Projekt z przedmiotu Podstawy Informatyki i Programowania

Semestr: 22Z

Autor projektu: Katarzyna Wójtowicz

Polecenie:

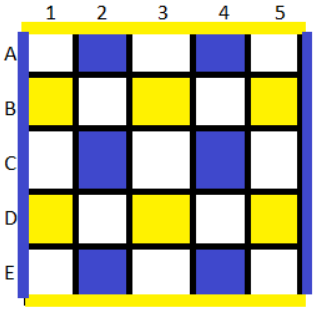
Należy stworzyć program umożliwiający rozgrywanie partii w grę "Shannon switching" w wariancie Gale między człowiekiem a graczem komputerowym. Należy stworzyć co najmniej dwa rodzaje graczy komputerowych:

- wybierający w każdej turze losowo jeden z możliwych do wykonania ruchów
- wybierający w każdej turze najlepszy ruch według pewnych prostych kryteriów

Gra może mieć interfejs terminalowy bądź okienkowy.

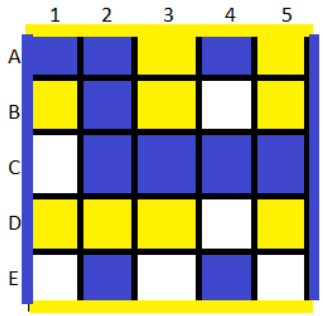
Cel i opis projektu:

Celem projektu było stworzenie programu, który umożliwia rozgrywkę w Shannon Switching Gale. Gra "Shannon switching" w wariancie Gale polega na wybieraniu przez dwóch graczy odpowiednich pól na planszy. Celem gry jest połączenie w ten sposób odpowiednich krawędzi planszy. Część pól jest wypełniona od początku gry.

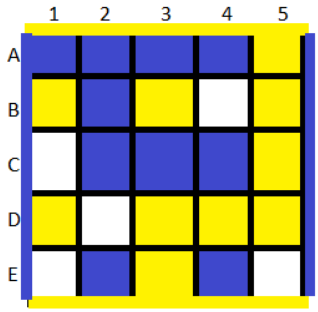


Rys1. Schemat planszy na początku gry. Białe pola są wolne i mogą zostać wypełnione przez graczy podczas gry. Kolor niebieski oznacza pola ,,zajęte'' przez pierwszego gracza, kolor żółty natomiast – przez drugiego gracza.

Pierwszy gracz dąży do połączenia polami o swoim znaku lewej i prawej krawędzi, natomiast drugi gracz górnej i dolnej krawędzi. Raz zajęte pole nie może zostać wybrane ponownie przez żadnego z graczy. Zakończenie gry następuje, gdy któryś z graczy osiągnie cel. Remis nie jest możliwy.



Rys2. Przykład planszy po grze, w której wygrał gracz pierwszy.



Rys3. Przykład planszy po grze, w której wygrał gracz drugi.

Polecenie projektu zakładało stworzenie dwóch wariantów gry: rozgrywki z graczem komputerowym wykonującym losowe ruchu oraz graczem komputerowym, który wybiera pola według prostych kryteriów. W rozwiązaniu uwzględnione zostały te dwa elementy, ale także dodany wariant gry z drugim użytkownikiem korzystającym z tego samego urządzenia dla urozmaicenia gry.

Podział programu na moduły:

1. **classes.py -** zawiera główne klasy potrzebne do rozgrywki:

- Field – klasa reprezentująca pole. Ma atrybuty położenia takie jak litera i cyfra, które umożliwiają jednoznaczne wskazanie pola podczas rozgrywki, a także atrybut znaku i atrybut- informację o tym, czy pole jest wolne.

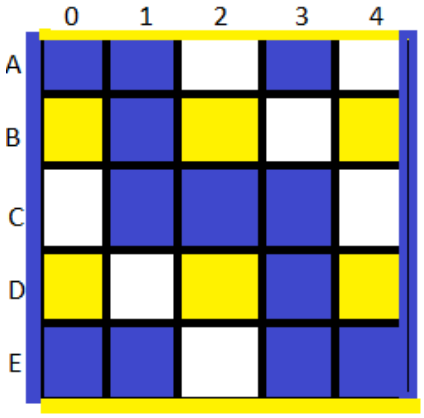
- Board – klasa reprezentująca planszę. Przechowuje słownik, gdzie kluczami są dane typu string w formacie f'{litera}{numer}, a wartościami obiekty klasy Field. Atrybutem planszy jest także jej rozmiar.

- Game – klasa reprezentuje grę jako połączenie graczy oraz planszy. Posiada metody umożliwiające sprawdzenie, czy gra została zakończona przez ,,skanowanie'' planszy od lewej do prawej lub od górnej do dolnej krawędzi.

Algorytm działania metody if_game_ended_left_right (sprawdzanie, czy gra jest wygrana przez gracza pierwszego):
Do sprawdzenia, czy gra jest skończona, posłuży lista kluczy f'{litera}{cyfra} oznaczających pola, które mogą być połączeniem lewej i prawej krawędzi. Na początku jest ona pusta.

1. Dodanie do listy wszystkich pól z kolumny 0, które mają znak gracza pierwszego (są niebieskie).
2. Sprawdzenie, czy lista jest niepusta. Jeśli nie, działanie algorytmu jest przerwane i zwracana jest informacja, że gra trwa dalej. Jeśli tak, należy przejść do punktu 3.
3. Sprawdzanie, czy któreś z pól ma współrzędną cyfry równą {wymiar planszy – 1} (dla numeracji pól od 0). Jeśli tak jest, zwracana jest informacja, że grę wygrał gracz, dla którego wygranie gry jest sprawdzane. Jeśli ten warunek nie został spełniony, to podmienia się obecną listę na listę kluczy pól, które są niebieskie i mają współrzędną cyfry o 1 większą niż pola w starej liście (dla każdego z pól z poprzedniej listy, o ile jest to możliwe.)
4. Następnym krokiem jest dodanie do obecnej listy wszystkich niebieskich pól, które z obecnymi na liście łączą się górną lub dolną krawędzią. Następuje powrót do punktu 2 algorytmu.

Dla planszy z Rys4. Algorytm przebiegnie następująco:
1.Lista: A0, E0
2. Lista: A0, E0
3. Lista: A1, E1
4. Lista: A1, E1, B1, C1, F1, G1
2. Lista: A1, E1, B1, C1, F1, G1
3. Lista: C2
4. Lista: C2
2. Lista: C2
3. Lista: C3
4. Lista: C3, D3, E3
2. Lista: C3, D3, E3
3. Lista: E4
4. Lista: E4
2. Lista: E4
3. Lista: E4, wygrywa gracz o znaku niebieskim, ponieważ współrzędna pola '4' jest równa (wymiar planszy – 1).



Rys4. Plansza, w której wygrywa gracz pierwszy.

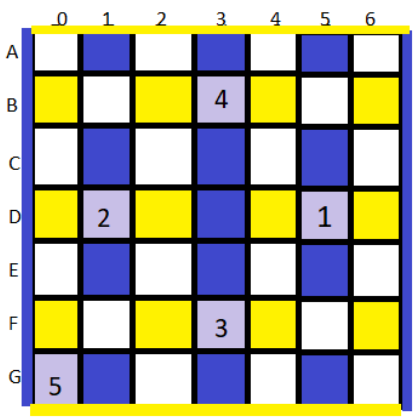
Klasa zawiera analogiczną metodę sprawdzania wygranej gracza dążącego do połączenia górnej i dolnej krawędzi planszy.

- GameRun – klasa reprezentuje rozgrywkę, przechowuje informacje o graczach, trybie gry, oraz samej grze (obiekt klasy game), a także wyniku gry. Posiada metody, które umożliwiają przeprowadzenie rozgrywki, zawiera informacje o przebiegu gry.

2. **Human_Player.py** – zawiera klasę HumanPlayer, które reprezentuje gracza, wybierającego pola wpisując ręcznie jego współrzędne.
3. **RandomComputerPlayer.py** – zawiera klasę RandomComputerPlayer, która reprezentuje gracza wybierającego pole losowo.

4. **HeavyComputerPlayer.py** – zawiera klasę HeavyComputerPlayer, która reprezentuje gracza wybierającego pole według prostych kryteriów.

Kryteria wyboru pola – odpowiednia metoda umożliwia wybór pola w zależności od pola wybranego ostatnio przez przeciwnika. W pierwszej kolejności zajmowane jest pole o dwa na prawo od punktu odniesienia. Jeśli nie jest to możliwe, sprawdzane są kolejno pola: o dwa na lewo, o dwa w dół, o dwa w górę. Jeśli zajęcie żadnego z tych pól nie jest możliwe, wtedy wybierane jest pierwsze wolne losowo wybrane pole.



Rys5. Kolejność zajęcia fioletowych pól według algorytmu zawartego w metodzie klasy HeavyComputerPlayer. Ostatnim wyborem przeciwnika było pole D3.

5. **lets_play.py** – zawiera funkcję main, która pobiera od użytkownika potrzebne informacje, takie jak rozmiar planszy (aby wyświetlała się poprawnie musi być to 5, 7 lub 9), tryb gry, nazwy graczy. Tworzy obiekt klasy GameRun, po przeprowadzeniu rozgrywki zwraca jej wynik. Uruchomienie modułu przez interpreter powoduje rozpoczęcie rozgrywki.

Instrukcja użytkownika:

Należy uruchomić przez interpreter plik lest_play.py i postępować według wyświetlających się komunikatów (wybrać tryb gry - przeciwnika, rozmiar planszy, wpisać imię lub imiona graczy), następnie rozpocznie się gra.

Podsumowanie:

Polecenie projektu zostało spełnione, program został także wzbogacony o dodatkowy tryb gry z innym użytkownikiem. Udało się osiągnąć pełną funkcjonalność, zabezpieczenie przed podaniem przez użytkownika niewłaściwych danych. Program został sprawdzony wieloma testami. Przeprowadzone zostało także wiele rozgrywek, co dowodzi, że program działa bezbłędnie i umożliwia rozegranie partii w grę Shannon Switching Gale. Kluczowe dla projektu było opracowanie algorytmu skanowania planszy. Dostępne są trzy rozmiary planszy, co pomaga dostosować długość rozgrywki do preferencji użytkownika.

Do dopracowania jest natomiast interfejs użytkownika, należałoby stworzyć np. bardziej czytelny niż terminalowy interfejs graficzny. Początkowo stworzone zostały metody umożliwiające wybór rozmiaru planszy, który ograniczała tylko ilość znaków alfabetu angielskiego, jednak pojawiła się potrzeba redukcji tej możliwości, ze względu na problem przedstawienia w terminalu planszy z taką liczbą znaków (problem z liczbami dwucyfrowymi, których szerokość byłaby większa niż szerokość kolumny).