**Dokumentacja techniczna – Tetris**

**Przygotował: Kacper Szczerba Gr.7   
nr. indeksu 135192**

**1. Wprowadzenie**

Gra Tetris jest klasyczną grą logiczną, w której celem gracza jest układanie spadających klocków w taki sposób, aby tworzyć pełne poziome linie. Gra kończy się, gdy klocki dotkną górnej krawędzi planszy. Stworzona przez Aleksieja Pażytnowa i jego współpracowników, Dimitrija Pawłowskiego i Wadima Geriasimowa. Pojawiła się na rynku po raz pierwszy 6 czerwca 1984 roku w Związku Radzieckim.

Oryginalna wersja powstała podczas pracy zespołu Pażytnowa w Akademii Nauk ZSRR w Moskwie na komputerze Elektronika 60. Jest to jedna z najbardziej znanych gier komputerowych, posiadająca dużą liczbę różnorodnych wariacji i wariantów.

Gra rozpoczyna się na prostokątnej planszy (początkowo pustej) zwanej tetrionem lub matriksem, ułożonej krótszym bokiem w poziomie. Tetrion ma wymiary 20 wierszy na 10 kolumn. W trakcie gry, pośrodku górnej krawędzi planszy, pojawiają się pojedynczo klocki złożone z czterech małych kwadratów nazywanych też blokami. Klocki te (określane mianem „tetrimino”) przemieszczają się w kierunku dolnej krawędzi w miarę możliwości. Kiedy jedno tetrimino opadnie na samo dno, zostaje unieruchomione, a następne ukazuje się u góry planszy. Gra trwa aż do momentu, w którym klocek nie będzie mógł pojawić się na planszy.

Zadaniem gracza jest układanie tetrimino na planszy (poprzez wykorzystanie rotacji   
i przesuwanie klocków w poziomie) w taki sposób, aby kwadraty składające się na nie utworzyły wiersz na całej szerokości prostokąta. W takiej sytuacji wiersz ten zostaje usunięty, a pozostałe klocki opadają w kierunku dna, tworząc więcej przestrzeni dla następnych elementów. Po usunięciu określonej liczby wierszy prędkość gry wzrasta o pół sekundy, co utrudnia tym samym precyzyjne sterowanie kolejnymi tetrimino. Możliwe jest jednoczesne usunięcie maksymalnie 4 wierszy – umożliwia to tetrimino „I”. Sytuacja taka nosi nazwę identyczną jak gra, czyli „tetris”.

**2. Architektura aplikacji**

Aplikacje stworzyłem w programie Visual Studio 2022 przy użyciu języka C# oraz aplikacji WPF.

**Aplikacja Tetris składa się z kilku głównych komponentów:**

**MainWindow -** główne okno gry, które wyświetla planszę i obsługuje logikę gry.

**GameGrid-** klasa reprezentująca plansze gry.

**GameState -** klasa odpowiadająca za aktualny stan gry.

**BlockQueue -** klasa odpowiedzialna za zarządzanie kolejką klocków.

**Position -** klasa przechowująca wiersze i kolumny w grze(Potrzebna do określenia pozycji klocków i ich poruszania się).

**Block –** klasa obsługująca pojawianie się klocków, ich rotację oraz poruszanie się.

Oraz klas bloków które się zawierają w grze Tetris: **IBlock, JBlock, LBlock, OBlock, SBlock, TBlock, ZBlock.**

**3. Klasa MainWindow**

Klasa GameWindow jest głównym oknem gry i dziedziczy po klasie Window z biblioteki WPF. Odpowiada za inicjalizację i wyświetlanie planszy gry oraz obsługę logiki gry.

Posiada ona ładowanie tekstur bloków oraz ich kolorów. Kolejność ich wprowadzania nie jest przypadkowa. Odpowiada ona indeksowaniu, czyli kolory i kształty klocków zgadzają się   
z id przypisanym danym klockom.

Przechowuje podwójną tabele

**Metody jakie się w niej znajduą to:**

**SetupGameCanvas() –** pozwala na ustawienie imageControls bezpośrednio w canvie. Canva została ustawiona na szerokość 250px i wysokość 500px, co daje nam wielkość kratki 25px.  
Dwa wiersze zostały ukryte, w celu umożliwienia pojawienia się bloków.

**DrawGrid()-** przechodzi przez wszystkie pozycje. Dla każdej pozycji mamy id i ustawa źródło obrazu na tę pozycję poprzez jego id.

**DrawBlock()-** przechodzi przez TilePosition i aktualizuje źródła obrazu tak jak w poprzedniej metodzie.

**DrawNextBlock()-**sprawia, że w oknie pokazuje nam który klocek pojawi się jako następny.

**DrawHeldBlock()-** sprawia, że po lewej stornie w oknie pokazuje nam czy trzymamy jakiś klocek i jaki klocek trzymamy.

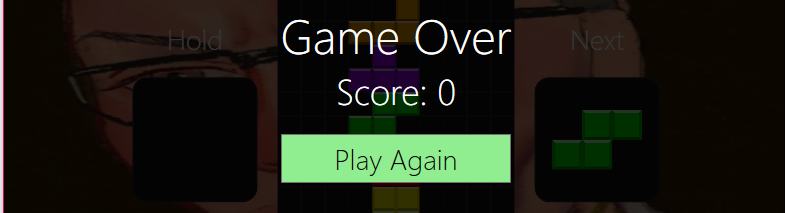
**DrawGhostBlock()-** sprawia, że możemy zobaczyć gdzie klocek wyląduje .

**Draw()-** rysuje grida i aktualny klocek w użyciu.

**GameLoop()-**sprawia, że blok się porusza w dół. Zawiera w sobie menu GameOver

**Window\_KeyDown()-** sprawia, że po zakończeniu gry wciśnięcie dowolnego przycisku nic nie zrobi. Przechowuje w switch casie klawisze, za pomocą których poruszamy klockami.

**GameCanvas\_Loaded()-** ładuje canve do gry.

**PlayAgain\_Click()-** jak sama nazwa wskazuje metoda tworzy nowy przycisk, którego zadaniem będzie zrestartowanie gry po naciśnięciu przycisku.  


**4. Klasa GameGrid**

Klasa GameGrid reprezentuje plansze gry. Przechowuje ona Dwuwymiarową prostokątną tablice. Pierwszym wymiarem jest wiersz, a drugim kolumny. Definiuje również indeksowanie.  
Konstruktor będzie brał liczę kolumn i wierszy, a następnie traktował je jako parametry. Pozwoli to na tworzenie nieregularnych i różnych wersji Tetrisa.

**Metody jakie się w tej klasie znajdują:**

**IsInside()-** sprawdza, czy podany wiersz i kolumna znajdują się w gridzie czy też nie.

**IsEmpty()-** sprawdza czy podana kratka jest pusta czy też nie.

**IsRowFull()-** sprawdza czy cały wiersz jest pełny

**IsRowEmpty()-** sprawdza, czy cały wiersz jest pusty.

**ClearRow()-** jeśli wiersz jest pełny to wyczyści go.

**MoveRowDown()-** jeśli wiersz został wyczyszczony, reszta wierszy zostanie przeniesiona niżej o liczbę wierszy, które zostały wyczyszczone.

**ClearFullRows()-** sprawdza, czy wiersz jest pełny, a następnie usuwa pełne wiersze od dołu do góry i pozostałe przenosi na sam dół.

**5.Klasa GameState**

Klasa GameState odpowiada za aktualny stan gry.   
Posiada ona właściwość z tylnym polem bieżącego bloku. Kiedy aktualizujemy currentBlock, reset jest wywoływany i klocek jest ustawiany w dobrej pozycji.  
Posiada deklaracje tablicy gry na 22 wiersze i 10 kolumn(normalna plansza Tetrisa jest 20 na 10. Dwa dodatkowe wiersze pozostaną ukryte i pozwolą na generowanie nowych klocków)  
W konstruktorze GameState() przechowywane są właściwości aktualnej gry.

**Metody jakie się w tej klasie znajdują:**

**BlockFits()-** sprawdza, czy klocek jest w dozwolonej pozycji, czy też nie.

**HoldBlock()-** pozwala na przetrzymanie bloku, który właśnie pojawił się na planszy na później.

**RotateBlockCW()-** pozwala na obrócenie klocka o 90 stopni zgodnie z ruchem wskazówek zegara. Jeśli po wykonaniu obrotu klocek będzie w pozycji niedozwolonej, to powróci on do stanu przed obróceniem.

**RotateBlockCCW()-** pozwala na obrócenie klocka o 90 stopni przeciwnie do ruchu wskazówek zegara. Jeśli po wykonaniu obrotu klocek będzie w pozycji niedozwolonej, to powróci on do stanu przed obróceniem.

**MoveBlockLeft()-** pozwala na przesunięcie bloku w lewo w wierszu.

**MoveBlockRight()-** pozwala na przesunięcie bloku w prawo w wierszu.

**IsGameOver()-** sprawdza, czy górne wiersze są puste. Jeśli są one puste, gra się nie zakończy.

**PlaceBlock()-** sprawdza czy aktualny klocek nie może być już przesunięty w dół. Następnie sprawdza poprzez metodę IsGameOver() czy gra się zakończyła, czy też nie. Pokazuje również wynik, który jest ilością zapełnionych wierszy.

**MoveBlockDown()-** pozwala na przesunięcie bloku w dół jeżeli nic nie stoi na przeszkodzie.

**TileDropDistance()-** bierze pozycje i zmienia ilość pustych wierszy poniżej tego klocka na najniższy możliwy.

**BlockDropDistance()-** bierze pozycje i z metody TileDropDistance() bierze minimum i o tyle zrzuca klocek.

**DropBlock()-** pozwala na zrzucenie kocka na sam dół planszy i wstawia go do grida.

**6. Klasa BlockQueue**

Klasa BLockQueue jest odpowiedzialna za zarządzanie kolejką klocków.

**Metody jakie się w tej klasie znajdują:**

**Blocks –** definiuje nową tablice.

**Blockqueue()-**konstruktor, który pozwala na losowanie za każdym razem innego klocka.

**RandomBlock()-**zwraca następny klocek.

**GetAndUpdate()-** zwraca następny blok i aktualizuje właściwość. Będzie wybierał klocki tak, aby się nie powtarzały jeden za drugim.

**7. Klasa Position**

Klasa Position przechowuje wiersze i kolumny w grze(Potrzebna do określenia pozycji klocków i ich poruszania się).

Posiada jedynie konstruktor Position zawierający wiersze i kolumny.

**8. Klasa Block**

Klasa Block przechowuje pozycje, gdzie klocki się pojawią, zidentyfikuje rodzaj klocka po id jaki został nałożony na niego oraz odpowiada za ruchy klockiem.

**Metody jakie się w tej klasie znajdują:**

**TilePositions()-** zwraca nam pozycje grida okupowanego przez klocek, uwzględniając rotację i offseta.

**RotateCW()-** pozwala na obrócenie klocka o 90 stopni zgodnie z ruchem wskazówek zegara.

**RotateCCW()-** pozwala na obrócenie klocka o 90 stopni przeciwnie do ruchu wskazówek zegara.

**Move()-** pozwala na przesunięcie klocka o dany numer wierszy i kolumn.

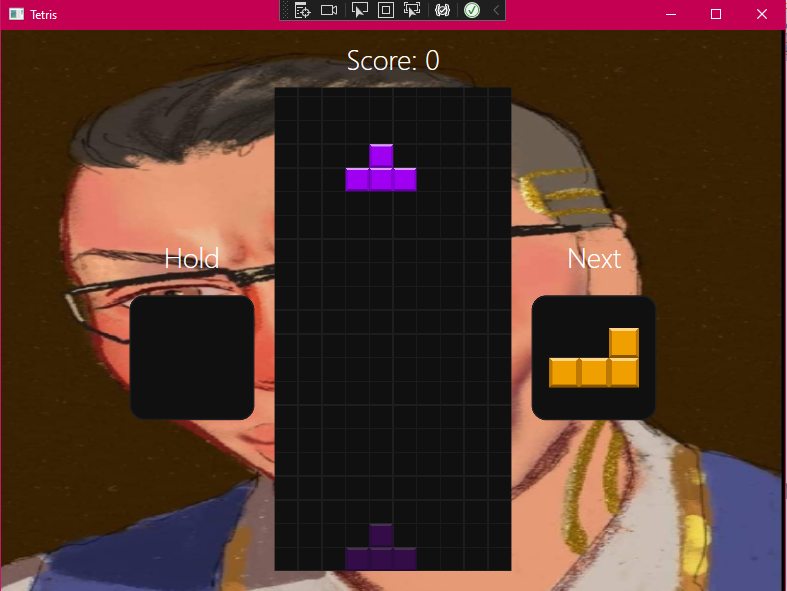
**Reset()-** Jak sama nazwa wskazuje pozwala na zresetowanie rotacji i pozycji klocka.

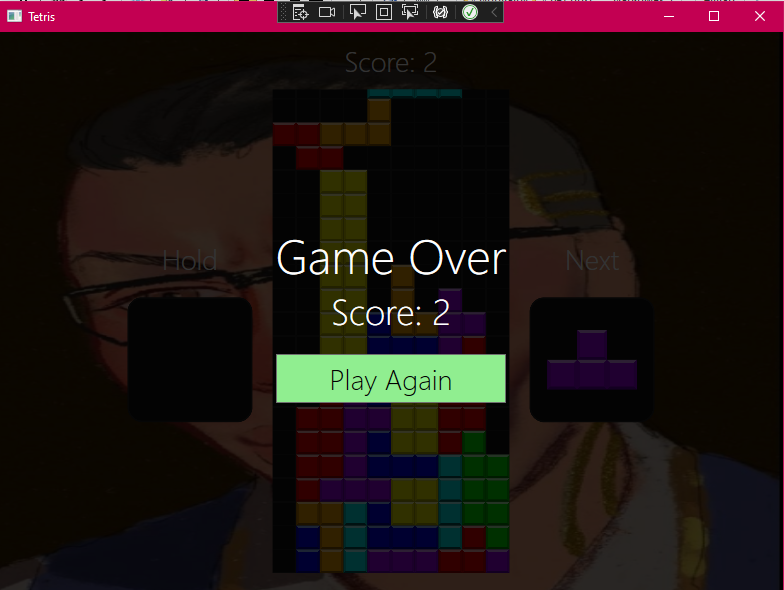
**9.Klasy IBlock,JBlock itd.**

Klasy IBlock, JBlock, LBlock, OBlock, SBlock, TBlock, ZBlock zawierają w sobie   
4 pozycje na różne stopnie rotacji klocka. Posiadają przypisane do nich id oraz pozycje   
w której zaczną spadać na planszy.

Wyjątkową klasą jest OBlock, której klocek jest kwadratem i jego rotacji nie widać. Dlatego przechowuje ona tylko 1 pozycje do rotacji.

**10.Prezentacja okienka Gry**

Przedstawiam kilka zdjęć z ekranu gry oraz jak ona na obecną chwilę wygląda.  


****