Домашнее задание 5

Постановка задачи

В этом домашнем задании требуется разработать версию игры «Пинг-понг» – одной из первых аркадных видео-игр (1972г.) [9]. Реализация этой игры позволит развить необходимые навыки для разработки игры «Астероиды», задание по которой вы получите позднее.

Чтобы получить представление о создаваемой игре, можно воспользоваться одной из её реализаций в web, например: http://libcanvas.github.io/games/pingpong/ (управление клавишами w-s и вверх-вниз). Ваша реализация будет в некоторых деталях отличаться от этой.

Для упрощения работы рекомендуется использовать шаблон pong_template.py.

Рекомендованный порядок работы над заданием

1. Добавьте код к шаблону программы, которая рисует мяч, движущийся по пинг-понг столу. Рекомендуется добавить обновление позиции мяча в обработчик рисования, как в примерах ball.py, velocity_control.py.

2. Добавьте код в функцию `spawn_ball()`, которая помещает мяч в середину игрального стола и присваивает мячу фиксированную скорость по Вашему выбору (на данный момент). Игнорируйте параметр `direction` на данном этапе.

3. Добавьте вызов функции `spawn_ball()` в функцию `new_game()`, которая запускает игру Пинг-понг. Обратите внимание, что шаблон программы включает первоначальный вызов `new_game()` в основной части вашей программы, чтобы запустить игру.

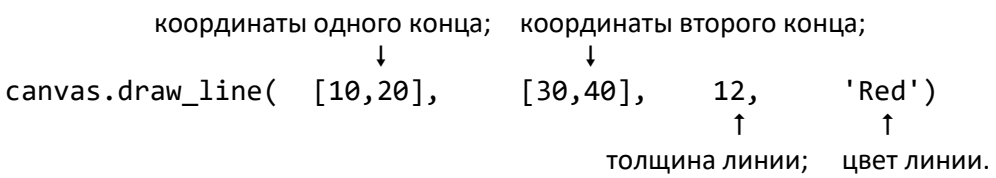
4. Измените существующий код так, чтобы при столкновении с нижней и верхней стенками мяч отскакивал от них. Протестируйте код с различными начальными скоростями мяча.

5. Добавьте случайный выбор скорости в `spawn_ball(direction)`. Скорость мяча должна быть направлена вверх и вправо, если `direction == RIGHT`, и вверх и влево, если `direction == LEFT`.
   Значения для горизонтальных и вертикальных компонентов скорости должны быть получены с использованием `random.randrange()`. Предполагается, что для горизонтальной скорости хорошо использовать `random.randrange(120, 240)` пикселей в секунду, для вертикальной скорости – `random.randrange(60, 180)` пикселей в секунду.

6. Добавьте в обработчика рисования код для проверки касания мяча с левым и правым бортами стола. Помните, что борта смещены от левого и правого краёв холста на ширину ракетки. При касании мячом бортика, используйте функции `spawn_ball(LEFT)` и `spawn_ball(RIGHT)`, чтобы установить мяч в центре части стола так, чтобы он двигался к противоположному борту.

7. Далее добавьте код, который рисует левую и правую ракетки соответствующих сторон. Вертикальные позиции обеих ракеток должны храниться в двух глобальных переменных. В шаблоне использовались переменные `paddle1_pos` и `paddle2_pos` (от англ. paddle – ракетка).

Подсказка: для рисования ракетки можно использовать функцию `canvas.draw_line()`, задав большую толщину линии:



8. Добавьте код, который изменяет значения вертикальных позиций ракеток, в обработчик рисования. Новая позиция ракеток должна зависеть от двух глобальных переменных, отвечающих за вертикальные скорости ракеток. В шаблоне такими переменными являются `paddle1_vel` и `paddle2_vel`.

9. Для обновления значений обеих вертикальных скоростей используйте обработчики клавиш. Клавиши `"w"` и `"s"` изменяют вертикальную скорость левой ракетки, клавиши `"стрелка вверх"` и `"стрелка вниз"` – скорость правой ракетки. Левая ракетка движется вверх и вниз с постоянной скоростью при нажатии `"w"` и `"s"` соответственно и остаётся неподвижной, если ни одна из клавиш не нажата. Для достижения такого результата следует использовать обработчики `KeyDown()` и `KeyUp()`, чтобы соответствующим образом изменять вертикальную скорость.

10. Чтобы ракетки не выходили за край холста, следует ограничить их движение, добавив проверку перед обновлением вертикальных позиций ракетки в обработчике рисования. В частности, следует проверить, не выходит ли за край экрана часть ракетки из-за текущего обновления её позиции. Если часть ракетки не видна, не производите данное обновление.

11. Измените код, контролирующий столкновение с левым и правым бортами (шаг 6) таким образом, чтобы осуществлялась проверка, что мяч ударился об ракетку, когда он касался бортика. Если это так, отразите мяч обратно на поле. Такая модель столкновения исключает возможность мяча попасть в ребро ракетки и значительно упрощает код, обрабатывающий столкновение/отражение.

12. Для усложнения игры увеличивайте скорость мяча на 10% каждый раз при отбивании ракеткой.

13. Добавьте ведение счета игры. Каждый раз, когда мяч попадает в левый или правый бортик и не попадает в ракетку, игрок напротив получает очко и мяч помещается в центр стола для разыгрывания.

14. Добавьте код в функцию `new_game()` для обнуления счёта перед вызовом функции `spawn_ball()` и добавьте кнопку «Перезапустить игру», которая вызывает `new_game()` для сброса счет и разыгрывания мяч.

Окончательный вариант игры достаточно сильно похож на оригинал аркадной игры Пинг-понг. Реализация игры на Python может занять чуть больше 100 строк кода с комментариями.

Оценка задания

Мяч корректно запускается в начале игры – 10 %.

Мяч корректно отражается от верхней и нижней стенок – 10 %.

Учитывается попадание в левый и правый край стола (а не в край холста), мяч перезапускается в сторону игрока, выигравшего последнее очко – 10 %.

Корректная работа с ракетками:

ракетки управляются клавишами w/s и вверх/вниз – 20 %;

ракетки рисуются вровень с краями поля и не выходят вверх/вниз – 20 %;

мяч отражается от ракеток – 20 %.

Подсчёт очков и перезапуск игры кнопкой – 10%.

В случае сдачи задания с опозданием, полученные за него баллы будут уменьшены:

при задержке на 1 неделю: баллы умножаются на 0.9;

при задержке на 2 недели: баллы умножаются на 0.75;

при задержке на 3 и более недель: баллы умножаются на 0.65.