Dokumentacja

Dane autora

- Autor: Stanisław Dutkiewicz 329076

Nazwa projektu: Neutron Board Game

Opis projektu:

Projekt jest symulacją gry Neutron Board Game, która jest dwuosobową, abstrakcyjną grą strategiczną, która rozgrywana jest na planszy z kwadratami 5x5. Gracze na zmianę przesuwają swoje pionki po planszy i neutron, aby przenieść swój pionek neutronu na swoją stronę planszy. Gracz ma możliwość wybrania dwa tryby graczy komputerowych: wybierający w każdej turze losowo jeden z możliwych do wykonania ruchów i wybierający w każdej turze najlepszy ruch według pewnych prostych kryteriów. Projekt graficzny interfejs użytkownika (GUI).

Podział programu

Program podzielony jest na cztery klasy:

- NeutronBoard: Główna klasa, która reprezentuje planszę gry i zawiera metody inicjalizacji planszy, przesuwania pionków, sprawdzania zwycięzcy i innej logiki gry.
- Player: Klasa reprezentująca graczy w grze Neutron. Zawiera metody i właściwości do określania ruchów i strategii gracza.
- NeutronBoardGUI: Podklasa klasy NeutronBoard, która reprezentuje wersję graficznego interfejsu użytkownika (GUI) gry. Pozwala graczom na interakcję z planszą gry za pomocą interfejsu Tkinter.
- main: Klasa zawierająca główną funkcję, która uruchamia grę.

Opis klas

Klasa **NeutronBoard** reprezentuje planszę i jej logikę dla gry Neutron. Zawiera metody inicjowania planszy, wykonywania ruchów i sprawdzania zwycięzcy. Zawiera bieżący również stan gry, planszę i lokalizację neutronu. Klasa ma metody, aby znaleźć neutron na tablicy, zaktualizować planszę po ruchu, sprawdzić, czy ruch jest ważny i sprawdzić, czy jest zwycięzca.

Klasa **Players** reprezentuje gracza w grze Neutron. Zawiera metody i właściwości określania ruchów i strategii gracza. Ma metodę get_computer_move, która określa najlepszy ruch gracza komputerowego w oparciu o bieżącą strategię, oraz metodę get_computer_move_neutron, która jest używana, gdy komputer porusza neutronem. Posiada również metodę is_valid_move, która sprawdza, czy ruch jest ważny dla gracza.

Klasa **NeutronBoardGUI** jest podklasą klasy NeutronBoard i jest odpowiedzialna za tworzenie GUI dla gry. Obejmuje metody tworzenia przycisków, wyświetlania planszy do gry i obsługi interakcji

użytkownika z GUI. Zawiera również metodę uzyskiwania strategii komputerowej od użytkownika. Klasa tworzy okno GUI i wyświetla planszę do gry za pomocą przycisków. Zawiera również metodę obsługi kliknięć przycisków i odpowiedniej aktualizacji stanu gry. GUI mówi graczowi, jaki pionek lub neutron musi zostać przesunięty i blokuje niewłaściwy wybór elementów lub przesuwanie ich do ściany lub w inne pionki. Dodatkowo posiada metodę wyświetlania wiadomości o zwycięzcy oraz zapytania czy gracz nie chciałby zagrać jeszcze raz.

Instrukcja obsługi

Aby zagrać w grę, należy uruchomić plik main.py. Gra poprosi użytkownika o wprowadzenie strategii dla gracza komputerowego. Użytkownik może następnie nacisnąć na pionek w planszy, który chce ruszyć lub neutron w interfejsie graficznym, wybrać ruchy spośród listy umieszczonej pod spodem a następne poruszyć pionek, poprzez naciśniecie przyciska „Move".

Celem gry jest przesunięcie neutronu do swojego rzędu domowego lub sprawienie, by przeciwnik przesunął neutron do twojego rzędu domowego. Również całkowite zablokowanie neutronu, aby przeciwnik nie mógł go poruszyć kończy się zwycięstwem gracza, który zablokował neutron.

Po zakończeniu gry, gra wyświetli komunikat wskazujący zwycięzcę oraz okienko pytające o powtórzenie rozrywki.

Refleksje:

Projekt został zakończony pomyślnie i zawiera wszystkie zaplanowane funkcje. Gra może być rozgrywana w dwóch trybach: losowym i inteligentnym. W trybie losowym komputer wykonuje losowe ruchy, podczas gdy w trybie inteligentnym komputer używa algorytmu do określenia najlepszego ruchu. Gra kończy się, gdy gracz przesunie Neutron do swojego rzędu lub go całkowicie zablokuje, żeby przeciwnik by nie mógł nim poruszyć. Wersja GUI gry zawiera wyświetlanie aktualnego stanu planszy, a także przyciski do wykonywania ruchów i wychodzenia z gry.

Podczas realizacji projektu stanąłem przed pewnymi wyzwaniami. Jednym z największych wyzwań było wdrożenie algorytmu smart move dla gracza komputerowego. Trudno było wymyślić algorytm, który konsekwentnie wykonywałby najlepszy ruch. Kolejnym wyzwaniem była integracja GUI z logiką gry. Trudno było upewnić się, że GUI poprawnie odzwierciedla aktualny stan planszy i że przyciski są prawidłowo włączone i wyłączone. Również same tworzenie GUI było czasochłonne. Gdybym miał więcej czasu, spróbowałbym dodać ikonki pionków i neutrona do planszy.

Jeśli chodzi o zmiany w planowanym rozwiązaniu, pierwotny plan zakładał wprowadzenie bardziej zaawansowanych strategii AI dla komputerowego przeciwnika, ale nie było to możliwe dla mnie do zrealizowania w ramach projektu. Dodatkowo planowałem stworzyć możliwość przesuwania pionków poprzez przeciągnięcie pionka, ale to również wymaga zaawansowanej wiedzy o Tkinter o i stanąłem na tworzeniu możliwości poruszania poprzez kliknięcie na pionek, wyboru kierunku z listy i kliknięcia w przycisk do poruszania się.

Ogólnie rzecz biorąc, jestem zadowolony z wyniku projektu. Gra jest funkcjonalna i można w nią grać. Implementacja algorytmu inteligentnego ruchu dla gracza komputerowego mogłaby zostać ulepszona, ale nadal wykonuje ruchy, które są wyzwaniem dla ludzkiego gracza. Wersja GUI gry zapewnia bardziej wciągające wrażenia dla gracza.