PROGRAMOWANIE W JĘZYKU C++ - PROJEKT

Temat: Pong, Arkanoid, Simon Says - gry

Student: Maksymilian Dendura

Grupa: P01 EF2-DI

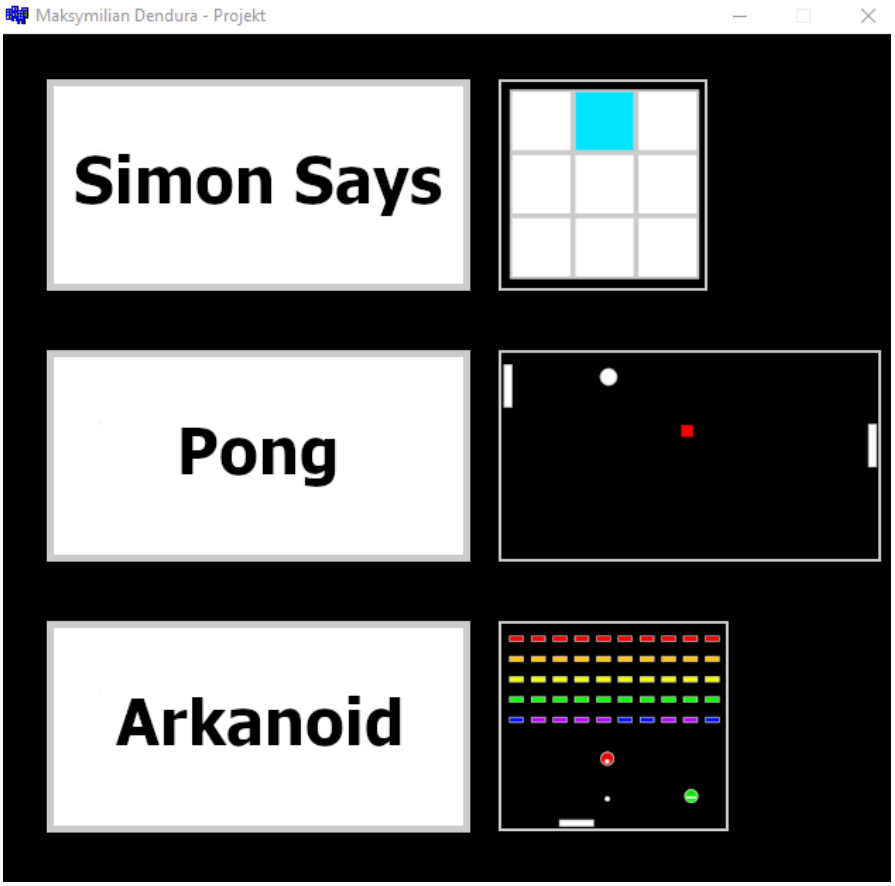
Spis treści

Instrukcja obsługi

Program uruchamiamy poprzez plik PROJEKT.exe – uwaga, nie należy go wyciągać z folderu projektu!

img	11.01.2021 18:25	Folder plików	
scr	11.01.2021 18:25	Folder plików	
snd	11.01.2021 18:25	Folder plików	
Project1.obj	27.11.2020 11:45	3D Object	22 KB
Score.obj	09.12.2020 17:32	3D Object	261 KB
Unit1.obj	27.11.2020 11:45	3D Object	61 KB
Unit2.obj	09.12.2020 18:13	3D Object	149 KB
Unit3.obj	09.12.2020 18:13	3D Object	127 KB
Unit4.obj	09.12.2020 17:45	3D Object	79 KB
PROJEKT.exe	09.12.2020 18:13	Aplikacja	3 836 KB
Project1.bpr	09.12.2020 18:21	BCBProject	4 KB
Score.h	08.12.2020 21:07	C/C++ Header	1 KB
Unit1.h	27.11.2020 11:28	C/C++ Header	5 KB
Unit2.h	08.12.2020 23:02	C/C++ Header	4 KB
Unit3.h	09.12.2020 17:44	C/C++ Header	2 KB
Unit4.h	27.11.2020 11:25	C/C++ Header	2 KB
Project1.cpp	27.11.2020 11:25	C++ Source	2 KB
Score.cpp	08.12.2020 23:27	C++ Source	4 KB
Unit1.cpp	27.11.2020 11:25	C++ Source	14 KB
Unit2.cpp	09.12.2020 18:11	C++ Source	20 KB
Unit3.cpp	09.12.2020 18:11	C++ Source	8 KB
Unit4.cpp	27.11.2020 11:25	C++ Source	2 KB
Unit1.dfm	09.12.2020 18:13	C++Builder Form	2 631 KB
Unit2.dfm	09.12.2020 18:12	C++Builder Form	2 251 KB
Unit3.dfm	09.12.2020 18:12	C++Builder Form	334 KB
Unit4.dfm	09.12.2020 18:12	C++Builder Form	1 766 KB
Project1.res	02.11.2020 13:56	Compiled Resourc...	1 KB
Project1.tds	09.12.2020 18:21	Plik TDS	3 776 KB

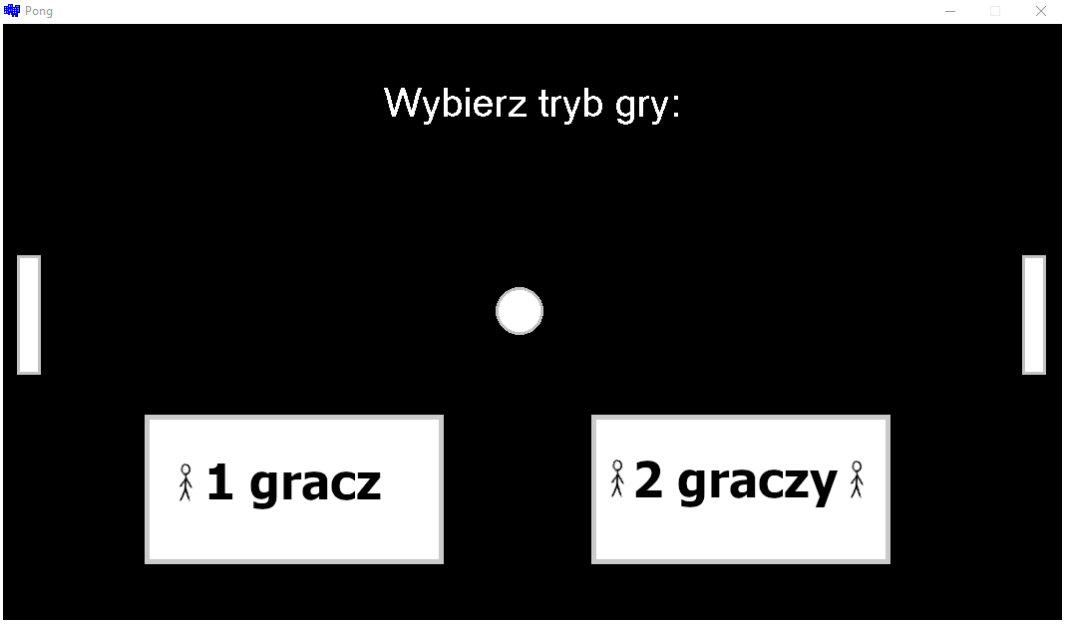
Po uruchomieniu widzimy okno wyboru rozgrywki. Należy kliknąć w tryb, który nas interesuje. Otworzy się wtedy nowe okno z grą.



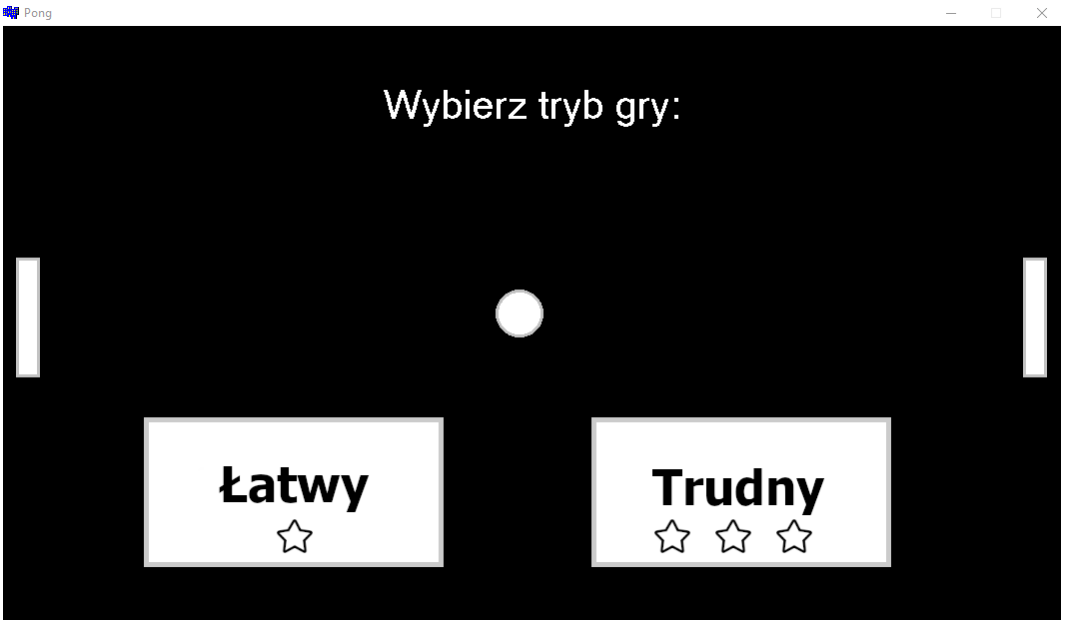
Pong

Gra polega na odbijaniu piłki paletkami – kto pierwszy jej nie odbije, ten przegrywa.

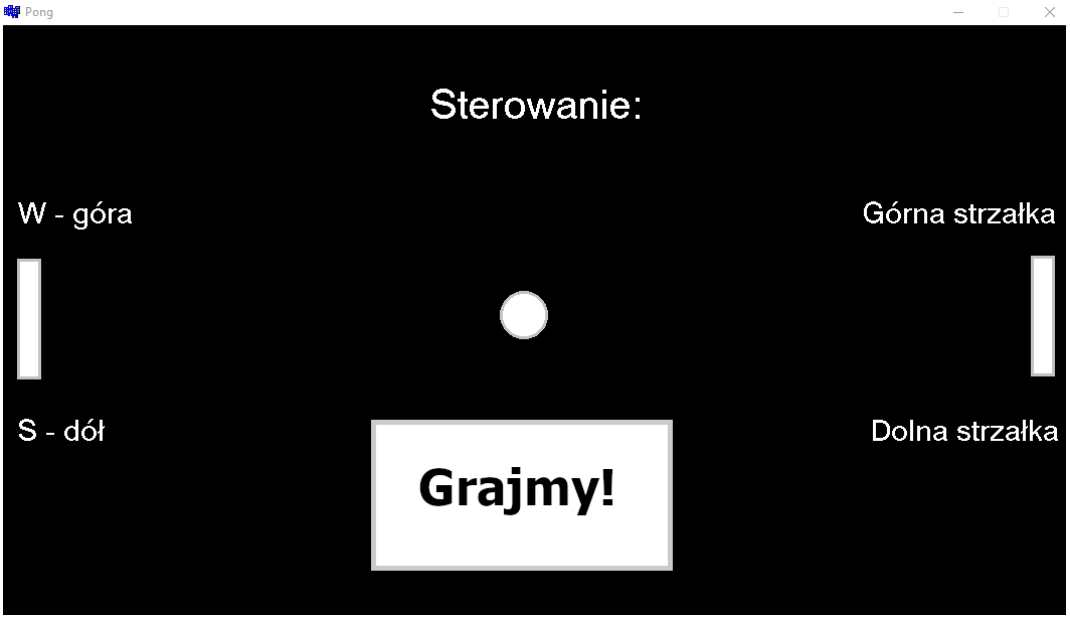
Na początku należy sprecyzować, jaki tryb gry nas interesuje – 1 gracz (przeciwko komputerowi) lub 2 graczy.



Następnie należy wybrać poziom trudności. Poziom trudny sprawia, że w trakcie rozgrywki pojawia się czerwony kwadrat utrudniający rozgrywkę poprzez odbicie piłki. Jeśli wybrano tryb gry z komputerem, będzie on również szybciej reagował na zmianę pozycji piłki.



Sterowanie odbywa się poprzez klawisze W oraz S dla gracza nr 1 oraz strzałkami w górę i dół dla gracza nr 2.



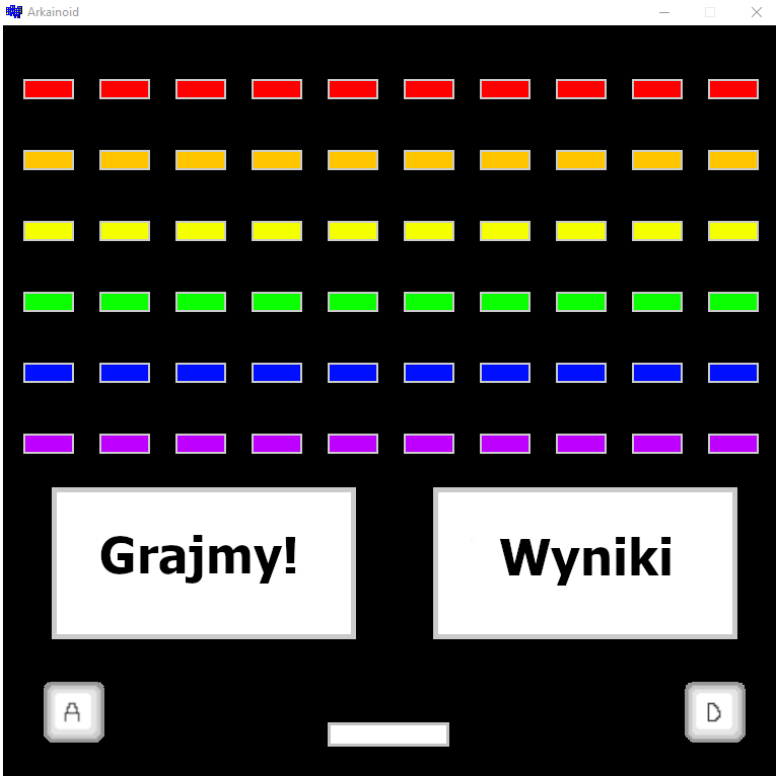
Wciśnięcie przycisku „Grajmy!” uruchomi rozgrywkę.

Po zakończeniu rozgrywki możemy zagrać jeszcze raz poprzez przycisk „Jeszcze raz!” lub wrócić do menu głównego przyciskiem „Menu” aby zmienić tryb rozgrywki.

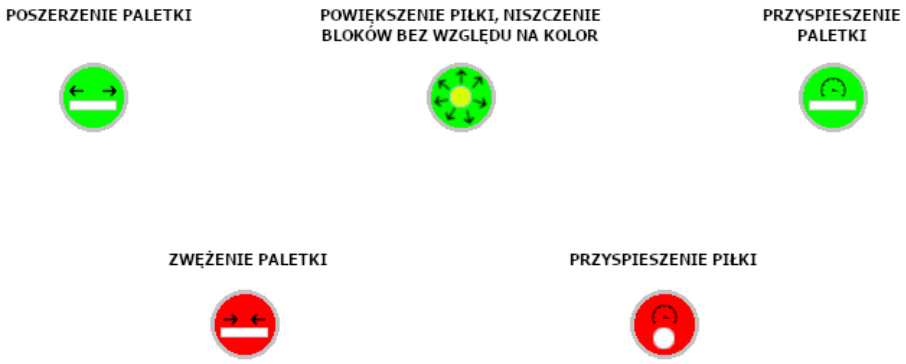
Arkanoid

Gra polega na odpijaniu piłeczki i zbiciu jak największej ilości bloków. Przegrana jest powodowana przez nie odbicie piłeczki.

Aby rozpocząć rozgrywkę, należy wcisnąć przycisk „Grajmy!”. Możemy także sprawdzić nasze wyniki poprzez kliknięcie w przycisk „Wyniki”. Sterowanie odbywa się za pomocą klawiszy A oraz D.



Podczas gry możemy spotkać wiele bonusów. „Dobre” bonusy zostały oznaczone na zielono, „złe” bonusy – na czerwono.



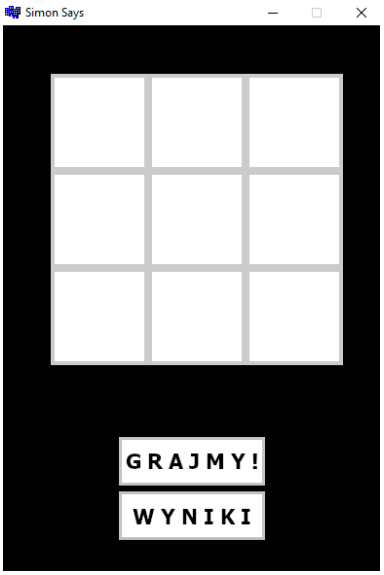
Jeśli przegramy i chcemy zapisać swój wynik, należy napisać swój nick w odpowiednim polu oraz kliknąć „OK”.

Jeśli jednak uda nam się zbić wszystkie bloki, będziemy mogli wejść na kolejny, inny poziom.

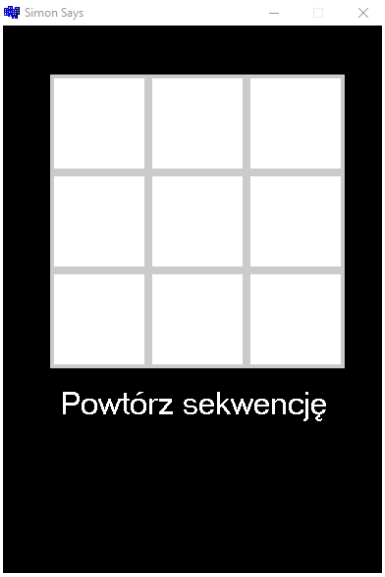
Simon Says

Gra polega na powtarzaniu losowo wygenerowanych sekwencji. Pierwsza pomyłka oznacza przegraną.

Aby rozpocząć rozgrywkę, należy wcisnąć przycisk „Grajmy!”. Możemy także sprawdzić nasze wyniki poprzez kliknięcie w przycisk „Wyniki”. Sterowanie odbywa się za pomocą myszki.



Po rozpoczęciu rozgrywki należy poczekać, aż komputer skończy wykonywać sekwencję i pojawi się napis „Powtórz sekwencję”



Następnie należy powtórzyć sekwencję. Jeżeli będzie ona poprawna, pojawi się niebieski bloczek. Jeśli był to ostatni bloczek w sekwencji, otrzymamy stosowną informację, po której należy odczekać na kolejną sekwencję.

Jeśli przegramy i nasz wynik będzie większy od 0, zostaniemy poproszeni o nick aby zapisać wynik.

Ponowna rozgrywka jest możliwa przez kliknięcie przycisku „Restart”.

Konstrukcja projektu

W projekcie znajdują się **3 foldery:**

- **img** – zawiera pliki graficzne wykorzystywane przez program
- **scr** – zawiera pliki tekstowe w których zapisywane są wyniki graczy
- **snd** – zawiera pliki dźwiękowe wykorzystywane przez program

5 plików nagłówkowych:

- **Score.h** (klasa Score obsługująca wczytywanie oraz zapisywanie wyników do pliku)
- **Unit1**.**h** (klasa TForm1 odpowiadająca za pierwsze okno – grę Pong)
- **Unit2**.**h** (klasa TForm2 odpowiadająca za drugie okno – grę Arkanoid)
- **Unit3.h** (klasa TForm3 odpowiadająca za trzecie okno – grę Simon Says)
- **Unit4.h** (klasa TForm4 odpowiadająca za okno główne wyboru rozgrywki)

6 plików źródłowych:

- **Project1.cpp** – plik wygenerowany automatycznie przez środowisko C++ builder, odpowiadający za uruchomienie całego projektu
- **Score**.**cpp**
- **Unit1.cpp**
- **Unit2.cpp**
- **Unit3.cpp**
- **Unit4.cpp**

1 plik wykonywalny PROJEKT.exe

Pozostałe pliki (.bpr, .obj, .dfm, .res, .tds) zostały stworzone automatycznie przez środowisko.

Klasy

TForm1

Klasa odpowiada za okno z grą Pong. Plik nagłówkowy tej klasy to Unit1.h, plik źródłowy to Unit1.cpp.

Pola składowe stworzone za pomocą narzędzi środowiska:

- **TImage* Ball –** wskaźnik na obiekt wyświetlający piłkę
- **TImage* PaddleLeft -** wskaźnik na obiekt wyświetlający lewą paletkę
- **TImage* PaddleRight -** wskaźnik na obiekt wyświetlający prawą paletkę
- **TShape* Background -** wskaźnik na obiekt wyświetlający tło
- **TLabel* StringMain -** wskaźnik na obiekt wyświetlający główny napis na górze
- **TImage* ButtonOnePlayer -** wskaźnik na obiekt wyświetlający przycisk trybu dla 1 gracza
- **TImage* ButtonTwoPlayers -** wskaźnik na obiekt wyświetlający przycisk trybu dla 2 graczy
- **TLabel* StringW -** wskaźnik na obiekt wyświetlający „W”
- **TLabel* StringS -** wskaźnik na obiekt wyświetlający „S”
- **TLabel* StringUpper -** wskaźnik na obiekt wyświetlający „Strzałka w górę”
- **TLabel* StringDown -** wskaźnik na obiekt wyświetlający „Strzałka w dół”
- **TImage* ButtonPlay -** wskaźnik na obiekt wyświetlający przycisk „Grajmy!”
- **TTimer* TimerBall -** wskaźnik na obiekt odliczający czas podczas którego wykonuje główną mechanikę gry, w tym ruch piłki
- **TTimer* TimerLeftPaddleUp -** wskaźnik na obiekt odliczający czas podczas którego wykonuje ruch lewej paletki w górę
- **TTimer* TimerLeftPaddleDown -** wskaźnik na obiekt odliczający czas podczas którego wykonuje ruch lewej paletki w dół
- **TTimer* TimerRightPaddleUp -** wskaźnik na obiekt odliczający czas podczas którego wykonuje ruch prawej paletki w górę
- **TTimer* TimerRightPaddleDown -** wskaźnik na obiekt odliczający czas podczas którego wykonuje ruch prawej paletki w dół
- **TImage* ButtonRestart -** wskaźnik na obiekt wyświetlający przycisk „Jeszcze raz!”
- **TImage* ButtonEasy -** wskaźnik na obiekt wyświetlający przycisk „Łatwy”
- **TImage* ButtonHard -** wskaźnik na obiekt wyświetlający przycisk „Trudny”
- **TShape* ReversePoint -** wskaźnik na obiekt wyświetlający czerwony kwadrat utrudniający rozgrywkę
- **TImage* ButtonMenu -** wskaźnik na obiekt wyświetlający przycisk „Menu”

Publiczne pola składowe dodane ręcznie:

- **bool game** – zmienna jest ustawiana na true podczas rozgrywki oraz false gdy rozgrywka się skończy
- **int gameMode** – zmienna przechowuje informację o trybie gry, 1 – jeden gracz, 2 – dwóch graczy
- **int BallSpeedx** – zmienna przechowująca prędkość piłki w kierunku poziomym
- **int BallSpeedy** - zmienna przechowująca prędkość piłki w kierunku pionowym
- **int point** - zmienna przechowuje informację o tym, kto ostatnio odbił piłkę - 1 czy 2 gracz
- **int hardiness** - zmienna przechowuje poziom trudności – 1 - łatwy, 2 - trudny
- **int randomNumber** - zmienna pomocnicza do zmiany szybkości piłki w kierunku pionowym

Metody stworzone za pomocą narzędzi środowiska:

- **void __fastcall ButtonOnePlayerClick(TObject* Sender)** - funkcja obsługująca zdarzenie wciśnięcia przycisku gry 1-osobowej
- **void __fastcall ButtonTwoPlayersClick(TObject* Sender)** - funkcja obsługująca zdarzenie wciśnięcia przycisku gry 2-osobowej
- **void __fastcall ButtonPlayClick(TObject* Sender)** - funkcja obsługująca zdarzenie wciśnięcia przycisku „Grajmy!”
- **void __fastcall TimerBallTimer(TObject* Sender)** - funkcja główna gry
- **void __fastcall TimerLeftPaddleUpTimer(TObject* Sender)** - funkcja wykonująca się przy naciśnięciu klawisza W
- **void __fastcall TimerLeftPaddleDownTimer(TObject* Sender)** - funkcja wykonująca się przy naciśnięciu klawisza S
- **void __fastcall TimerRightPaddleUpTimer(TObject* Sender)** - funkcja wykonująca się przy naciśnięciu strzałki do góry
- **void __fastcall TimerRightPaddleDownTimer(TObject* Sender)** - funkcja wykonująca się przy naciśnięciu strzałki w dół
- **void __fastcall FormKeyDown(TObject* Sender, WORD& Key, TShiftState Shift)** - funkcja obsługująca wciśnięcie przycisku
- **void __fastcall FormKeyUp(TObject* Sender, WORD& Key, TShiftState Shift)** - funkcja obsługująca opuszczenie przycisku
- **void __fastcall ButtonRestartClick(TObject* Sender)** - funkcja obsługująca zdarzenie wciśnięcia przycisku restartującego grę
- **void __fastcall ButtonEasyClick(TObject* Sender)** - funkcja obsługująca zdarzenie wciśnięcia przycisku trybu łatwego
- **void __fastcall ButtonHardClick(TObject* Sender)** - funkcja obsługująca zdarzenie wciśnięcia przycisku trybu trudnego
- **void __fastcall ButtonMenuClick(TObject* Sender) -** funkcja obsługująca zdarzenie wciśnięcia przycisku menu
- **void __fastcall FormClose(TObject* Sender, TCloseAction& Action)** – funkcja obsługująca zdarzenie zamknięcia okna
- **__fastcall TForm1(TComponent* Owner)** - konstruktor

Publiczne metody dodane ręcznie:

- **void endGame()** – funkcja zatrzymująca rozgrywkę
- **void computerMove()** – funkcja obsługująca ruch komputera
- **void ballMove()** – funkcja obsługująca ruch piłki
- **void reversePoint()** – funkcja odpowiadająca za czerwony punkt utrudniający rozgrywkę

TForm2

Klasa odpowiada za okno z grą Arkanoid. Plik nagłówkowy tej klasy to Unit2.h, plik źródłowy to Unit2.cpp.

Pola składowe stworzone za pomocą narzędzi środowiska:

- **TShape* Background -** wskaźnik na obiekt wyświetlający tło
- **TImage* Paddle** - wskaźnik na obiekt wyświetlający paletkę
- **TImage* Ball** - wskaźnik na obiekt wyświetlający piłkę
- **TTimer* TimerBall** - wskaźnik na obiekt odliczający czas podczas którego wykonuje ruch piłki
- **TTimer* TimerRight** - wskaźnik na obiekt odliczający czas podczas którego wykonuje ruch paletki w prawo
- **TTimer* TimerLeft** - wskaźnik na obiekt odliczający czas podczas którego wykonuje ruch paletki w lewo
- **TImage* ButtonPlay** - wskaźnik na obiekt wyświetlający przycisk „Grajmy!”
- **TImage* ButtonRestart** - wskaźnik na obiekt wyświetlający przycisk „Jeszcze raz!”
- **TLabel* StringMain** - wskaźnik na obiekt wyświetlający główny napis (np. ten z punktami po przegranej)
- **TImage* BallSpeedPoint** - wskaźnik na obiekt wyświetlający bonus przyspieszający piłkę
- **TTimer* TimerBallSpeedPointEnd** - wskaźnik na obiekt odliczający czas podczas którego trwa bonus szybszej piłki
- **TImage* PaddleSpeedPoint** - wskaźnik na obiekt wyświetlający bonus przyspieszający paletkę
- **TTimer* TimerPaddleSpeedPointEnd** - wskaźnik na obiekt odliczający czas podczas którego trwa bonus szybszej paletki
- **TTimer* TimerHyperBallPointEnd** - wskaźnik na obiekt odliczający czas podczas którego trwa bonus większej piłki
- **TTimer* TimerPaddleWidthDownPointEnd** - wskaźnik na obiekt odliczający czas podczas którego trwa bonus węższej paletki
- **TTimer* TimerPaddleWidthUpPointEnd** - wskaźnik na obiekt odliczający czas podczas którego trwa bonus szerszej paletki
- **TImage* HyperBallPoint** - wskaźnik na obiekt wyświetlający bonus powiększający piłkę
- **TImage* PaddleWidthDownPoint** - wskaźnik na obiekt wyświetlający bonus zwężający paletkę
- **TImage* PaddleWidthUpPoint** - wskaźnik na obiekt wyświetlający bonus poszerzający paletkę
- **TImage* aKey** - wskaźnik na obiekt wyświetlający klawisz sterowania A
- **TImage* dKey** - wskaźnik na obiekt wyświetlający klawisz sterowania D
- **TImage* ButtonScores** - wskaźnik na obiekt wyświetlający przycisk „Wyniki”
- **TLabel* StringScores** - wskaźnik na obiekt wyświetlający tablicę wyników
- **TImage* ButtonOK** - wskaźnik na obiekt wyświetlający przycisk „OK” podczas wyświetlania wyników
- **TImage* ButtonNextLevel** - wskaźnik na obiekt wyświetlający przycisk następnego poziomu
- **TLabel *StringNick** - wskaźnik na obiekt wyświetlający informacje o wpisaniu nicku
- **TEdit *EditNick** - wskaźnik na obiekt wyświetlający oraz pobierający wpisywany nick
- **TImage *ButtonNickOK** - wskaźnik na obiekt wyświetlający przycisk „OK” podczas wpisywania nicku

Publiczne pola składowe dodane ręcznie:

- **int ballSpeedx** – zmienna przechowująca prędkość piłki w kierunku poziomym
- **int ballSpeedy** – zmienna przechowująca prędkość piłki w kierunku pionowym
- **int hardiness** – zmienna przechowująca licznik trudności zwiększający się wraz z długością rozgrywki zwiększając prędkość piłki w kierunku pionowym
- **int hardinessx** - zmienna przechowująca licznik trudności zwiększający się wraz z długością rozgrywki zwiększając prędkość piłki w kierunku poziomym
- **int paddleCollisionTicks** – zmienna przechowująca ilość odbić od paletki w danym czasie
- **int maxPaddleCollisionTicks** - zmienna przechowująca maksymalną ilość odbić od paletki w danym czasie (podczas bonusu powiększającego piłkę)
- **int points** - zmienna przechowująca liczbę punktów
- **int prevPoints** - zmienna przechowująca liczbę punktów w poprzedniej rundzie
- **int blockHitCount[60]** – zmienna przechowująca ile uderzeń w dany blok pozostało
- **int paddleSpeed** – zmienna przechowująca prędkość paletki
- **int collisionsIterator** – zmienna przechowująca jak dużo uderzeń od bloków było w danym czasie
- **bool blocksTrue[60]** – zmienna przechowująca czy dany blok istnieje
- **bool game** – zmienna przyjmująca wartość true podczas rozgrywki oraz false jeśli się skończyła (lub nie zaczęła)
- **bool isHyperBall** – zmienna przyjmująca wartość true jeśli bonus powiększający piłkę został zebrany, false w przeciwnym wypadku
- **bool isPaddleSpeedPoint** - zmienna przyjmująca wartość true jeśli bonus przyspieszający paletkę znajduje się na mapie lub został zebrany, false w przeciwnym wypadku
- **bool isBallSpeedPoint** - zmienna przyjmująca wartość true jeśli bonus przyspieszający piłkę znajduje się na mapie lub został zebrany, false w przeciwnym wypadku
- **bool isPaddleWidthUpPoint** - zmienna przyjmująca wartość true jeśli bonus poszerzający paletkę znajduje się na mapie lub został zebrany, false w przeciwnym wypadku
- **bool isPaddleWidthDownPoint** - zmienna przyjmująca wartość true jeśli bonus zwężający paletkę znajduje się na mapie lub został zebrany, false w przeciwnym wypadku
- **bool isHyperBallPoint** - zmienna przyjmująca wartość true jeśli bonus przyspieszający paletkę znajduje się na mapie lub został zebrany, false w przeciwnym wypadku
- **bool isPaddleWidthUpPointPicked** - zmienna przyjmująca wartość true jeśli bonus poszerzający paletkę znajduje się na mapie został zebrany i trwa, false w przeciwnym wypadku
- **bool wasHyperBallPicked** - zmienna przyjmująca wartość true jeśli bonus powiększający piłkę został już zebrany w tej rundzie, false w przeciwnym wypadku
- **std::string nick** – zmienna przechowująca nick gracza
- **TImage* Block[60]** – tablica wskaźników na obiekty wyświetlające bloki
- **TImage* lastBlock** – wskaźnik na ten blok, który został ostatnio uderzony
- **Score* score** – wskaźnik na obiekt odpowiadający za zapis oraz odczyt wyników

Metody stworzone za pomocą narzędzi środowiska:

- **void __fastcall TimerBallTimer(TObject* Sender)** – funkcja główna gry
- **void __fastcall TimerRightTimer(TObject* Sender)** – funkcja odpowiadająca za ruch paletki w prawo
- **void __fastcall TimerLeftTimer(TObject* Sender)** – funkcja odpowiadająca za ruch paletki w lewo
- **void __fastcall FormKeyDown(TObject* Sender, WORD& Key, TShiftState Shift)** – funkcja odpowiadająca za obsługę wciśnięcia przycisku
- **void __fastcall FormKeyUp(TObject* Sender, WORD& Key, TShiftState Shift)** - funkcja odpowiadająca za obsługę puszczenia przycisku
- **void __fastcall ButtonPlayClick(TObject* Sender)** – funkcja obsługująca kliknięcie w przycisk „Grajmy!”
- **void __fastcall ButtonRestartClick(TObject* Sender)** – funkcja obsługująca kliknięcie w przycisk „Jeszcze raz!”
- **void __fastcall FormClose(TObject* Sender, TCloseAction& Action)** – funkcja obsługująca zamknięcie okna
- **void __fastcall TimerBallSpeedPointEndTimer(TObject* Sender)** – funkcja kończąca bonus przyspieszający piłkę
- **void __fastcall TimerPaddleSpeedPointEndTimer(TObject* Sender)** - funkcja kończąca bonus przyspieszający paletkę
- **void __fastcall TimerPaddleWidthDownPointEndTimer(TObject* Sender)** - funkcja kończąca bonus zwężający paletkę
- **void __fastcall TimerPaddleWidthUpPointEndTimer(TObject* Sender)** - funkcja kończąca bonus poszerzający paletkę
- **void __fastcall TimerHyperBallPointEndTimer(TObject* Sender)** - funkcja kończąca bonus powiększający piłkę
- **void __fastcall ButtonScoresClick(TObject* Sender)** - funkcja obsługująca kliknięcie w przycisk „Wyniki”
- **void __fastcall ButtonOKClick(TObject* Sender)** - funkcja obsługująca kliknięcie w przycisk „OK” podczas wyświetlania wyników
- **void __fastcall ButtonNextLevelClick(TObject* Sender)** - funkcja obsługująca kliknięcie w przycisk następnego poziomu
- **void __fastcall ButtonNickOKClick(TObject* Sender)** - funkcja obsługująca kliknięcie w przycisk „OK” podczas wpisywania nicku
- **__fastcall TForm2(TComponent* Owner)** - konstruktor
- **__fastcall ~TForm2()** – destruktor

Publiczne metody dodane ręcznie:

- **bool collision(TImage* Object1, TImage* Object2)** – funkcja zwraca wartość true jeśli dwa obiekty ze sobą kolidują, false w przeciwnym wypadku
- **void colorAllBlocks()** – funkcja koloruje wszystkie bloki
- **void ballMove()** – funkcja odpowiada za ruch piłki
- **void blockCollisionControl()** – funkcja odpowiada za kontrolę kolizji bloków z piłką
- **void endGame()** – funkcja kończąca grę
- **void hardinessControl()** – funkcja kontrolująca trudność rozgrywki
- **void bonusCreateControl(TImage* Block)** – funkcja odpowiadająca za możliwość stworzenia bonusu, jako argument przyjmuje ostatni uderzony blok
- **void ballSpeedPointCreate(TImage* Block)** – funkcja odpowiadająca za stworzenie bonusu przyspieszającego piłkę, jako argument przyjmuje ostatni uderzony blok
- **void ballSpeedPointCollision()** – funkcja kontrolująca kolizję z utworzonym bonusem przyspieszającym piłkę
- **void hyperBallPointCreate(TImage* Block)** – funkcja odpowiadająca za stworzenie bonusu powiększającego piłkę, jako argument przyjmuje ostatni uderzony blok
- **void hyperBallPointCollision()** - funkcja kontrolująca kolizję z utworzonym bonusem powiększającym piłkę
- **void paddleSpeedPointCreate(TImage* Block)** – funkcja odpowiadająca za stworzenie bonusu przyspieszającego paletkę, jako argument przyjmuje ostatni uderzony blok
- **void paddleSpeedPointCollision()** - funkcja kontrolująca kolizję z utworzonym bonusem przyspieszającym paletkę
- **void paddleWidthDownPointCreate(TImage* Block)** – funkcja odpowiadająca za stworzenie bonusu zwężającego paletkę, jako argument przyjmuje ostatni uderzony blok
- **void paddleWidthDownPointCollision()** - funkcja kontrolująca kolizję z utworzonym bonusem zwężającym paletkę
- **void paddleWidthUpPointCreate(TImage* Block)** – funkcja odpowiadająca za stworzenie bonusu poszerzającego paletkę, jako argument przyjmuje ostatni uderzony blok
- **void paddleWidthUpPointCollision()** - funkcja kontrolująca kolizję z utworzonym bonusem poszerzającym paletkę
- **void setDefaultValues()** – funkcja ustawia wartości domyślne pól składowych

TForm3

Klasa odpowiada za okno z grą Simon Says. Plik nagłówkowy tej klasy to Unit3.h, plik źródłowy to Unit3.cpp.

Pola składowe stworzone za pomocą narzędzi środowiska:

- **TShape* Background** - wskaźnik na obiekt wyświetlający tło
- **TImage* ButtonPlay** - wskaźnik na obiekt wyświetlający przycisk „Grajmy!”
- **TTimer* TimerBeforeSequence** - wskaźnik na obiekt odliczający czas podczas którego wyświetlana jest informacja „dobrze” oraz kolorowane są bloki
- **TTimer* TimerSequence** - wskaźnik na obiekt odliczający czas podczas którego odgrywana jest sekwencja
- **TLabel* StringMain** - wskaźnik na obiekt wyświetlający główny napis
- **TImage* ButtonRestart** - wskaźnik na obiekt wyświetlający przycisk „Restart”
- **TImage* ButtonScores** - wskaźnik na obiekt wyświetlający przycisk „Wyniki”
- **TLabel* StringScores** - wskaźnik na obiekt wyświetlający wyniki
- **TImage* ButtonOK** - wskaźnik na obiekt wyświetlający przycisk „OK” w wynikach
- **TLabel *StringNick** - wskaźnik na obiekt wyświetlający informację o wpisaniu nicku
- **TEdit *EditNick** - wskaźnik na obiekt wyświetlający wpisywany nick oraz wczytujący go
- **TImage *ButtonNickOK** - wskaźnik na obiekt wyświetlający przycisk „OK” podczas wpisywaniu nicku

Publiczne pola składowe dodane ręcznie:

- **int order[100]** – tablica zawierająca informację o kolejności kliknięcia w bloki: indeks to nr kliknięcia, a wartość to nr bloku
- **int orderIterator** – licznik kolejności używany podczas gry jako indeks tablicy order
- **int points** – liczba punktów (w praktyce: liczba punktów – 1)
- **std::string nick** – zmienna przechowująca nick gracza
- **TImage* tmp** – wskaźnik na ostatni blok
- **TImage* block[9]** – tablica wskaźników na obiekty wyświetlające bloki
- **Score* score** – wskaźnik na obiekt odpowiadający za zapisywanie i wczytywanie wyników

Metody stworzone za pomocą narzędzi środowiska:

- **void __fastcall ButtonPlayClick(TObject* Sender)** – funkcja obsługująca kliknięcie w przycisk „Grajmy!”
- **void __fastcall TimerSequenceTimer(TObject* Sender)** – funkcja wyświetlająca sekwencję
- **void __fastcall TimerBeforeSequenceTimer(TObject* Sender)** – funkcja wyświetlająca informacje przed kolejną sekwencją
- **void __fastcall ButtonRestartClick(TObject* Sender)** – funkcja obsługująca kliknięcie w przycisk „Restart”
- **void __fastcall FormClose(TObject* Sender, TCloseAction& Action)** – funkcja obsługująca zamknięcie okna
- **void __fastcall ButtonScoresClick(TObject* Sender)** - funkcja obsługująca kliknięcie w przycisk „Wyniki”
- **void __fastcall ButtonOKClick(TObject* Sender)** - funkcja obsługująca kliknięcie w przycisk „OK” podczas wyświetlania wyników
- **void __fastcall ButtonNickOKClick(TObject *Sender)** - funkcja obsługująca kliknięcie w przycisk „OK” podczas wpisywania nicku
- **__fastcall TForm3(TComponent* Owner)** - konstuktor
- **__fastcall ~TForm3()** - desktruktor

Publiczne metody dodane ręcznie:

- **void __fastcall BlockClick(TObject* Sender)** – funkcja obsługująca kliknięcie w blok
- **void activateBlock(int blockNumber)** – funkcja aktywująca dany blok
- **void endGame()** – funkcja kończąca grę

TForm4

Klasa odpowiada za okno główne z wyborem gry. Plik nagłówkowy tej klasy to Unit4.h, plik źródłowy to Unit4.cpp.

Pola składowe stworzone za pomocą narzędzi środowiska:

- **TShape* Background** - wskaźnik na obiekt wyświetlający tło
- **TImage* ButtonArkanoid** - wskaźnik na obiekt wyświetlający przycisk „Arkanoid”
- **TImage* ButtonPong** - wskaźnik na obiekt wyświetlający przycisk „Pong”
- **TImage* ButtonSimonSays** - wskaźnik na obiekt wyświetlający przycisk „Simon Says”
- **TImage* ImageSimonSays** - wskaźnik na obiekt wyświetlający obrazek gry Simon Says
- **TImage* ImagePong** - wskaźnik na obiekt wyświetlający obrazek gry Pong
- **TImage* ImageArkanoid** - wskaźnik na obiekt wyświetlający obrazek gry Arkanoid

Metody stworzone za pomocą narzędzi środowiska:

- **void __fastcall ButtonSimonSaysClick(TObject* Sender)** – funkcja odpowiadająca za obsługę kliknięcia w przycisk „Simon Says”
- **void __fastcall ButtonPongClick(TObject* Sender)** - funkcja odpowiadająca za obsługę kliknięcia w przycisk „Pong”
- **void __fastcall ButtonArkanoidClick(TObject* Sender)** - funkcja odpowiadająca za obsługę kliknięcia w przycisk „Arkanoid”
- **__fastcall TForm4(TComponent* Owner)** – konstruktor

Score

Klasa odpowiada za zapisywanie oraz odczytywanie wyników z pliku. Plik nagłówkowy tej klasy to Score.h, plik źródłowy to Score.cpp.

Prywatne pola składowe dodane ręcznie:

- **std::string currentGame** – zmienna przechowuje nazwę aktualnie rozgrywanej gry

Publiczne metody dodane ręcznie:

- **void setCurrentGame(std::string game)** – ustawia pola currentGame na podaną w argumencie
- **void save(int points, std::string nick)** – zapisuje wynik wraz z nickiem do odpowiedniego pliku
- **void normalise()** – normalizuje zapisane wyniki, zalecane przed wpisaniem lub odczytaniem wyniku
- **void sortScores(scoreStruct array[], int size)** – sortuje wyniki w pliku od największego do najmniejszego, funkcja normalise jej używa
- **std::string load()** – ładuje wyniki z pliku oraz zwraca je jako string
- **std::string getFileName()** – zwraca nazwę pliku (utworzonego na podstawie nazwy aktualnej gry)