Programowanie
Projekt – *Kółko i krzyżyk*

Karol Zając

Politechnika Śląska

Wydział Matematyki Stosowanej

Informatyka stopień I, rok II, semestr III

2023/2024

1. Temat projektu

Tematem projektu jest zadanie 2 z edycji 2015 konkursu Algorytmion o nazwie „Kółko i krzyżyk”. Celem zadania jest stworzenie programu, który gra z użytkownikiem w grę kółko i krzyżyk. Przeciwnik kontrolowany przez komputer zawsze musi obierać strategię prowadzącą do jego wygranej lub co najmniej remisu.

2. Opis danych pobieranych przez program

Wszystkie dane potrzebne do działania programu są pobierane od użytkownika za pośrednictwem przycisków znajdujących się w oknach programu.

Okno wyboru figury:

- Użytkownik wybiera figurę, którą będzie grać za pomocą dwóch przycisków w oknie odpowiadającym kółku i krzyżykowi.

Okno gry:

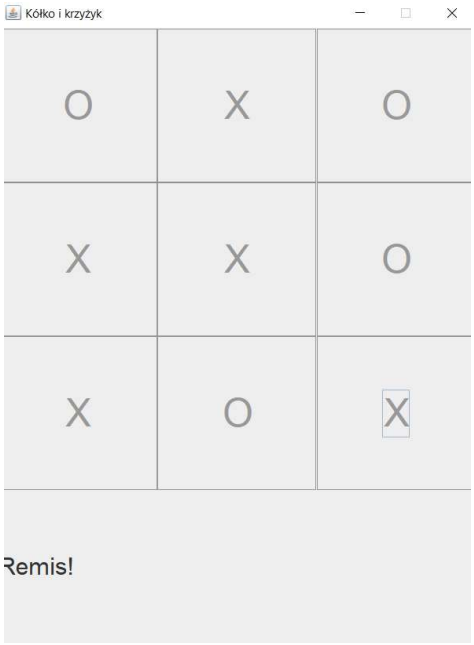
- Użytkownik wybiera, w którym miejscu na planszy chce umieścić swoją figurę naciskając na pożądane miejsce na planszy. Ruch jest rejestrowany, jeżeli wskazane pole jest dostępne.

3. Opis otrzymywanych rezultatów

Po uruchomieniu programu, wyświetla się okno wyboru.

Zrzut ekranu 1 - okno wyboru

Następnie wyświetla się okno z planszą gry. Pod planszą wyświetlana jest informacja o stanie gry.



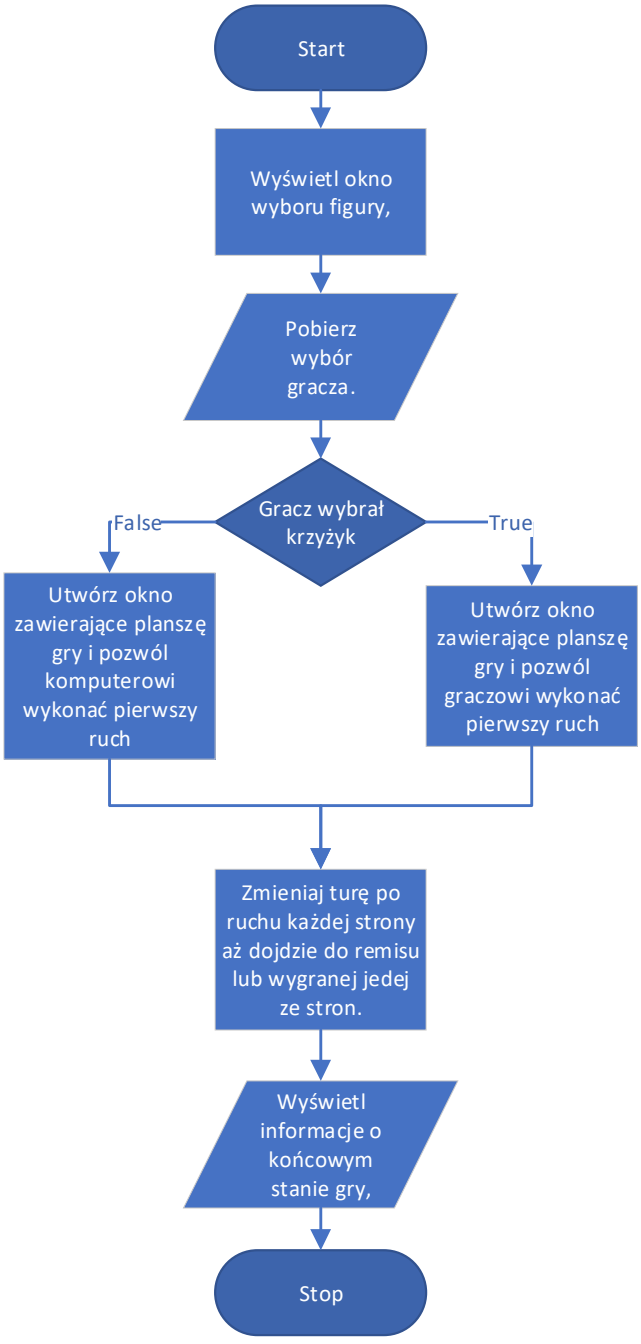
Zrzut ekranu 2 - okno gry

4. Algorytm zastosowany do rozwiązania zadania

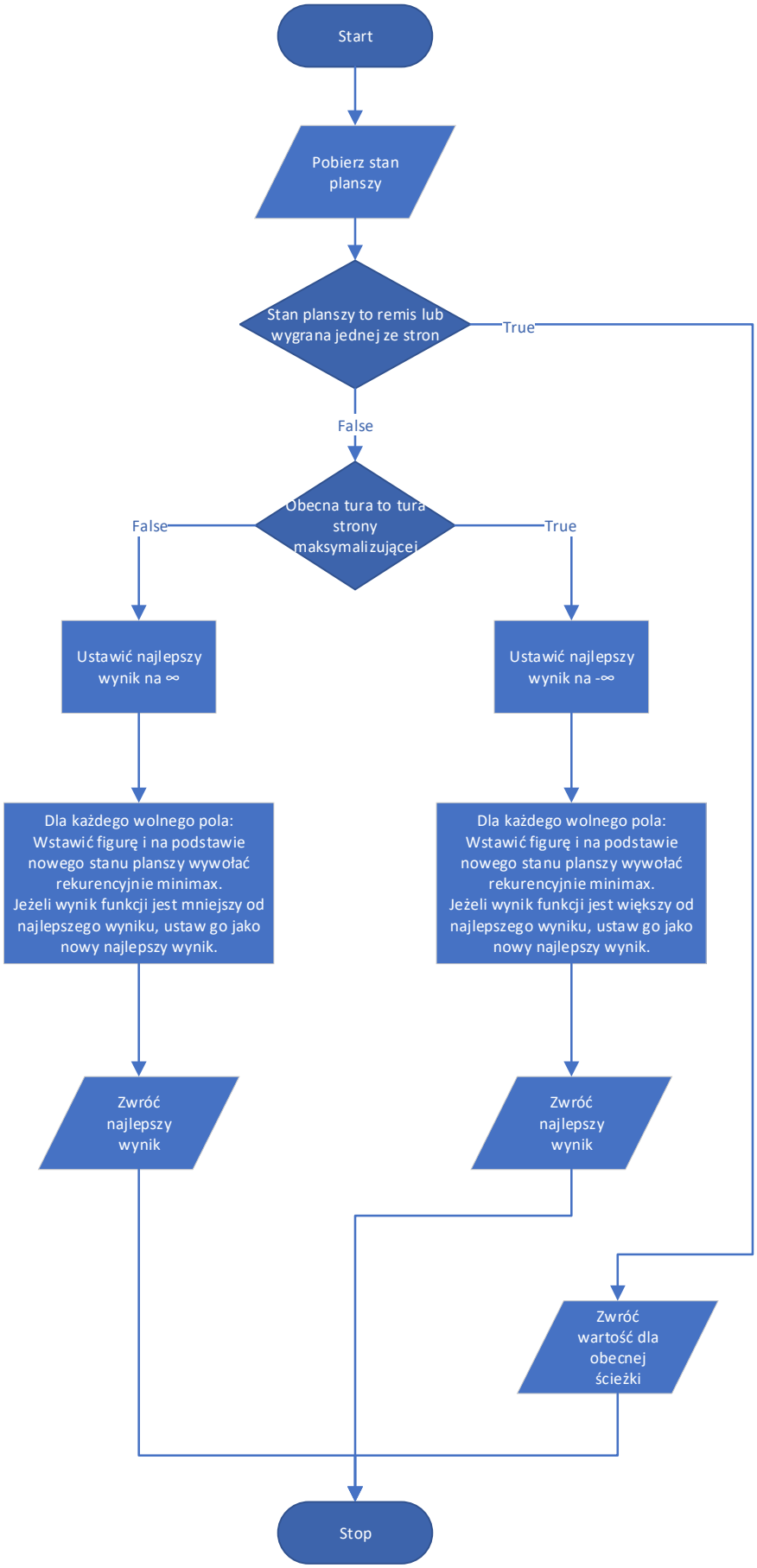
a) słowny opis wykorzystanych algorytmów

W celu rozwiązania zadania w projekcie wykorzystano algorytm minimax. Algorytm na podstawie obecnego stanu planszy sprawdza kolejne możliwe stany i przypisuje im wartość, w zależności od tego, czy są pożądane do zakończenia gry wygraną lub remisem. Następnie zwraca najlepszy możliwy ruch dla obecnego stanu planszy. Kolejne stany planszy można przedstawić w formie drzewa.

b) schemat blokowy programu



Rysunek 1 - schemat obrazujący ogólne działanie programu



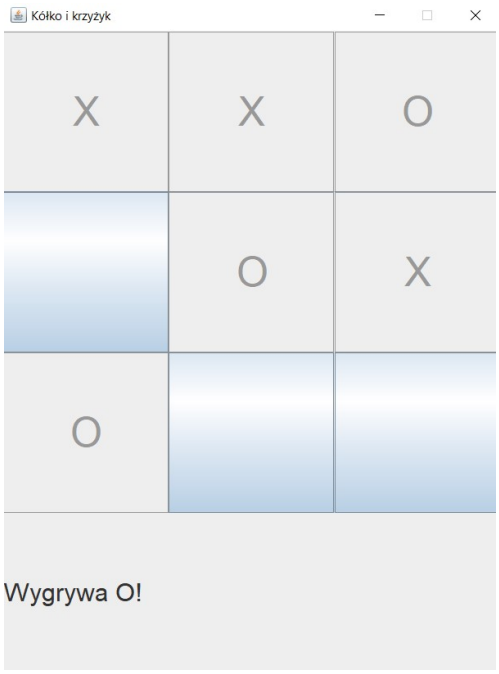
Rysunek 2 - schemat pokazujący działanie zastosowanego algorytmu minimax

c) Zastosowane klasy i metody w programie

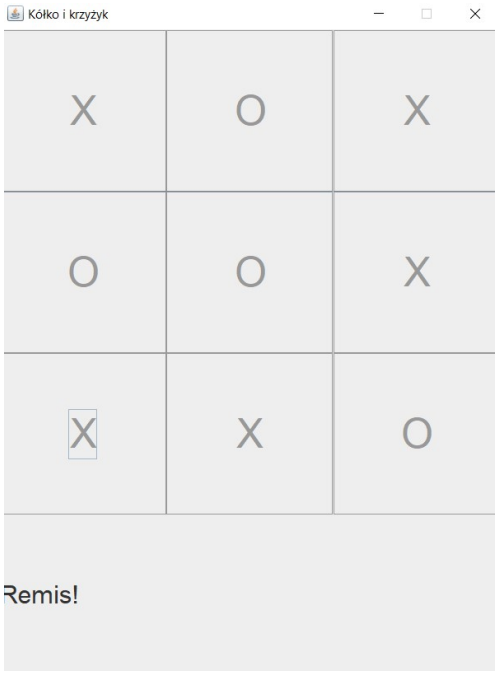
- **Klasa „Main”** – klasa tworzy okno wyboru figury.
- Klasa „ChoiceWindow” – klasa zawierająca okno wyboru figury. W celu utworzenia okna wykorzystano bibliotekę Swing i AWT. Klasa zawiera dwa przyciski JButton oraz JLabel, gdzie wyświetlana jest prośba o wybór figury. Klasa zawiera metodę „actionPerformed”, która rozpoczyna grę w zależności od wyboru użytkownika, tworząc obiekt klasy „Frame”.
- **Klasa „Frame”** – klasa zawiera okno z planszą gry. Plansza złożona jest z dziewięciu obiektów JButton, pod którymi znajduje się obiekt JLabel zawierający informacje o stanie gry. Klasa zawiera metodą „setUpFrame”, tworzącą okno gry i metodę „actionPerformed”, rejestrującą ruchy gracza i modyfikującą na ich podstawie planszę. Klasa zawera również obiekty klas „TicTacToe” oraz „AI” potrzebne do przeprowadzenia rozgrywki.
- **Klasa „TicTacToe”** – klasa zawierająca najważniejsze informacje o stanie gry. W tablicy „board” przechowuje informacje o obecnym stanie planszy, zmienna „turn” informuje o tym, która strona wykonuje ruch i zmienna „playerTurn” informuje o figurze, jaką wybrał gracz, a zmienna „gameOver” o tym, czy gra się skończyła. Klasa zawiera metody „checkWin” oraz „checkTie”, sprawdzające czy dla danego układu planszy któraś ze stron wygrała lub doszło do remisu. Klasa zawiera metody zwracające informacje o obecnym stanie gry oraz umożliwiające jego modyfikacje.
- **Klasa „AI”** – klasa zawierająca przeciwnika kontrolowanego przez komputer. Klasa zawiera metodę „makeMove” umożliwiającą komputerowi wykonanie ruchu na planszy. Metody „bestMoveMaximize” oraz „bestMoveMinimize” na podstawie przesłanego stanu planszy wybierają najlepszy ruch dla gracza maksymalizującego lub minimalizującego. Obie metody wywołują metodę „minimax” sprawdzającą kolejne ułożenia planszy.

5. Testy na poprawność działania programu

W celu przetestowania poprawności działania programu, przeprowadzono kilkadziesiąt rozgrywek gracza z przeciwnikiem kontrolowanym przez komputer. Sprawdzono działanie używając najlepszej strategii jak również popełniając błędy. W pierwszym przypadku rozgrywka zawsze kończyła się remisem, a w drugim wygraną komputera. Testy wykonano zarówno grając kółkiem, jak i krzyżykiem.

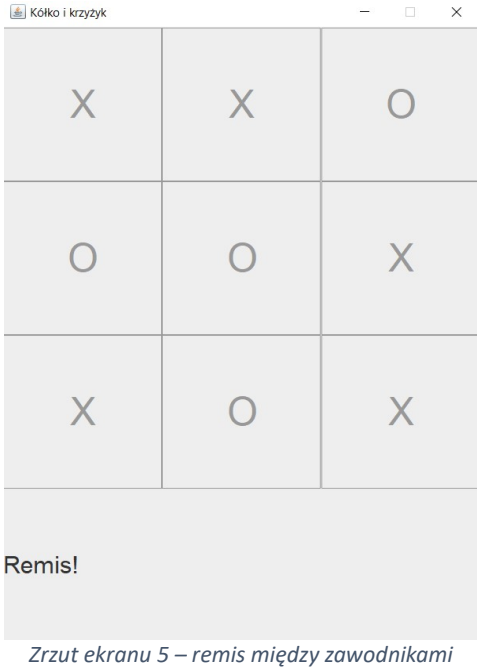


Zrzut ekranu 3 - wygrana komputera

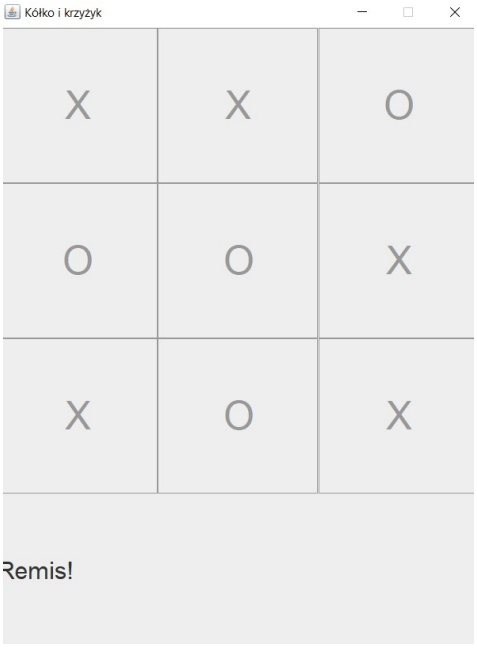


Zrzut ekranu 4 - remis

W celu przetestowania poprawności działania algorytmu minimax zmodyfikowano program i przeprowadzono kilkanaście rozgrywek między dwoma zawodnikami kontrolowanymi przez komputer. W tym przypadku, tak jak się spodziewano, każda rozgrywka zakończyła się remisem.



Zrzut ekranu 5 – remis między zawodnikami kontrolowanymi przez komputer



Zrzut ekranu 6 - kolejny remis między zawodnikami kontrolowanymi przez komputer

6. Wnioski

Jak zamierzono, w ramach projektu stworzono program grający z użytkownikiem w kółko i krzyżyk. Program zawsze wygrywa lub doprowadza do remisu, w zależności od strategii użytkownika. Wykorzystano w tym celu algorytm minimax, który okazał się dobrym wyborem dla tego zastosowania.

Istnieje wiele możliwości rozwoju projektu:

- Zaimplementowanie algorytmu alfa-beta. Jest to algorytm, który pozwala na zredukowanie liczby węzłów, które komputer musi rozważyć podczas wykorzystywania algorytmu minimax. Chociaż w przypadku gry tak prostej jak kółko i krzyżyk nie jest to niezbędne, pozwoliłoby na znaczne poprawienie wydajności programu przez wyeliminowanie stanów gry, które nie prowadzą do pożądanego rezultatu.
- Dodanie opcji wybrania rozgrywki między dwoma graczami, dwoma komputerami oraz w przypadku rozgrywki między użytkownikiem a komputerem, dodanie poziomów trudności (np. łatwy – komputer wybiera ruchy losowo, średni – komputer czasem popełnia błędy, trudny – komputer nigdy nie popełnia błędów).
- Uogólnienie wykorzystanego algorytmu, aby możliwa była gra na większych planszach, np. 5x5 lub 7x7.
- Uatrakcyjnienie interfejsu graficznego programu. Możliwe jest dostosowanie okien gry w taki sposób, aby był on bardziej czytelny dla użytkowniku i przyjemny w użytkowaniu.