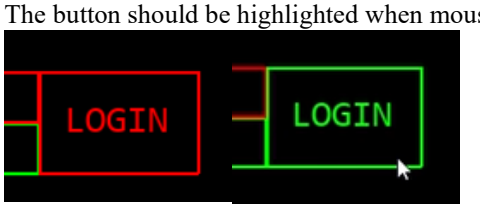
If you can implement all the problem. you can make a simple bouncy ball game. What is this? https://www.youtube.com/watch?v=_HkQQhmeOiA (Example game play video) But actually there are too many components. so, implement only this, basically. And then, if you can, try more!

- The ball bounces continuously.
- The ball can step on the block. If you fall into the bottom, then your ball will be died.
- The condition for "clear" can be vary. You can gather some star like the game, or you can get somewhere (destination).

Exercise 2

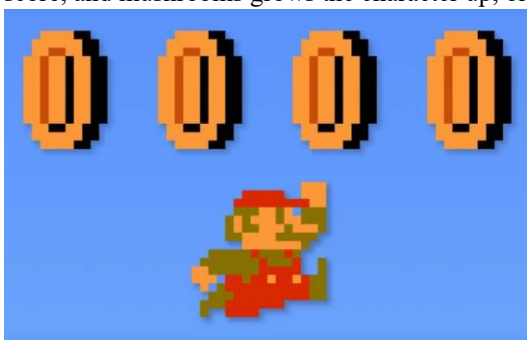
We need several kinds of user interface (UI). For example, you can click some button, or use trackbar(slider), or give some text input. For practice, we will make a simple button.

- Define a class, `Button`.
- It may have member variables like button color or image, some Rect for indicating region, and some text.
- The button should be highlighted when mouse-over.



Exercise 3

Items can affect the game and its state in many ways. For an example of **Super Mario Bros**, coins increase your score, and mushrooms grows the character up, or increases the life point.



It seems that we can implement some event when our character catches some item, by using **collision**, right? Note that, in most cases, item should vanish once it is caught. (i.e. we cannot reuse)

(3.1) Simple item event

Use the result of **(1.1)** and make a program as follows:

- **Set several items in advance. (not generated in run-time)**
- If your Rect collides with item, then print, **"Gotcha!"**
- After "Gotcha!", the item must **vanish**.

(3.2) Randomly generated item event

Use the result of **(3.1)** and make a program as follows:

- There is no item at the first, but **the program generates an item every 5 seconds**.
- The location where the item is generated is **random**. But it should be inside of window.
- Other things are same with (3.1).

Hint: For randomization, you can use, **`random`** library. Also, if you want to generate some number (integer) between two lower / upper bounds, you can use it:

```
import random
print(random.randint(0, 3))
```

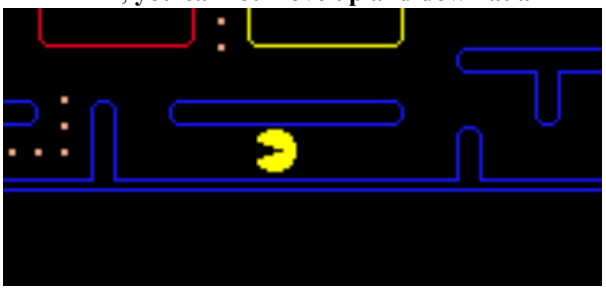
`random.randint(start, end)` function returns a random number between `start` and `end` (includes both `start` and `end`: `0, 1, 2, 3` in the example)

(3.EX) Simplified PACMAN

(You can pacman game here: https://www.google.com/logos/2010/pacman10-i.html)

Now, you can make a **simplified PACMAN game**. You can follow the instruction, but if you can, you can try to make your game more similar with the original game:

- **There is no enemy,** so you can freely gather the items!
- You can freely make a map, but the **character should fit to the wall everywhere**. It means that, in the below case, **you cannot move up and down at all**.



- Also, you cannot move in diagonal direction.
- I recommend to make a map small-sized
- If you gather all of item, then **print, "congratulations!"**, and terminate the game.