бильярд

Автор проекта -- лапа петр

как я пришел к идее проекта

В современном мире мы все больше времени тратим на работу, и соответственно остается меньше времени на отдых и например игры.

Многие люди (около 80% населения) обожают играть в бильярд, но не все могут позволить себе идти куда-то для этого. Поэтому я решил создать симулятор бильярда, чтобы люди могли отдыхать в свое свободное время

модули, классы и технологии

PYGAME, SYS, МАТЕМАТИКА ПИТОНА

Самая сложная часть, по-моему была в написании функции отвечающей за столкновения шариков, но о ней будет позже.

Оставшиеся ~360 строк кода не имеют ничего сложного и особенного.

Классы Ball, Keel, Board

board

```
class Board:
    def __init__(self, window):
        self.window = window
        self.static = True

    def draw(self):
        window = self.window

        pygame.draw.rect(window, (90, 90, 0),
                         (50, 50, WIDHT - 100, HEIGHT - 100))
        pygame.draw.rect(window, (0, 128, 0),
                         (70, 70, WIDHT - 140, HEIGHT - 140))
        pygame.draw.circle(window, (0, 0, 0), (75, 75), 15)
        pygame.draw.circle(window, (0, 0, 0), (WIDHT - 75, 75), 15)
        pygame.draw.circle(window, (0, 0, 0),
                           (WIDHT - 75, HEIGHT - 75), 15)
        pygame.draw.circle(window, (0, 0, 0), (75, HEIGHT - 75), 15)
        pygame.draw.circle(window, (0, 0, 0), (WIDHT // 2, 70), 15)
        pygame.draw.circle(window, (0, 0, 0),
                           (WIDHT // 2, HEIGHT - 70), 15)
```

Здесь есть функция которая рисует всё поле (#НетСпрайтам)

я знаю, это больно

ball

```
6   class Ball:
7       colors = [(255, 255, 255),  # белый
8                 (200,   0,   0),  # красный
9                 (230, 230,   0),  # желтый
10                (0,   200,   0),  # зеленый
11                (120,  60,   0),  # коричневый
12                (0,     0, 250),  # синий
13                (250, 120, 150),  # розовый
14                (0,     0,   0)]  # черный
15
16      def __init__(self, window, type, coords):...
23
24      def draw(self):...
28
29      def update(self):...
71
72      def collision(self, ball, ball2):...
111
```

В этом классе рисуем, обновляем и сталкиваем шарики. Из интересных технологий могу отметить то, как я не позволяю шарику

```
if max(abs(self.vx), abs(self.vy)) > 3:
    k = 3 / max(abs(self.vy), abs(self.vx))
    self.vx, self.vy = k * self.vx, k * self.vy
```

Так я уменьшаю скорость, сохраняя направление

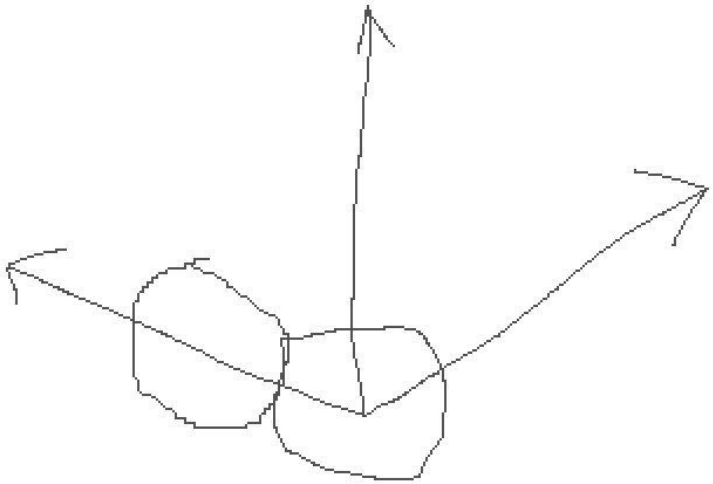
keel

```
class Keel:
    def __init__(self, screen):
        self.screen = screen

    def draw(self, coords, whiteball):
        pygame.draw.line(self.screen, (255, 255, 255),
                         (int(whiteball.coords[0]),
                          int(whiteball.coords[1])), coords, 3)

    def shoot(self, coords, whiteball):
        whiteball.vx = (whiteball.coords[0] - coords[0]) / 100
        whiteball.vy = (whiteball.coords[1] - coords[1]) / 100
        if max(abs(whiteball.vx), abs(whiteball.vy)) > 3:
            k = 3 / max(abs(whiteball.vx), abs(whiteball.vy))
            whiteball.vx = whiteball.vx * k
            whiteball.vy = whiteball.vy * k
        whiteball.vel = [whiteball.vx, whiteball.vy]
```

Тут тоже почти ничего нет.

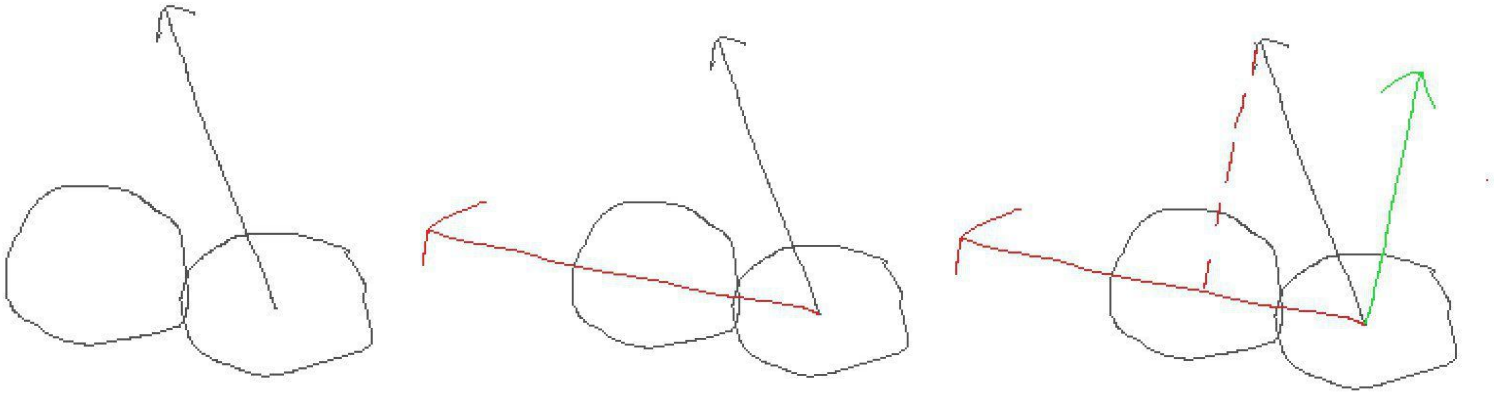
Отрисовка, удар

на пути к collision(ball, ball2)



Эта функция -- очень сложная задача, потому что не знаешь как к ней подступиться, сидя на олимпиаде по информатике, мне было нечего делать и я придумал эту картинку

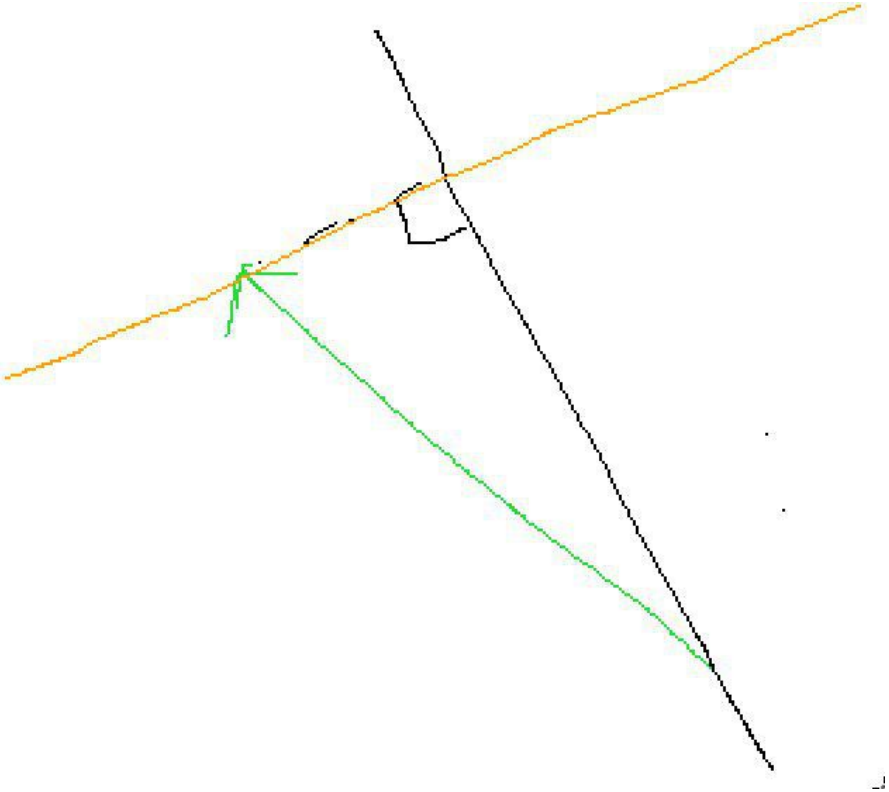
больше уродских картинок



Здесь видно прогрессию мысли.

На самом деле при столкновении (рис 1) скорости будут зеленая и отрезок красной стрелки, который будет катетом треугольника

алгеброй или по-людски?



Поняв, что нужно делать я встал перед вопросом, что же использовать, алгебру или геометрию, т е через координаты и пересечение прямых или через уголки и тригонометрию. Я думаю понятно, что я выбрал алгебру.

триумф человеческой мысли

```
def collision(ball, ball2):
    x1, y1 = ball.coords
    x2, y2 = ball2.coords
    black_k = (y1 - y2) / (x1 - x2)
    black_b = y1 - black_k * x1
    i = ball.vel[0] - ball2.vel[0] + x1
    j = ball.vel[1] - ball2.vel[1] + y1
    orange_k = -(x1 - x2) / (y1 - y2)
    orange_b = j - orange_k * i
    x = (orange_b - black_b) / (black_k - orange_k)
    y = orange_k * x + orange_b
    x, y = x - x1, y - y1
    x3, y3 = i - x, j - y
```

Эти 12 строк стоили мне многого. Здесь происходят пересечения прямых, вычисление перпендикуляра и конечного ответа

скриншоты


ВЫ ПРОИГРАЛИ ВАШ СЧЕТ
0
ПОЗДРАВЛЯЮ ВЫ ПОБЕДИЛИ
pygame window
ВАШ СЧЕТ 0
ВАШ СЧЕТ 2
ВАШ СЧЕТ 15
ЖЕЛТЫЙ

заключение

Хочу выразить благодарность моему сыну Андрею, жене Елизовете и моему учителю, Бояну Александровичу за моральную поддержку и идею создания проекта. Следуйте за своей мечтой! Дерзайте! И у вас все получится!

Отдельная благодарность ПАЧу за то что он существует.

Вывод: У меня все получилось, хейтеры проиграли. Доработки невозможны, программа идеальна!