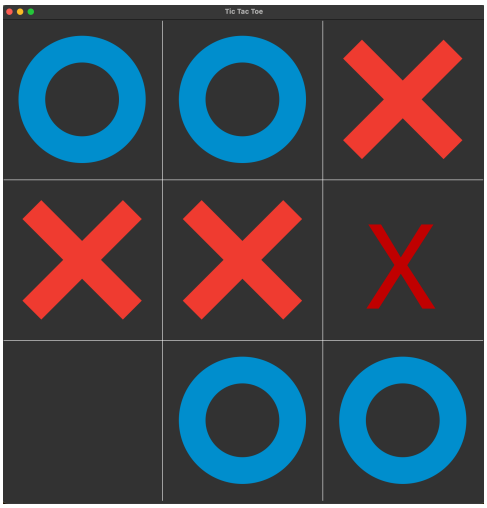
Sprawozdanie Lab 3

Jakub Kwaśniak 331396

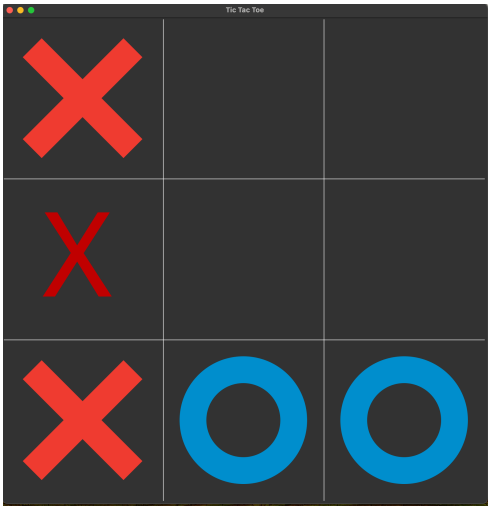
1. Treść ćwiczenia
    - Zaimplementować algorytm minimax z obcinaniem α − β grający w kółko i krzyżyk, używając przygotowanego w tym repozytorium kodu.
    - Przeprowadzić następujące symulacje gier próbując znaleźć jak najlepsze parametry algorytmu minimax:
        - Gracz minimax vs gracz losowy
        - Gracz minimax vs gracz minima
    - Rozegrać samemu kilka gier przeciwko:
        - Graczowi losowemu
        - Graczowi minimax ze znalezionymi parameterami
    - Zbadać eksperymentalnie wpływ głębokości odcinania drzewa gry.

2. Cel i opis eksperymentów oraz wyniki (do zdjęć dopisywany jest ostatni ruch gracza w rozgrywce)
    - Gracz minimax korzysta z heurystyki przypisującej każdemu polu na planszy wartość równą ilości wygranych kombinacji w których to pole jest używane
    - Gracz x minimax (depth=5) vs gracz o random – wygrywa gracz x minimax, gdy zaczyna gracz O gracz minima wykonuje więcej ruchów by wygrać

Zaczyna O

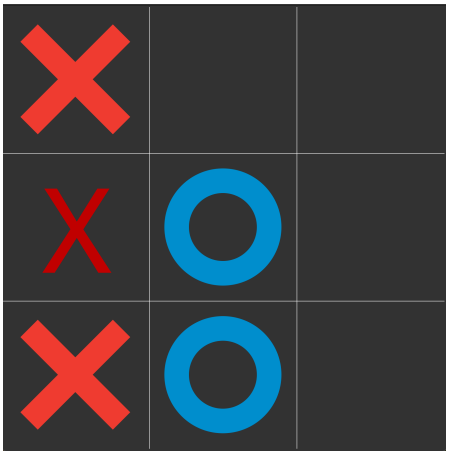


Zaczyna X

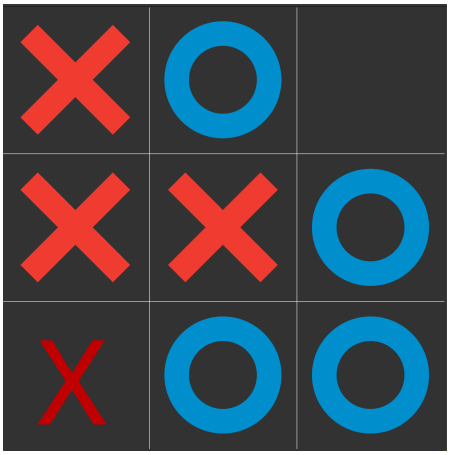


- Gracz x minimax (depth=6) vs gracz o random - wygrywa gracz x minimax, gdy zaczyna gracz O gracz minima wykonuje więcej ruchów by wygrać

Zaczyna X

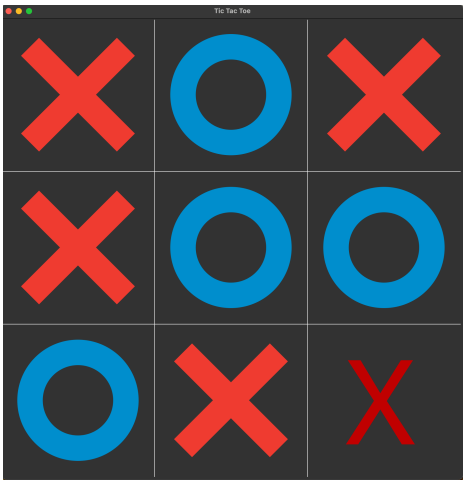


Zaczyna O

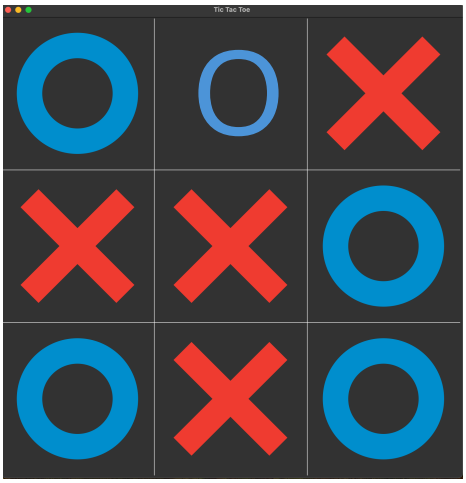


- Gracz x minimax (depth=6) vs gracz o minimax (depth=6) – zawsze tie, obaj gracze grają optymalnie więc żaden nie wygra niezależnie kto rozpoczął rozgrywkę

Zaczyna X

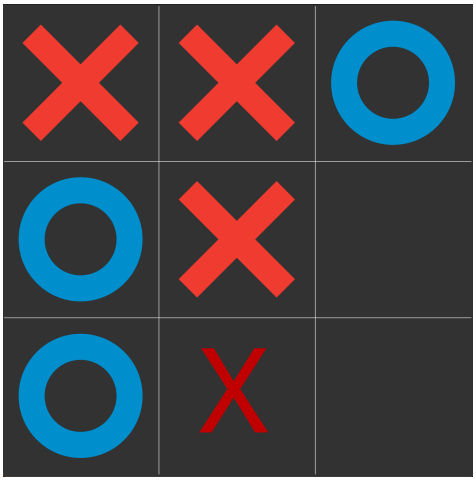


Zaczyna O

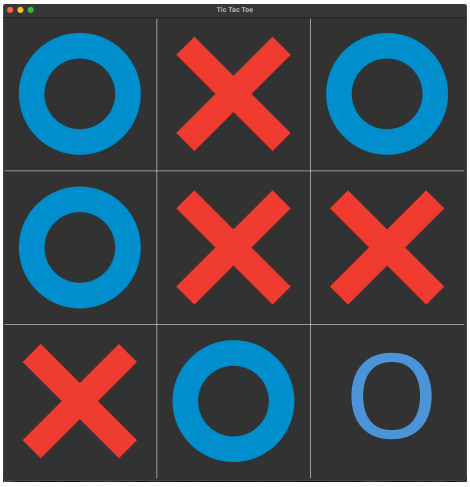


- Gracz x minimax (depth=6) vs gracz o minimax (depth=2) – gracz o małym depth przegrywa gdy gracz o wyższym depth rozpoczyna grę a gdy sam zaczyna doprowadza do remisu

Zaczyna X

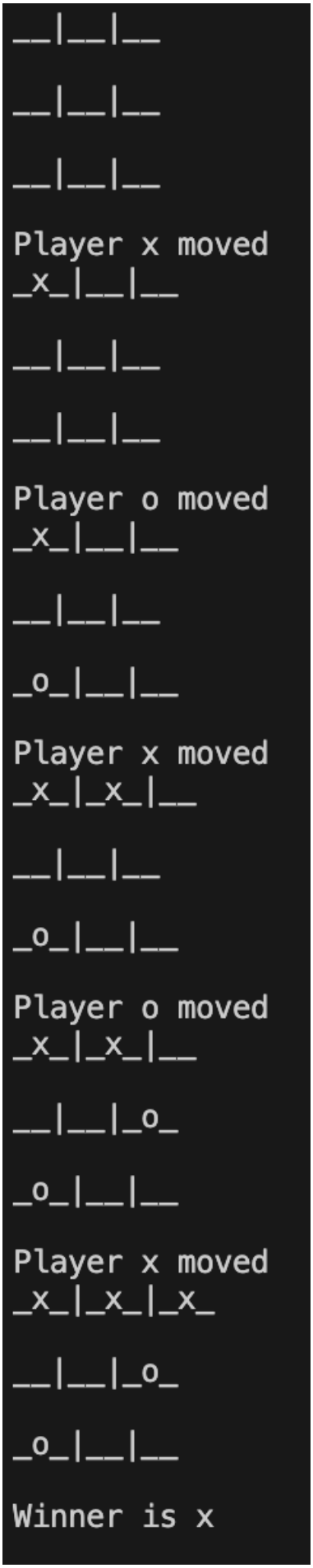


Zaczyna O

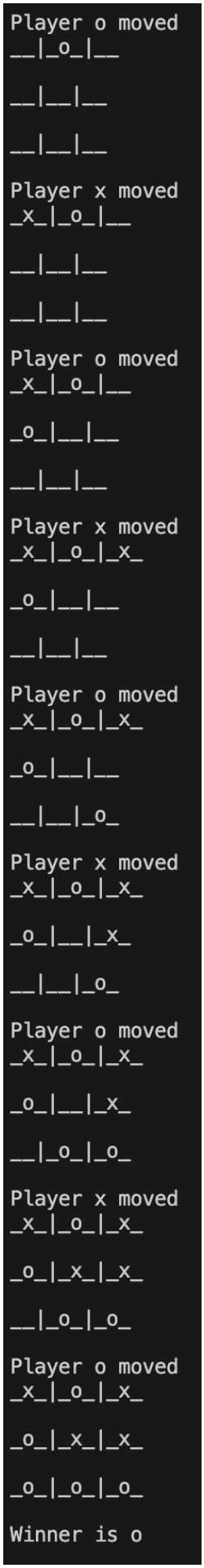


- Gracz x minimax (depth=2) vs gracz random o

Zaczyna X



Zaczyna O



3. Instrukcja potrzebna do odtworzenia wyników (wraz z przygotowaniem środowiska, danych)
   Skrypt należy uruchomić komendą
   python3 main.py --config config.json
   - config.json – plik z zadeklarowanymi graczami sformatowany w podany sposób:

```
{
        "x": {
                "type": "minimax",
                "depth": 5
        },
        "o": {
                "type": "random"
        },
        "gui": false
}
```

   - x, o - gracze gry w kółko i krzyżyk,
   - type – określa jakim typem gracza jest x / o, przyjmuje jedną z 3 możliwych wartości [minimax, human, random] odpowiednio ‘gracz algorytm minimax alpha-beta’, ‘gracz człowiek’, ‘gracz losowy’,
   - depth - głębokość odcinania drzewa gry, należy ją podać w przypadku wybrania gracza minimax
   - gui – dla wartości true rozgrywka odbywa się w interfejsie graficznym, dla false w interfejsie tekstowym

4. Wnioski:
   - Optymalna wartość głębokości odcinania drzewa gry dla planszy 3x3 to depth=6, przy tej wartości algorytm minimax osiąga bardzo dobre wyniki, wygrywa z graczem grającym nieoptymalnie (nieoptymalnie chociaż w jednym ruchu) oraz doprowadza do remisu w rozgrywce z graczem grającym optymalnie – w eksperymentach nie było przypadku chociażby 1 porażki.
   - Gracz minimax o niskim depth np. depth=2 nie jest w stanie wygrać z innym graczem grającym optymalnie ale jeśli sam zaczyna to broni się doprowadzając często do remisu , jeśli natomiast zaczyna gracz grający optymalnie to przegrywa z nim.
   - Z kolei gdy gracz minimax o niskim depth np. depth=2 gra z innym graczem grającym nieoptymalnie np. random - wówczas minimax wygrywa większość rozgrywek ale mogą zdarzyć się rozgrywki - np. te które rozpoczyna gracz o (np. random) – które gracz o wygrywa lub doprowadza do remisu – są to sporadyczne przypadki wynikające z niedostatecznie dużej głębokości drzewa przeszukiwań oraz faktu że w trakcie przeszukiwania drzewa ruchów zakładamy że obaj gracze grają optymalnie – co nie jest prawdą w tym przypadku.
   - W niektórych rozgrywkach gracz minimax wygrywa rozgrywkę w nie najmniejszej możliwej liczbie ruchów – dzieję się tak gdyż wybierany jest ruch który jako pierwszy

został oceniony za najlepszy nawet jeśli w późniejszym przeszukiwaniu znaleziono ruchy o równej ocenie, nie wpływa to na wyniki ale w celu optymalizacji można wypróbować następującą metodę - np. od znalezionej wartości ruchu odejmować głębokość drzewa przeszukiwań na której oceniono ruch.

- Gracz człowiek nie może wygrać z graczem minimax o dobrze dobranej wartości depth, może jedynie doprowadzić do remisu grając jedynie optymalnie, z kolei gracz random gra z zasady nieprzewidywalnie więc i nieoptymalnie dlatego gracz random w większości przypadków przegrywa z optymalnie grającym minimax (takim o dobrze dobranym depth).