Название проекта: **«Wandering Wind»**

Автор проекта: **Шакирова Камиля**

Идея проекта: Персонажа зовут *Ветер* (англ. *Wind*), он бродит по лабиринту, отсюда появилось название игры *Wandering Wind* (русс. *Блуждающий Ветер*). В начале игры у него есть 3 жизни на прохождение всех уровней. Персонаж при взаимодействии с огнем теряет 1 жизнь. Если его поймает стражник, игра сразу заканчивается.

В лабиринте спрятаны монетки, дополнительные жизни и сундуки с ключами, открывающими дверь на новый уровень.

Персонаж управляется клавишами ↑, ↓, ← и →.

Вам нужно выбраться и набрать как можно больше очков.

Монетки – 55 очков

Дополнительные жизни – 133 очка

Сундук – 200 очков

Прохождение каждого уровня – 1000 очков

Всего в игре 3 уровня, с каждым уровнем сложность растёт.

Задачи:

* Придумать сюжет
* Найти необходимые картинки в интернете
* Разобраться с анимацией спрайтов
* Изучить новые модули из pygame
* Продумать и написать карты для уровней разной сложности
* Создать необходимые классы и функции
* Добавить звуковые эффекты
* Подобрать красивые заставки

Используемые модули:

* pygame и его отдельные классы
* os – для загрузки из файлов

Используемый инструментарий pygame:

* Спрайты и их взаимодействие
* Звук
* Фоновые изображения

Созданные спрайты:

* Персонаж с анимацией
* Тайлы уровня
* Стражник с анимацией
* Огонь с анимацией
* Ключ
* Монетки с анимацией
* Сердца (доп. жизни) с анимацией
* Сундуки с анимацией (разные на каждом уровне)

Описание реализации: **Функции**: сохранение и выход из игры, смена дизайна курсора, загрузка спрайтов, загрузка нескольких изображений, загрузка уровня, создание карты для уровня, вывод текста, движения персонажа, меню, правила, запуск уровней, 2 концовки. **Классы**: тайлы уровня, персонаж, стражник, монетки, сердце, сундук, огонь, камера. Классы

Класс **Camera**, методы

* Конструктор \_\_init\_\_()
* Метод apply – фиксация на объекте
* Метод update – перемещение всего окружения относительно камеры

Класс **Sprite**, методы

* Конструктор \_\_init\_\_()
* Пустой метод update
* Пустой метод get\_event

Класс **Tile**, методы

* Конструктор \_\_init\_\_()

Класс **Heart**, методы

* Конструктор \_\_init\_\_()
* Метод update – для замены на следующий кадр анимации, проверка на столкновение с персонажем

Класс **Coin**, методы

* Конструктор \_\_init\_\_()
* Метод cut\_sheet – нарезка одного изображения на равные части
* Метод update – для замены на следующий кадр анимации, проверка на столкновение с персонажем

Класс **Fire**, методы

* Конструктор \_\_init\_\_()
* Метод update – для замены на следующий кадр анимации, проверка на столкновение с персонажем

Класс **Chest**, методы

* Конструктор \_\_init\_\_()
* Метод cut\_sheet – нарезка одного изображения на равные части
* Метод update – для замены на следующий кадр анимации, проверка на столкновение с персонажем

Класс **Player**, методы

* Конструктор \_\_init\_\_()
* Метод cut\_sheet – нарезка одного изображения на равные части
* Метод update – для замены на следующий кадр анимации или смены вида
* Метод move – перемещения героя

Класс **Guard**, методы

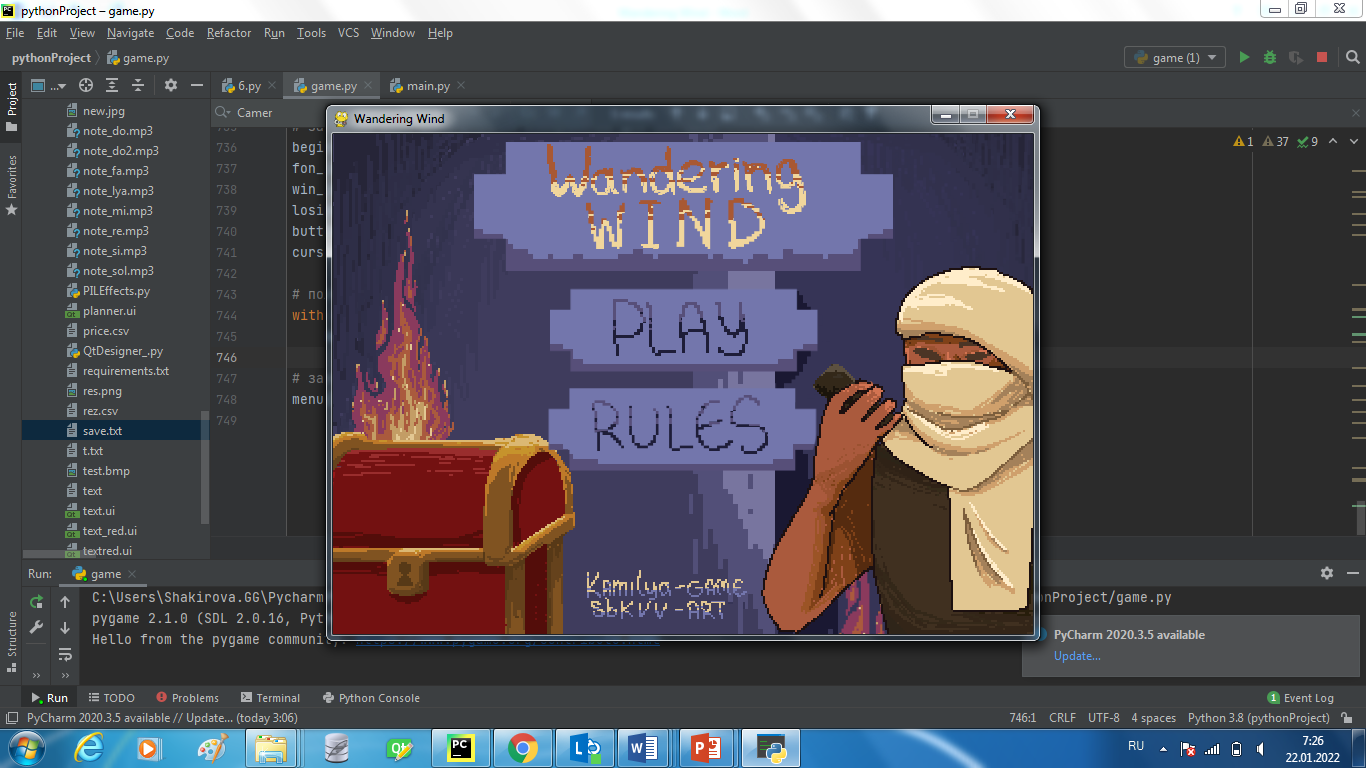
* Конструктор \_\_init\_\_()
* Метод cut\_sheet – нарезка одного изображения на равные части
* Метод update – для замены на следующий кадр анимации или смены вида, проверка на столкновение с персонажем
* Метод find\_path\_step – поиск кратчайшего пути к персонажу
* Метод move – перемещения стражника

Заключение:

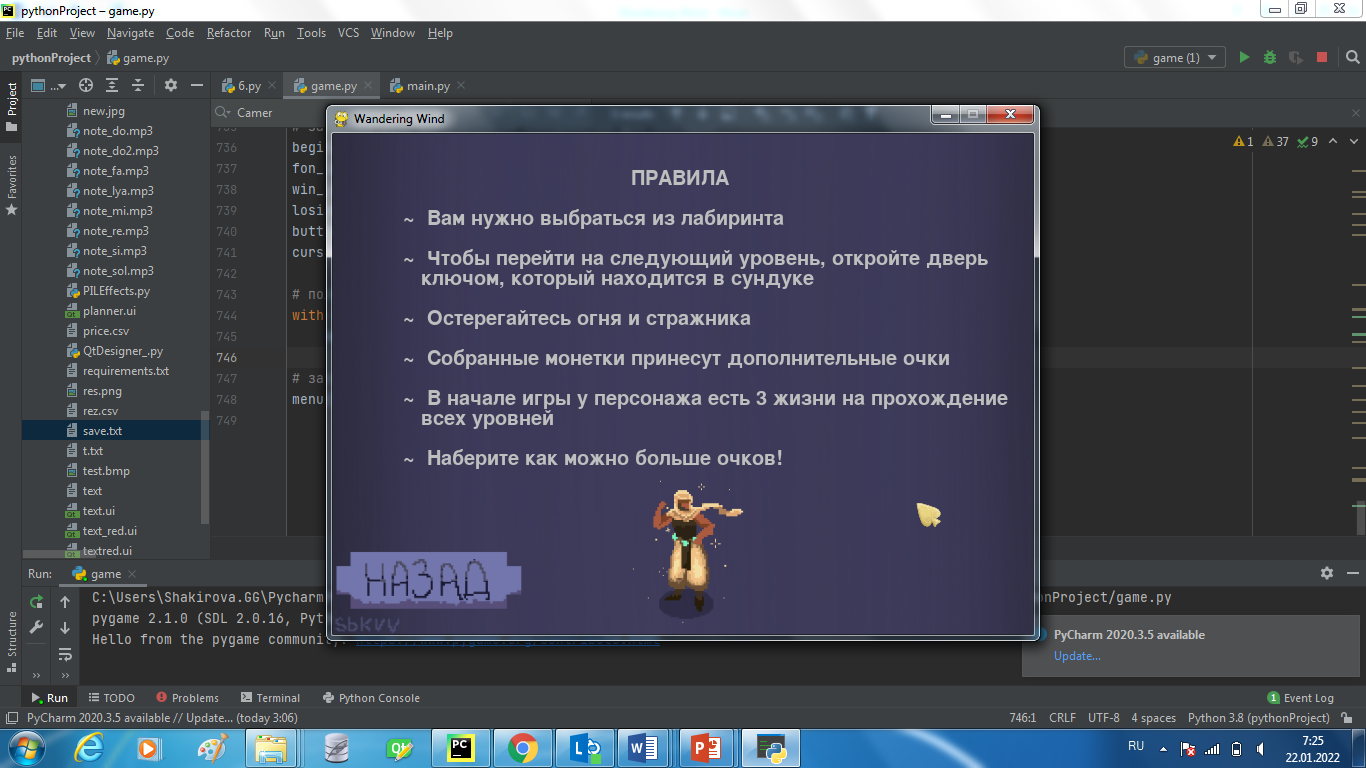
Я сделала огромную работу и считаю, что хорошо справилась с ней. Все поставленные задачи выполнены. Результат превзошёл мои ожидания, чему я очень рада.

Скриншоты:

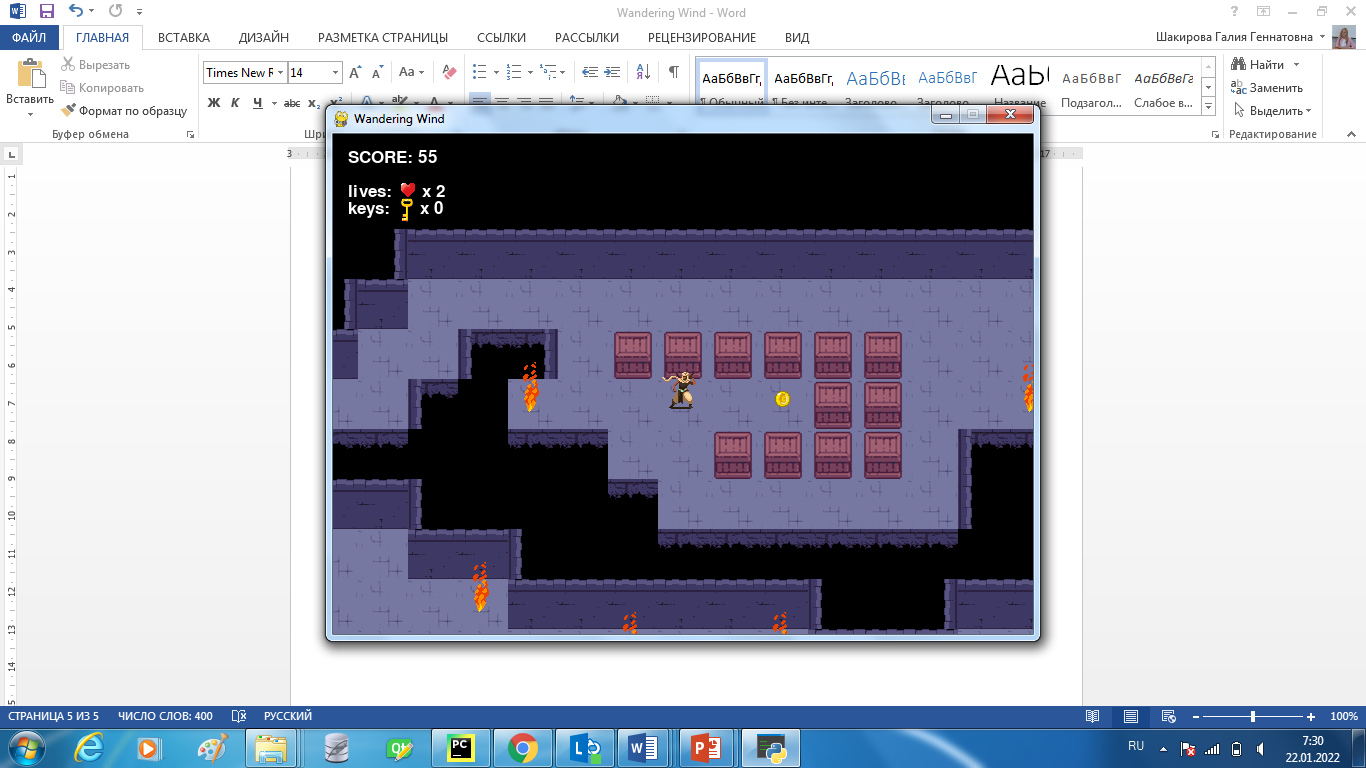
Меню



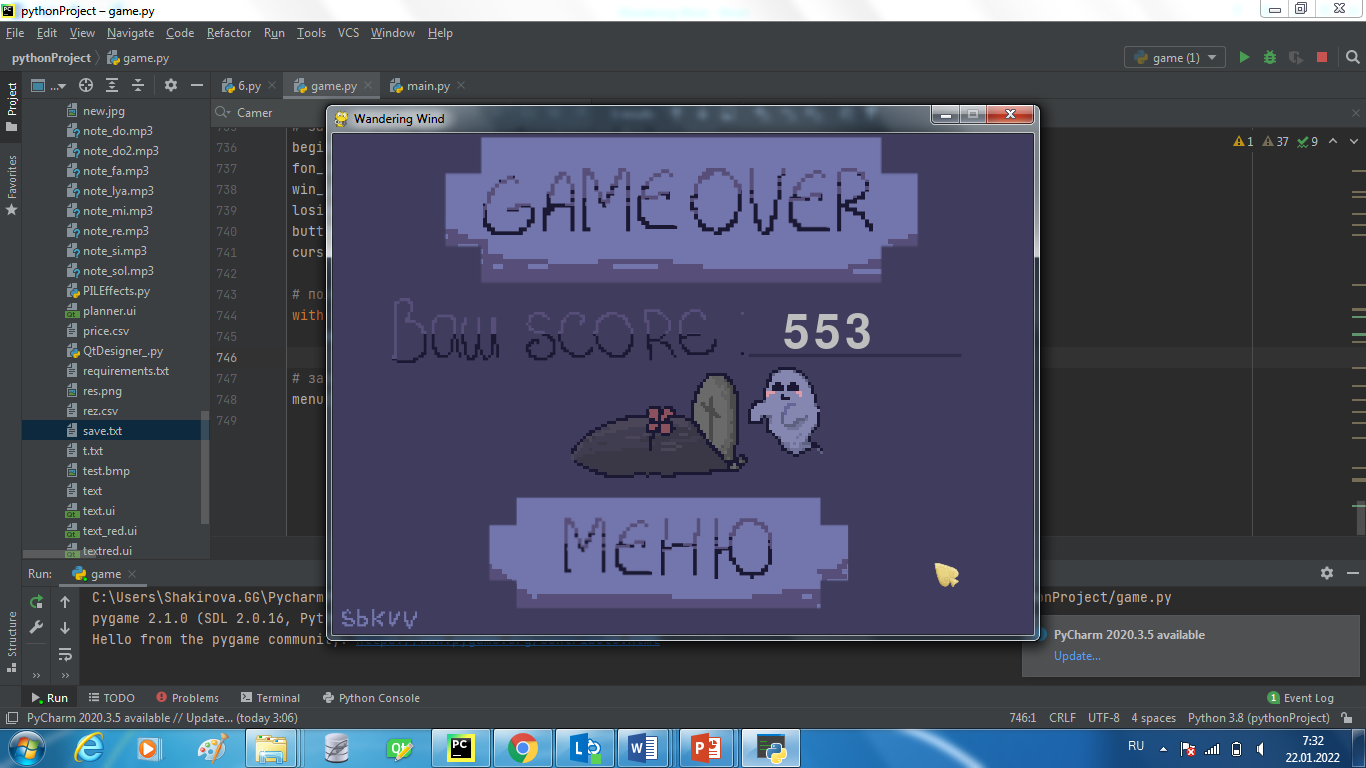
Правила



Игровой процесс



Проигрыш



Выигрыш

