Projekt: gra pędzące żółwie

Autorki: Róża Wójcicka, Emilia Wiśniewska

Projekt jest dostępny na githubie:
https://github.com/emilia276223/Turtles_game.git

Klasy (spis):

1. Card
2. Deck
3. Player
4. Field, StartField
5. Board
6. Game
7. GameMock
8. GUI
9. DrawBoard
10. DrawTurtle
11. DrawCard
12. Connection
13. ConnectionMock
14. Client
15. TextUI
16. Server
17. ServerMock
18. UserInfo

Opis klas:

Klasa Card:

Klasa karta ma kolor i wartość, która jest informacją, jak zmienia położenie żółwia, na którego działa. Metody klasy:

1. get_state() - wypisywanie karty jako słownika
2. __str__() - wypisywanie karty

Klasa Deck:

Klasa talia kart przechowuje stos kart do dobierania i stos kart odrzuconych. Na początku gry dodaje do talii kart odpowiednia ilość różnych rodzajów kart i tasuje karty. Metody klasy:

1. take_card() - usuwa pierwszą karte ze stosu kart do dobierania (gracz dobiera kartę), w razie gdy skończą się karty do dobierania (zostanie ostatnia) tasuje karty

odrzucone i dodaje je do stosu do dobierania więc stos kart odrzuconych zostaje pusty
2. throw_card() - dodaje kartę do stosu kart odrzuconych (gracz zagrał kartę)

Klasa Player:

Klasa gracz przechowuje ip gracza i jego aktualne karty. Metody klasy:

1. add_card() - dodanie dobranej ze stosu karty do kart na ręce
2. remove_card() - usunięcie karty z ręki gracza (gracz zagrywa kartę)
3. get_state() - wypisanie aktualnych kart na ręce gracza

Klasa Field:

Klasa pole przechowuje listę żółwi, które na nim stoją (w kolejności w jakiej były stawiane na to pole). Metody klasy:

1. add_turtle() - dodanie żółwia do listy (żółw został postawiony na to pole)
2. take_turtle() - usunięcie z listy żółwia wraz ze wszystkimi żółwiami stojącymi na nim (żółw został zabrany z tego pola)
3. get_state() - wypisuje aktualny stan pola (listę stojących na nim żółwi)

Klasa StartField:

Podklasa klasy pole - pole startowe. Na początku gry stoją na nim wszystkie żółwie. Ma nadpisaną metodę take_turtle(), która usuwa z listy tylko tego jednego (podanego) żółwia (żółw został zabrany z tego pola).

Klasa Board:

Klasa plansza przechowuje informacje o stanie planszy, ma pola (klasy Field), między którymi przechodzą żółwie. Metody klasy:

1. finish() - gdy gra się kończy tworzy ranking
2. get_ranking() - tworzenie rankingu (w jakiej kolejności żółwie były od mety na końcu gry)
3. get_state() - zwraca aktualny stan planszy (słownik kolejnych pól z ich stanem - na którym polu które żółwie stoją i w jakiej kolejności)
4. accept_card() - przesuwa żółwia na planszy zgodnie z informacją na zagranej karcie

Klasa Game:

Klasa gra koordynuje przebieg gry: kolejność graczy, ich tury, zachowanie się planszy. Dostaje listę adresów ip graczy i liczbę pól, które są na planszy (w sumie ze startem i metą). Tworzy stos kart do dobierania (klasy Deck), planszę (klasy Board), graczy (klasy Player) i daje każdemu z nich po 5 kart. Metody klasy:

1. get_ip_of_next() - zwraca ip gracza który następny będzie wykonywał ruch
2. turn() - przebieg tury gracza (karta rusza żółwiem, zabieramy ją z ręki gracza i wrzucamy na stos kart odrzuconych, gracz dobiera nową kartę)
3. card_on_desk() - zagranie karty przez gracza (jeśli właściwy gracz zagrał to przeprowadzamy ruch i zwracamy stan gry, gdy zagrał gracz który nie ma teraz ruchu nic się nie dzieje)
4. get_state() - zwraca aktualny stan planszy i lista stanów kart graczy, a gdy koniec gry zwraca ranking

Klasa GameMock
Klasa imitująca klasę Game, do testów.

Klasa GUI:
Klasa GUI zajmuje się obsługą interfejsu graficznego. Metody klasy:

1. start() - uruchamia okno i dodaje informację o żółwiu gracza
2. end() - zamyka okno i kończy grę
3. ask_if_needed() - dopytuje gracza o to, jakim żółwiem chce poruszyć; jest wykorzystywana gdy gracz wybierze kartę tęczową (na przykład tęczowe ++)
4. show() - uaktualnia wyświetlane graczowi położenie żółwi
5. go() - uaktualnia wyświetlane położenie żółwi oraz wyświetla karty gracza, a następnie gracz wybiera, którą z nich chce położyć; jeśli gra została już zakończona to wyświetla informację o wygranej / przegranej

Klasa DrawBoard
Służy do wyświetlania planszy i żółwi się na niej znajdujących. Ma metody:

1. start() - ustawienie żółwia danego gracza, który będzie później wyświetlany
2. draw() - wyświetlanie tła i żółwi w odpowiednich miejscach

Klasa DrawTurtle
Służy do wyświetlenia odpowiedniego żółwia w odpowiednim miejscu. Ma metodę draw(), która właśnie to robi (w tym dobiera odpowiednią z grafik).

Klasa DrawCard
Służy do wyświetlenia karty. Ma metody:

1. draw() - wyświetla odpowiednie grafiki tak, by w całości powstała odpowiednia karta
2. chosen_card() - sprawdza, która karta została wybrana przez gracza na podstawie miejsca kliknięcia

Klasa Connection
Klasa ta odpowiada za komunikację z serwerem. Ma metody:

1. card_on_table() - podaje serwerowi informację, jaka karta została wybrana przez gracza
2. get_state() - pobiera z serwera informację o aktualnym stanie gry

Klasa ConnectionMock
Klasa potrzebna była do testów, imitowała klasę Connection

Klasa Client
Klasa, której obiekt jest danym graczem - obsługuje ona pośrednio dołączenie do serwera, przebieg gry i zakończenie. Ma metodę play() - odświeżenie statusu gry (pobranie nowego przez własny obiekt klasy Connecction) i wykonanie ruchu, jeśli to kolej danego gracza.

Klasa TextUI

Klasa, która obsługuje grę w czasie dołączania graczy do serwera. Ma metody:

1. start() - pyta gracza o serwer, do którego chce dołączyć oraz o nick
2. go_on() - sprawdza czy gra się rozpoczęła i przekazuje odpowiednią informację graczowi

Klasa Server

Klasa, której obiekt jest serwerem, na którym odbywa się gra i do którego podłączają się gracze. Ma metody:

1. card_table() - przekazuje informację grze, jaki gracz położył jaką kartę
2. user_init() - obsługuje dołączenie się nowego gracza do serwera
3. get_nick_list() - zwraca listę nicków graczy
4. get_users_info() - zwraca informacje o wszystkich graczach (ich nicki i przydzielone im żółwie)
5. get_state() - przekazuje stan gry

Klasa ServerMock

Klasa imitująca w pewnym stopniu klasę Serwer, służyła do testów.

Klasa UserInfo

Przetrzymuje informacje o danym graczu (jego nick i żółwia mu przydzielonego). Ma metodę __repr__, która zwraca informacje o tym graczu w postaci napisu.