ВВЕДЕНИЕ

Изначально в плане было 6 пунктов:

1. Графический интерфейс

* Заставка
* Сами карты
* Фон
* Окончания игры на случаи выигрыша и проигрыша
* Правила игры

1. Музыкальное оформление

* Фоновая музыка
* Звуки на передвижение карт

1. Логика игры

* Движение карт на экране
* Возможность разложить пасьянс
* Раздача карт по одной и по три
* Стиль раздачи – стандартный и Лас-Вегас. Их отличие заключается в том, что при стандартном стиле колоду можно просматривать сколько угодно раз, а при стиле Лас-Вегас – только трижды

Следовательно, предполагалось, что будет 4 уровня сложности: стандарт с раздачей по одной карте – самый лёгкий, Лас-Вегас с одной картой – посложнее, стандарт с тремя картами – третий по сложности, и Лас-Вегас с тремя картами – самый сложный.

* Счёт: за отправку карты из колоды в ячейку добавляется 5 очков, из ячейки в дом – 10, из колоды в дом – 20
* Время
* Бонусы к счёту за время: чем быстрее разложен пасьянс и чем выше его уровень сложности, тем больше бонус, и наоборот.

1. Статистика и работа с пользователем

* После завершения игры на экране появляется таблица со статистикой, по типу:

|  |  |
| --- | --- |
| Ваш счёт | Счёт  (он считается во время игры) |
| Время | Время  (отсчитывается с момента перехода с начального экрана на основное игровое поле) |
| Бонус за время | Бонус  (прибавляется к счёту после окончания игры) |
| Общий счёт | Счёт + бонус |
| Лучший счёт | Лучший счёт  (он хранится в текстовом документе; если общий счёт за текущую игру больше – то он обновляется) |
| Лучшее время | Лучшее время  (оно также хранится в текстовом документе; если время за текущую игру меньше, то оно обновляется) |
| Игр сыграно | Кол-во сыгранных игр  (тоже хранится в текстовом документе) |
| Игр выиграно | Кол-во выигранных игр  (хранится в том же текстовом документе) |
| Процент выигрыша | Процент выигрыша, равный отношению всех сыгранных игр к выигранным играм, и округлённый, до целого |

Для каждого из четырёх уровней сложности ведётся своя статистика, которая обновляется после каждой сыгранной партии.

1. PyQt5

* Кнопка, отвечающая за количество карт в раздаче
* Кнопка, отвечающая за стиль раздачи
* Кнопка начала новой игры
* Кнопка правил игры
* Кнопка отключения звука

PyQt5 планировался для создания кнопок, но внезапно возникли сложности с его интегрированием в программу pygame, поэтому я решила сделать кнопки, также используя pygame. Это получилось быстрее, удобнее, и практичнее.

1. Дополнительные возможности

* Разные темы карт. Они либо заложены в одном из пунктов меню, либо добавляются пользователем через командную строку
* Различные соревнования и челленджи, по типу присутствующих в игре Microsoft Solitaire Collection, например, разложить пасьянс меньше чем за определённое число ходов, отправить в дом определённую карту, набрать определённый счёт и так далее. Различные вариации челленджей записаны в список и выбираются из него рандомным образом
* Возможность отправить все доступные карты в дом нажатием правой кнопки мыши
* Кнопка «собрать», появляющаяся только в том случае, когда вся колода разобрана и ни в одной ячейке нет закрытых карт

ОПИСАНИЕ РЕАЛИЗАЦИИ

Сначала идёт импорт библиотек, опередиление констант и запуск фоновой музыки.

Затем функции: *terminate(), load\_background(name, colorkey=None), load\_image(name, colorkey=None), start\_screen(), finish\_screen()*, *card\_choice()*. Рассмотрим работу каждой из них.

Функция *terminate()* отвечает за выход из программы при нажатии красного крестика.

Функции *load\_background(name, colorkey=None)* и *load\_image(name, colorkey=None)* имеют одинаковый принцип работы и различаются только тем, что первая загружает стиль карт из папки data, а вторая – фоны из папки pics. Это было сделано для моего удобства – меня путало то, что карты и фоны лежат в одной папке.

После идёт функция *start\_screen(),* которая вызывается из главной функции *main()* и отвечает за красивую картинку на начальном экране.

Затем – функция *finish\_screen()*, которая также вызывается из главной функции *main(),* когда пасьянс разложен. Отвечает за надпись *«Congratulations, You Won!»*, время, за которое был разложен пасьянс и возможность начать новую игру, не закрывая окна.

Следующая – функция *card\_choice()*, которая должна позволять пользователю самому выбрать, сколько карт будет открываться за раздачу. Но что-то пошло не так, а время уже поджимало, поэтому выбор количества карт происходит рандомным образом, в функции *main()*.

Далее идут классы.

Класс *MovedCard(object)* отображает передвижение карт на экране. В нём инициализируются флаг *self.moved,* равный *True*, когда, хоть какая-то карта на экране, движется. Его функция *click\_up(self, deck\_list)* запускается после того, как пользователь отпускает левую кнопку мыши и проверяет, возможно ли вообще положить карту на другую. Функция *draw(self, screen, card\_dict)* отображает перемещение карты на экране.

Класс *Deck* – это родительский класс, проверяющий, попал ли игрок по карте или нет.

Класс *Deck1* – сыновний от класса *Deck.* Он позволяет класть карты в ячейках друг на друга только в правильной последовательности, чередуя цвета, а именно: король, дама, валет, десять, девять, восемь, семь, шесть, пять, четыре, три, два, туз.

Класс *Deck2* – это рисовка колоды. Переменная *kol*, которую мы опредилили ранее, применяется именно здесь – при нажатии на колоду, если она равна одна, открывается одна карта, а если трём – то три.

Класс *Deck3* – это «дома». Он отвечает за возможность положить друг на друга карты только одной масти в порядке: туз, два, три, четыре, пять, шесть, семь, восемь, девять, десять, валет, дама, король.

После классов снова идут функции.

*Shuffle\_cards()* отвечает за перемешивание карт и их распредиление на экране рандомным образом.

И, наконец, *main()* отвечает за основной цикл и нормальную работу всей программы. В ней:

* Опредиляются и рассчитываются переменные *vremya* и *kol*, отвечающие, соответственно, за время, за которое пользователь раскладывает пасьянс, и количество карт в раздаче;
* Загружаются карты;
* Рисуются и реализуются кнопки, отвечающие за быстрый рестарт игры и отключение звука фоновой музыки;
* Опредиляются 12 рабочих зон – колода, 7 обычных ячеек и 4 «дома», в которых происходит работа с картами

*DECK3*

*DECK2*

*DECK1*

*DECK*

*MOVEDCARD*

ЗАКЛЮЧЕНИЕ

Безусловно, на данный момент реализовано далеко не всё из того, что было запланировано в начале работы. Если считать по подпунктам, приведённым во введении, то можно с уверенностью сказать, что сделано не всё, но – больше половины.

Это связано с ограниченностью и нехваткой времени, и – некоторым недостатком знаний и умений. Функция *card\_choice()*, скорее всего, могла бы работать, начни я делать её немного раньше и будь моё эмоциональное состояние на тот момент стабильнее. Так она, конечно, работала, но ровно до тех пор, пока не была добавлена обработка событий нажатий на кнопку. Поэтому пришлось функцию закомментировать и реализовать её в трёх строчках в функции *main()*. Конечно, обидно, зато программа работает, а разобраться, в чём была проблема с *card\_choice()* стоит на первом месте в моих дальнейших планах на игру.

Так же, я планирую доработать статистику и раздачу карт в стиле Лас-Вегас. На мой взгляд, после этого игра заиграет новыми красками и станет гораздо интереснее.

Пункт VI введения планирую реализовать полностью, но чтобы это сделать, нужно для начала поднабраться знаний и умений.

Если по ходу процесса доработки мне в голову придут ещё какие-нибудь мысли, то и они обязательно будут воплощены в жизнь. Так, например, во время написания этого файла, я вспомнила, что разворот окна *pygame* на весь экран работает не совсем корректно – это тоже будет включено в планы по доработке.

ОТДЕЛЬНАЯ БЛАГОДАРНОСТЬ НИКИТЕ КУВШИННИКОВУ, КОТОРЫЙ СДЕЛАЛ ОКОЛО 10% ОБЪЁМА ВСЕЙ РАБОТЫ И ОКАЗЫВАЛ МНЕ МОРАЛЬНУЮ ПОДДЕРЖКУ.