Пояснительная записка к проекту

«Игра Transformers»

Выполнила: Бегеева Абидат

Проверил: Мурадов Марат

**Техническое задание.**

**Цель:** создать игру на языке программирования python по трансформерам.

**Требования к игре:** игра должна быть написана с помощью библиотеки «pygame» и представлять собой простой пиксельный 2D раннер, в котором игроку, управляющему автоботом-Клиффджампером, предстоит бегать в бесконечном уровне (на Кибертроне) до тех пор, пока не произойдёт столкновение с потолком тоннеля или же с врагами (коими являются десептиконы-вехиконы). Чтобы избежать столкновения игрок должен уметь прыгать (пробел/левая кнопка мыши) и трансформироваться (LShift/правая кнопка мыши). При превращении в автомобиль движение всех объектов вокруг героя должно ускоряться и наоборот: при превращении в робота – замедляться.

В игре должно быть всего три экрана: стартовое меню, уровень и экран проигрыша. Окно игры – 768 на 384 пикселей. Иконка – значок автоботов. Название окна – «Transformers».

В меню должен быть изображён космос («space.png»), Кибертрон («Cybertron.png») в правом нижнем углу, звёзды, летящие слева направо, надпись «Transformers» («title.png») слева от планеты, а под надписью кнопка «START». После нажатия на кнопку она должна сменить свой цвет на тёмно-синий, и через несколько секунд должен начаться уровень.

Он должен выглядеть так: внизу платформа («ground.png»), шириной в экран, а на ней, слева, Клиффджампер; на заднем плане тёмно-синий силуэт города, ночное небо (space.png). Препятствия должны появляться каждые 4 секунды справа и, соответственно, двигаться влево. Шанс появления тоннеля 25%, десептикона – 75%.

На экране проигрыша должно быть отображено количество очков за данную игру и лучший результат за все игры. Цвет фона – (9, 11. 33), цвет надписей – белый. Шрифт – «FFFFORWA.TTF».

В меню должна играть (целиком) песня «The transformers (theme)», взятая из мультфильма 80-х, в уровне – отрывок от продлённого саундтрека из заставки мультсериала «Transformers: Cybertron» (громкость – 1), а при проигрыше должен продолжать играть тот же трек, что и в уровне, но потише (громкость – 0.3).

**Проектирование.**

В программе должно быть:

1. четыре функции:
   1. **main\_menu()**;

Это функция, показывающая и отвечающая за работу главного меню.

* 1. **main\_game()**;

Это функция, показывающая и отвечающая за работу уровня и экрана проигрыша.

* 1. **collision(sprite, group)**;

Эта функция отвечает за спроверку на столкновение спрайта главного героя с группой спрайтов вехиконов или тоннелей (pygame.sprite.spritecollide()).

* 1. **save\_scores(score)**.

Эта функция заносит результаты в текстовый файл «scores.txt» при столкновении с препятствием.

1. семь классов (каждый из которых унаследован от pygame.sprite.Sprite, т.е всё это – спрайты):
   1. **Star**;

Это класс звёздочек из главного меню.

Изображение этого спрайта – картинка «a star.png». Звёздочке задаётся случайная скорость (число от 1 до 4) и случайные координаты x (от 1 до 768) и y (от 1 до 384). Есть метод update(self), в котором звезда смещается вправо и в котором запускается метод destroy(self). Тот проверяет вышел ли спрайт за границу экрана и, если да, уничтожает его.

* 1. **StartBtn**;

Класс кнопки «START» из главного меню. Изображением этого спрайта по умолчанию является картинка «start btn.png». Координаты кнопки: x=140, y=250. В методе update(self, \*args) происходит проверка на то, была ли кнопка нажата или нет. Если была, изображение спрайта меняется на «start btn pressed.png»

* 1. **City**;

Класс силуэта города на заднем плане уровня. Изображение – «bg city.png». У этого класса должен быть атрибут «speed», отвечающий за скорость передвижения города по экрану. В методе update(self) происходит сдвижение города влево и вызывается метод destroy(self), который, если город вышел за пределы экрана, уничтожает спрайт.

* 1. **Player**;

Анимированный класс игрока. В методе update(self) вызываются методы player(), проверяющий надо ли персонажу прыгнуть или трансформироваться в зависимости от того, какая кнопка была нажата игроком, animation(), отвечающий за анимацию, и apply\_gravity(), отвечающий за гравитацию персонажа. Изображения для анимации лежат в папке «cliff sprites», при превращении в автомобиль изображение должно меняться на «cliff alt mode.png», а при прыжке – на «cliff jump.png».

* 1. **TunnelCeiling**;

Класс потолка (с которым может произойти столкновение) тоннеля. Картинка– «tunnel ceiling.png». В методе update(self) происходит сдвижение спрайта влево и вызывается метод destroy(self), который уничтожает спрайт, если он вышел за пределы экрана.

* 1. **TunnelWall**;

Класс стены (с которой столкновение не происходит) тоннеля. Картинка– «tunnel wall.png». В методе update(self) происходит сдвижение спрайта влево и вызывается метод destroy(self), который уничтожает спрайт, если он вышел за пределы экрана.

* 1. **Enemy**

Класс врага (десептикона-вехикона, с которым может произойти столкновение), Изображение – «vehicon steve.png». В методе update(self) происходит сдвижение спрайта влево и вызывается метод destroy(self), который уничтожает спрайт, если он вышел за пределы экрана.

Каждая секунда, проведённая в игре, прибавляет по одному очку.

**Инструкция по игре.**

1. Чтобы начать игру нажмите на кнопу «START».
2. Для совершения прыжка нажмите на пробел или левую кнопку мыши, для трансформации в автомобиль и обратно – LShift или правую кнопку мыши.
3. При приближении десептикона надо прыгать, при приближении тоннелей – превращаться в автомобиль.
4. Чтобы возобновить уровень после поражения надо нажать на левую кнопку мыши.