Пояснительная записка

Описание функций:

1. Def load_image: обработка и получение спрайтов корабликов для игры.
2. Def get_coords: получение координат.
3. Def draw_point_success: отрисовка красной точки(попадание) на поле боя.
4. Def draw_point_unsuccess: отрисовка белой точки(промах) на поле боя.
5. Def del_error_f: закрытие решенной ошибки.
6. Def terminate: закрытие программы.
7. Def error_place: функция, которая вызывается при некорректной установке кораблика.
8. Def error_choose: функция, которая вызывается, если игрок не выбрал кораблик для установки.
9. Def error_count_ships: функция, которая вызывается, если игрок пытается поставить определенный кораблик больше раз, чем положено.
10. Def error_wrong_pole: функция, которая вызывается, если игрок пытается поставить кораблик не на свое поле.
11. Def error_before_w: функция, которая вызывается, если игрок пытается закончить расстановку раньше, чем поставит все кораблики.
12. Def hide_pole: функция, которая вызывается, чтобы скрыть поле и видимость кораблей, когда игрок закончил расстановку кораблей.
13. Def error_queue: функция, которая вызывается, если игрок пытается ходить не в свою очередь.
14. Def del_error_game_window: функция для очистки ошибок на поле боя.
15. Def error_same_cell: функция, которая вызывается, если игрок пытается выстрелить туда, куда уже стрелял.
16. Def player: функция для обозначения на поле боя, кто сейчас должен ходить.
17. Def start_screen: создание первого экрана с правилами и циклом для вызова экрана расстановки кораблей.
18. Def boards: функция для создания подписей к полям боя.
19. Def main_screen: создание всех спрайтов для экрана расстановки кораблей, расстановка этих спрайтов, цикл:
    1. Проверка на то, закрыли окно или нет;
    2. Проверка на то, что пользователь нажал на нижние таблицы, то есть на таблицы с экземплярами корабликов, внутри:
        1. Проверка, что нажата левая таблица с горизонтальными кораблями. Если правда, то закрашивается ячейка с данным кораблем, переменная cell приобретает значение кол-ва палуб на нажатом корабле, а также переменная side приобретает значение направления для построения кораблика на поле (в данном случае направо).
        2. Проверка, что нажата правая таблица с вертикальными кораблями. Все аналогично с горизонтальными, только переменная side приобретает другое значение (в данном случае вниз).

3. Проверка, что нажата “W”:
    1. Если W нажата первый раз и глобальные переменные, отвечающие за кол-во расставленных кораблей, равны все 0, то очищаются все ошибки, которые могли быть выведены, глобальные переменные, отвечающие за кол-во оставшихся кораблей для расстановки, приобретают свое изначальное значение, скрывается первое поле, глобальная переменная, отвечающая за кол-во нажатий на W, принимает значение 1, обозначающего, что она нажата один раз, глобальная переменная RED_FLAG_POLE, отвечающая за то, что расстановка на первом поле окончена, принимает значение True.
    2. Если W нажата второй раз и глобальные переменные, отвечающие за кол-во расставленных кораблей, равны все 0, то опять скрываются все ошибки с поля, скрывается расстановка со второго поля, глобальная переменная, отвечающая за кол-во нажатий на W, уже принимает значение 2, а после этого вызывается само поле боя, то есть то место, где будет происходить сама игра.
    3. Если W нажата раньше положенного, то выходит ошибка с предупреждением.
4. Проверка, что нажата ячейка игрового поля:
    1. Если нажато левое поле и RED_FLAG_POLE имеет значение False (то есть расстановка на поле первого игрока не окончена), то вызывается функция get_click для экземпляра класса Board, благодаря которой в последствии прорисовывается кораблик на поле.
    2. Если нажато правое поле и глобальная переменная, отвечающая за кол-во нажатий на W, равна 1 (то есть расстановка на первом поле уже завершена), то все аналогично процессу установки кораблика на поле первого игрока.
    3. Иначе появляется предупреждение о том, что игрок пытается поставить кораблик не на свое поле.

Далее идет отрисовка всех этих полей (2 поля боя и 2 поля с экземплярами корабликов), спрайтов и вызывается функция pygame.display.flip()

20. Def game_window: отрисовка нужного интерфейса для данного экрана и запуск цикла:
1. Проверка на то, закрыли окно или нет.
2. Проверка на то, что было нажатие мышки:
    1. Если нажата ячейка левого поля и игрок выстрелил в порядке очереди, то глобальная переменная HOD увеличивается на единицу, очищается экран от возможных ошибок, на первом поле меняется значение координаты ячейки, в которую был произведен выстрел; если игрок выстрелил не в свою очередь, то появляется предупреждение.
    2. Если нажата ячейка правого поля и игрок выстрелил в порядке очереди, то процесс аналогичный предыдущему, только значение координаты ячейки меняется на другом

поле; если игрок выстрелил не в свою очередь, то появляется предупреждение.

3. Проверка, что кораблей с поля первого игрока не осталось:
    1. Глобальная переменная принимает значение ‘Player 2’, вызывается конечное окно результатов.
4. Проверка, что кораблей с поля второго игрока не осталось:
    1. Глобальная переменная принимает значение ‘Player 1’, вызывается конечное окно результатов.

Далее идет отрисовка полей и точек выстрелов.

21. Def end_window: очистка экрана и вывод результатов: победитель и кол-во ходов каждого игрока.
22. Class Board():
    1. Def __init__: инициализация.
    2. Def set_view: инициализация.
    3. Def render: отрисовка поля.
    4. Def get_cell: поиск номера ячейки по заданным координатам; функция возвращает либо координату ячейки, либо ничего.
    5. Def on_click: здесь происходит подготовка к отрисовке корабля. Проверяется, нажата ли ячейка поля с экземплярами корабликов, далее проверяется, не закончилось ли дозволенное число корабликов той длины, которую выбрал пользователь. Если закончилось, то вызывается предупреждение об ошибке, чтоб пользователь выбрал другой тип корабля, а если все хорошо, то вызывается функция для отрисовки палуб ok_click.
    6. Def ok_click: сначала проверяется, на какое поле нужно ставить кораблик (от этого зависит то, какая матрица будет использоваться. Далее идет проверка, в какую сторону надо строить палубы:
        1. Если направо, то: идет проверка на то, правильно ли игрок хочет установить кораблик. Если положение неверное, то появляется предупреждение об этом. Если с установкой все хорошо, то далее от глобальной переменной, отвечающей за кол-во поставленных корабликов этого типа, отнимается единица, чтоб сократить дозволенное кол-во для расстановки далее. После этого запускается цикл, благодаря которому сначала каждая клетка прокрашивается в нужный цвет, а далее меняется значение в матрице на данном месте и местах вокруг будущего корабля. Благодаря этому можно отследить в проверках, можно ставить в этих местах еще один корабль или нет.
        2. Если вниз, то все абсолютно аналогично, только кораблик строится в другую сторону и меняются другие значения в матрице.
    7. Def get_click: функция, позволяющая понять, куда нажал пользователь: на поле или куда-то вне него. Если на поле, то вызывается функция on_click.
    8. Def fire: проверяется, на какое поле был сделан выстрел. Если пользователь попал в то же место, куда уже стрелял, то вызывается функция error_same_cell(). Иначе если пользователь попал в кораблик, то значение данной ячейки в матрице меняется на четверку и в зависимости от выбранного поля отнимается «одна палуба», также добавляется единица к глобальной переменной HOD, благодаря которой попавший пользователь

сможет сделать еще один ход, а если не попал, то значение ячейки меняется на тройку.

9. Def coor_fire: проверка на то, что пользователь стреляет на поле, а не в пустоту.

23. Class Board_Choose_Ship:

1. Def on_click: сюда попадает экземпляр выбранного кораблика для того, чтобы закрасить данную ячейку, либо же наоборот убрать с нее «нажатие». Убирается нажатие с ячейки корабля, если он уже поставлен, либо же закрашивается выбранное, так же убирается нажатие с прошлой ячейки, если вдруг пользователь не сразу определился, что хочет поставить, из-за чего нажимал на несколько корабликов.