**Змейка**

Для создания змейку создали 3 класса: Game(рисует поле и работает с ним), Snake(управляет и рисует змейку) и Food(рисует и следит за едой).

А теперь поподробнее:

Game:

В init’е сохранены цвета, чтобы не нужно было много писать, задается fps и указывается изначальное количество съеденной еды. В set\_surface\_and\_title задаем поверхность play\_surface, а также задаем название. В event\_loop отслеживаем нажатие клавиш игроком, возвращаем направление движения змейки. В refresh\_screen, как уже ясно из названия функции, обновляем экран и задаем fps(кстати, если вы измените fps, то будет изменяться скорость змейки). Благодаря функции show\_score отображаем результат, конструкции try и except используем, чтобы не выводило ошибок. В зависимости от содержимого choice, размещаем его в определенном месте, т.е. если choice = 1, то количество съеденной еды будет отображаться в левом верхнем окне, иначе при проигрыше оно высвечивается посередине. Функция game\_over выводит надпись проигрыша и результат. С помощью модуля time мы используем sleep, с помощью которого экран засыпает на 3 секунды, а затем выходит из программы.

Snake:

В init указаны такие переменные, как позиция головы змейки, начальное тело(состоит из 3 сегментов), цвет и начальное направление движения. В validate\_direction\_and\_change изменяем направление движения змейки и при том только тогда, когда оно не прямо противоположно текущему. В change\_head\_position изменяем положение головы. В snake\_body\_mechanism идет управление координат тела, проверяем поглощение еды и её новое расположение. В draw\_snake отображаем все сегменты змеи. check\_for\_boundaries нужна для того, чтобы проверять, столкнулась ли змейка о границы экрана или врезалась сама в себя.

Food:

В init сохраняем цвет и находим рандомные координаты еды, а draw\_food отображает её