Kamil Maksymowicz 130542

Projekt – Gra Arcanoid

Link do projektu: https://github.com/mvkso/ARCANOID-PROJECT

Ogólny opis kodu:

Do stworzenia interfejsów graficznych, komunikatów, menu głównego, dźwięków, zegara, obiektów i ich ruchów, oraz kolizji użyto bilbioteki PyGame.

Ustawiono płynność rozgrywki na 15 klatek na sekundę. Funkcja **tick()** odpowiada za ruch obiektów co daną klatkę.

Metody **youlost()** i **nextlevelbutton()** tworzą na ekranie w odpowiednich momentach prostokąt z napisem w formie komunikatu, gdy gracz przegrał, bądź może przenieść się do kolejnego etapu. Dodatkowo funkcją **isOver()** sprawdza się, czy kursor znajduje się nad powyższym prostokątem. Jeśli tak, jego kolor zmienia się z czerwonego na zielony, co pozwala na większą przejrzystość.

Funkcja **mainmenu()** tworzy początkowe menu główne, którym można przejść do pierwszego poziomu, bądź wyjść z gry. **Lvl_creator()** generuje poziomy dzięki podanym argumentom(który poziom, ilość bloków do zbicia, prędkość poruszania się paletki gracza, prędkość odbijanej „piłki” w wymiarze x, prędkość odbijanej „piłki” w wymiarze y). Wewnątrz generatora poziomów używana jest metoda **wait()**, dzięki której piłka odczekuje chwilę zanim ruszy się. Do wykrycia kolizji między piłką, a cegłą używa się **collision()**.

Do użytku stworzono również kilka klas:

- Button – klasa tworząca przycisk(prostokąt, który klika się dzięki eventom)
- Paddle – paletka sterowana przez gracza w lewo i prawo, która odbija piłkę
- Brick – cegły, które są niszczone przez piłkę
- Ball – odbijany obiekt

Co udało się, a z czym były problemy:

- Generowanie losowo poziomów powoduje, że cegły mogą na siebie nachodzić przy większych ilościach. Z drugiej strony tworzenie ręcznie poziomów byłoby zbyt czasochłonne.

- Nie napotkano większych problemów przy tworzeniu i sprawdzeniu testów.

- Tworzenie kilkunastu, a nawet kilkudziesięciu osobnych obiektów Brick było zbyt czasochłonne i kosztowne. Zamiast tego, dzięki generowaniu poziomów użyto listy przechowującej obiekty, którym przypisuje się losowe położenie.

- Obiekt Ball, może się wydawać, że czasami ma problemy z kolizją, lecz taka sytuacja zdarza się relatywnie rzadko.

- Przy tworzeniu nowych poziomów trzeba było użyć funkcji, dzięki której obiekt **Ball** zaczeka chwilę zanim ruszy i da szansę graczowi na ustawienie paletki, gdyż przy szybszym poruszaniu się byłoby to iryrujące i mało praktyczne.

Linki do przydatnych elementów kodu:

1. Klasy:
   - https://github.com/mvkso/ARCANOID-PROJECT/blob/master/ball.py
   - https://github.com/mvkso/ARCANOID-PROJECT/blob/master/brick.py
   - https://github.com/mvkso/ARCANOID-PROJECT/blob/master/button.py
   - https://github.com/mvkso/ARCANOID-PROJECT/blob/master/paddle.py

2. Metody w głównej klasie:
   - https://github.com/mvkso/ARCANOID-PROJECT/blob/master/main.py#L132
   - https://github.com/mvkso/ARCANOID-PROJECT/blob/master/main.py#L132
   - https://github.com/mvkso/ARCANOID-PROJECT/blob/master/main.py#L151
   - https://github.com/mvkso/ARCANOID-PROJECT/blob/master/main.py#L166
   - https://github.com/mvkso/ARCANOID-PROJECT/blob/master/main.py#L177
   - https://github.com/mvkso/ARCANOID-PROJECT/blob/master/main.py#L193

- https://github.com/mvkso/ARCANOID-PROJECT/blob/master/main.py#L203
- https://github.com/mvkso/ARCANOID-PROJECT/blob/master/main.py#L229
- https://github.com/mvkso/ARCANOID-PROJECT/blob/master/main.py#L277

3. Konstruktor klasy Game:

- https://github.com/mvkso/ARCANOID-PROJECT/blob/master/main.py#L25

4. Testy:

- https://github.com/mvkso/ARCANOID-PROJECT/blob/master/test.py

5. Podział na moduły:

- https://github.com/mvkso/ARCANOID-PROJECT