Jan Szwagierczak 331444

Tysiąc

Cel projektu: implementacja gry karcianej Tysiąc w wariancie dla 2 graczy (użytkownik i gracz komputerowy), ograniczoną do pojedynczej rozgrywki, bez zliczania sumarycznej wartości punktowej graczy w kolejnych rozgrywkach.

Opis projektu: Gra karciana Tysiąc zaimplementowana w języku Python w interfejsie terminalowym.

Gra rozpoczyna się od rozdania kart i rozpoczęcia licytacji przez gracza. Gracz może spasować lub podać wartość w zakresie określonym w zależności od posiadanych kart. Następnie komputer decyduje się czy przebić gracza. Licytacja kończy się wraz ze spasowaniem jednej ze stron, gracza lub komputera. Po licytacji gracz który ją wygrał rozpoczyna grę poprzez wybranie musiku i wzięciu z niego kart. Następnie odrzuca wybrane 2 karty ze swojej ręki. Na tym etapie obaj gracze mają po 10 kart i rozpoczyna się rozgrywka. Trwa ona 10 rund. Karty zbijają się zgodnie z zasadami tysiąca, obowiązują meldunki. Po zakończeniu się kart(10 rund minęło) program przedstawia podsumowanie. Ilość zdobytych punktów przez gracza i komputer, zadeklarowaną w licytacji, oraz ostateczny wynik zaokrąglony do pełnych dziesiątek (wynik jest ujemny gdy gracz otrzymał mniej punktów niż deklarował lub równy deklaracji jeżeli takowa była i została przebita lub wyrównana). Następnie użytkownik ma możliwość rozpoczęcia rozgrywki po raz kolejny lub wyjścia z programu.

Klasy:

- **Card:** klasa reprezentująca kartę do gry w tysiąca, posiadająca takie atrybuty jak nazwa, kolor i punkty.
- **Deck:** klasa reprezentująca talię do gry w tysiąca, posiada metody umożliwiające wygenerowanie talii oraz jej posortowanie.
- **Player:** klasa reprezentująca gracza. Posiada wszelkie metody pozwalające na obsługę gracza w grze, odnośnie punktów, meldunków, kart w ręce czy licytacji.
- **Musik:** klasa reprezentująca musik – kupkę kart zawierającą dwie karty które są zbierane przez wygranego licytacji. Klasa dziedziczy po klasie Deck.
- **Computer:** klasa reprezentująca przeciwnika, dziedziczy po klasie Player. Posiada metody umożliwiające komputerowi dokonywania trafnych licytacji oraz graniu odpowiednich kart w rundach, odpowiednie zgłaszanie meldunków.
- **Game:** klasa reprezentująca samą grę, zawiera metody odpowiedzialne za obsługę rund, walki pomiędzy kartami, kontrolę nad aktualnym meldunkiem, sprawdzanie poprawności zagranych kart.

Instrukcja użytkowania:

Po uruchomieniu programu postępować zgodnie z wypisywanymi komunikatami, w każdym momencie można wyjść z gry poprzez wpisanie „exit”, kiedy istnieje taka możliwość. Karty reprezentowane są poprzez nazwę, emoji oraz kolor tak aby można ją było z łatwością zinterpretować. Komunikacja z programem odbywa się głównie za pomocą wpisywania liczb oraz słów. Po zakończonej rozgrywce można wpisać „restart” aby zagrać ponownie. Dokładne zasady gry w tysiąca dla dwóch graczy znajdują się na stronie podanej w opisie do projektu: https://pl.wikipedia.org/wiki/Tysiąc_(gra).

Część refleksyjna:

Udało się zrealizować logikę gry, cały jej przebieg oraz czytelny interfejs za który odpowiedzialna jest biblioteka rich. Największą przeszkodą było zaimplementowanie dobrego systemu, którym posługiwał by się komputer w celu wykonywania licytacji czy ruchów kartami. Jeśli chodzi o licytacje było to prostsze gdyż istnieją opisane strategie liczenia punktów, również znajdowały się one na stronie podanej w opisie projektu. Implementacja prostszej z nich przebiegła skutecznie. Większym problemem była implementacja metody odpowiedzialnej za wykonywanie ruchów. Było tam dużo zmiennych do przeanalizowania i brak oczywistych strategii ruchów. Ostatecznie udało mi się opracować system oparty na sortowaniu oraz analizowaniu posiadanych meldunków. Jestem jednak pewien, że system ten nie jest najoptymalniejszy i bardzo prawdopodobne, że istnieją lepsze sposoby na stworzenie takiego algorytmu. Jednakże, metoda spełnia swoje zadanie i przeciwnik stara się meldować wszystkie posiadane meldunki oraz przebijać karty gracza. Korzystną informacją jest również fakt, że rzadko zdarza się aby komputer przelicytował swoje możliwości.