Ultimate Tic-Tac-Toe - Dokumentacja

Opis projektu

Treść zadania

```
Należy stworzyć program umożliwiający rozgrywanie partii w kółko i krzyżyk na sterydach między człowiekiem a graczem komputerowym.
Należy stworzyć co najmniej dwa rodzaje graczy komputerowych:
1. wybierający w każdej turze losowo jeden z możliwych do wykonania ruchów
2. wybierający w każdej turze najlepszy ruch według pewnych prostych kryteriów
```

Opis projektu

Projekt jest grą "Ultimate tic-tac-toe". W skrócie gra składa się z 9 plansz 3x3 (w klasycznym wymiarze), gdzie ruch poprzedniego gracza wpływa na ruch kolejnego. Wymiar planszy jest wartością ustawialną (liczba nieparzysta niemniejsza niż 3), a gracz ma możliwość gry zarówno z innym graczem, jak i z dwoma botami: losowym oraz opartym na pewnym algorytmie. Gra jest realizowana w terminalu, działa zarówno dla systemów Linux jak i Windows.

Opis klas

Program został podzielony na kilka klas opisanych poniżej:

Board

Klasa w pliku `boards.py` zwykłej planszy; nieużywana bezpośrednio w grze, jednak jest ona rodzicem dla klas *BigBoard* oraz *SmallBoard* oraz zawiera metody przydatne dla obu z tych klas. Nadaje się ona również do rozegrania klasycznej gry w kółko i krzyżyk na pojedynczej planszy. Do najważniejszych funkcji należą:

- `make_move` - wykonująca ruch na planszy
- `check_winner` - sprawdzająca zwycięzcę

BigBoard

Klasa w pliku `boards.py`, której obiekt jest główną planszą do gry. Zawiera w sobie tyle obiektów klasy *SmallBoard*, ile jest małych plansz do gry. Poza informacjami o samej sobie, zawiera także informacje o małych planszach, między innymi ich status aktywności. Funkcja `make_move` w tej planszy polega na wykonaniu ruchu w grze, także na małej planszy.

SmallBoard

Klasa w pliku `boards.py`, której obiekty są przechowywane w obiekcie klasy *BigBoard*. Cała logika obiektów tej klasy odbywa się w komunikacji z tą klasą, w której jest przechowywana.

RandomBot

Klasa w pliku `bots.py`, której obiekt jest botem wykonującym kompletnie losowy ruch na planszy.

SmartBot

Klasa w pliku `bots.py`, której obiekt wykonuje ruch na podstawie następującego algorytmu:

1. Wczytaj listę pól, na których wykonanie ruchu jest możliwe
2. Jeżeli wybierając któreś pole możesz wygrać grę, wybierz je - **KONIEC**
3. Jeżeli wybierając któreś pole możesz wygrać planszę, wybierz je - **KONIEC**
4. Jeżeli przeciwnik może wygrać planszę w następnym ruchu po wyborze tego pola, usuń je z możliwych
5. Jeżeli przeciwnik może wybrać pole na dowolnej planszy po wyborze tego pola, usuń je z możliwych
6. Wylosuj ruch z dostępnych pól <- jeżeli nie ma takich dodaj do puli pola ostanio usunięte (kolejno w 5 i 3 kroku) - **KONIEC**

Gameplay

Klasa w pliku `main.py` komunikująca się z logiką i wypisującą interfejs. Do ważniejszych funkcji należą:

- `get_size` i `get_mode` - funkcje pobierające rozmiar i tryb gry
- funkcje typowo wypisujące odpowiednie dane, takie jak `generate_board`
- `next_move` - funkcja odpowiadająca za pobranie ruchu od gracza

Klasy wyjątków

Znajdują się w `errors.py`. Służą one do łatwego wychwytywania błędów w wypadku podania nieprawidłowych danych przez gracza.

Instrukcja

Aby uruchomić program, należy uruchomić plik `main.py` w interpreterze Pythona. Do uruchomienia programu wymagany jest jedynie Python z bibliotekami standardowymi. Do prawidłowego działania potrzebne są pliki:

- `main.py`
- `boards.py`
- `bots.py`
- `errors.py`

Refleksje

Zakres prac

Projekt ten składał się z następujących części:

- logiki gry znajdującej się na planszach (`boards.py`)
- interfejsu oraz procesu rozgrywania całej gry (`main.py`)
- dwóch sposobów losowego wykonywania ruchów (`bots.py`)

Trudności

- Początkowo zakładałem wykonanie swojego projektu w PyQt. Niestety, napotkałem problemy m. in. przy skalowaniu interfejsu oraz zachowaniu pól kwadratami. Uznałem, że lepszym pomysłem będzie ostatecznie wykonanie projektu w terminalu.
- Program dla rozmiaru większego niż 5 brzydko się formatuje, teoretycznie można było próbować skondensować interfejs, jednak odbyłoby się to kosztem czytelności, także dla mniejszych rozmiarów. Niestety, jest to jedna z wad programów w terminalu i na to nic nie mogłem poradzić.
- Bot bazujący na algorytmie z racji na swoją złożoność działa zbyt wolno na rozmiarach większych niż 7, również dla rozmiaru 7 potrafi "chwilę myśleć".

Słowo końcowe

Podsumowując, projekt ten pozwolił mi rozwinąć swoje umiejętności pracy nad projektem posiadającym wiele plików oraz korzystania z Gita (a także pracy pod presją czasu ☺).
