

Języki skryptowe

dokumentacja projektu: Interaktywne Kółko i Krzyżyk

Artur Mendela

Wydział Matematyki Stosowanej, Informatyka

grupa 3F

8 stycznia 2021

Część I

Opis programu








Projekt składa się z dwóch części:

- a) Konsolowe Menu
- b) Gra Kółko I Krzyżyk








Menu służy do indywidualnej personalizacji gry i wyboru graczy, do uruchamiania aplikacji, a dodatkowo pozwala nam zainstalować wymagany pakiet pygame.

Kółko i Krzyżyk to wszystkim dobrze znana gra strategiczna. Rozgrywana przez dwóch graczy. Gracze obejmują pola na przemian dążąc do objęcia trzech pól w jednej linii, przy jednoczesnym uniemożliwieniu tego samego przeciwnikowi.

Wszystkie pliki w folderze TicTacToe (jeśli jakiegoś brakuje, tworzy się automatycznie):

	KolkoKrzyzyk	07.01.2021 22:28	File folder	
	KolkoKrzyzyk.bat	07.01.2021 22:39	Windows Batch File	4 KB
	Kolor.txt	07.01.2021 22:28	Text Document	1 KB
	Name1.txt	07.01.2021 22:28	Text Document	1 KB
	Name2.txt	07.01.2021 22:28	Text Document	1 KB
	Wyniki.html	07.01.2021 22:33	Chrome HTML Do...	1 KB
	Zaczynajacy.txt	07.01.2021 22:28	Text Document	1 KB

Wszystkie pliki w folderze KolkoKrzyzyk (python):

	.idea	07.01.2021 23:01	File folder	
	venv	07.01.2021 21:40	File folder	
	get-pip.py	05.01.2021 19:24	Python File	1 843 KB
	ImageOfO.jpg	02.01.2021 14:08	JPG File	16 KB
	ImageOfX.jpg	02.01.2021 14:09	JPG File	14 KB
	main.py	07.01.2021 22:28	Python File	10 KB
	StartBoard.jpg	02.01.2021 14:02	JPG File	57 KB

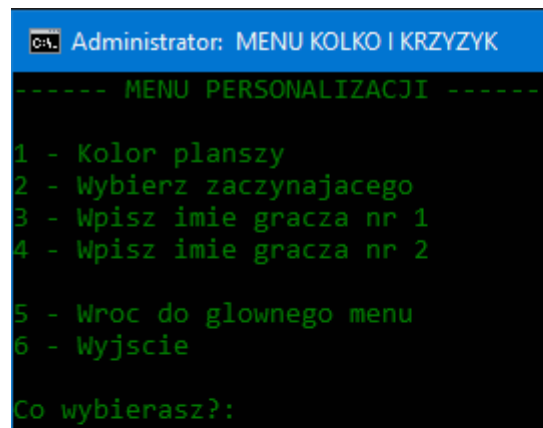
Konsolowe Menu:

```
C:\> Administrator: MENU KOLKO I KRZYZYK
----- MENU KOLKO I KRZYZYK -----
Folder TicTacToe musi znajdowac sie w sciezce "C:/", czyli na partycji C.

1 - Personalizacja
2 - Uruchom gre
3 - Wyjście
4 - ** INSTALACJA PAKIETU PYGAME **

Co wybierasz?:
```

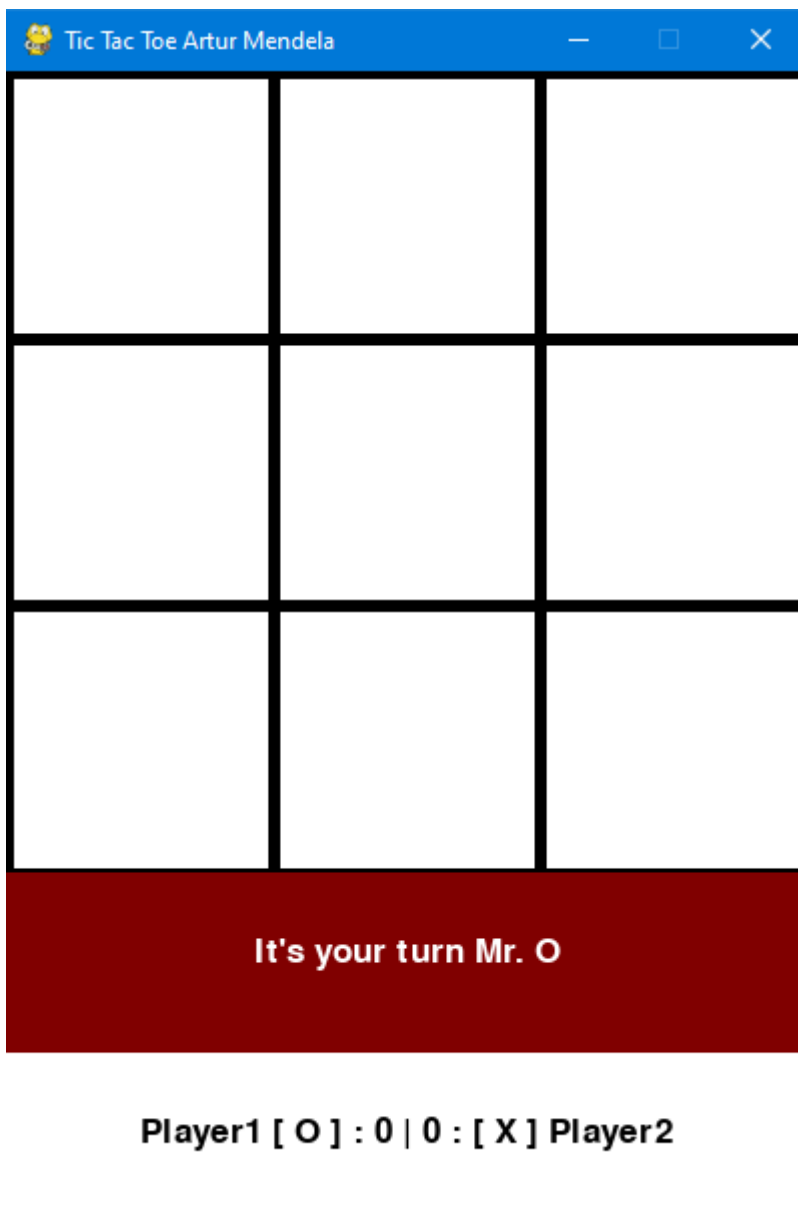
Konsolowe Menu (personalizacja):



The screenshot shows a Windows command prompt window with a blue title bar that reads "Administrator: MENU KOLKO I KRZYZYK". The console background is black, and the text is green. The menu is titled "----- MENU PERSONALIZACJI -----" and lists six options: 1 - Kolor planszy, 2 - Wybierz zaczynajacego, 3 - Wpisz imie gracza nr 1, 4 - Wpisz imie gracza nr 2, 5 - Wroc do glownego menu, and 6 - Wyjście. Below the menu, the prompt "Co wybierasz?:" is displayed.

```
Administrator: MENU KOLKO I KRZYZYK
----- MENU PERSONALIZACJI -----
1 - Kolor planszy
2 - Wybierz zaczynajacego
3 - Wpisz imie gracza nr 1
4 - Wpisz imie gracza nr 2
5 - Wroc do glownego menu
6 - Wyjście
Co wybierasz?:
```

Wygląd aplikacji:



Instrukcja obsługi

Folder TicTacToe, zawierający aplikację i wszelkie potrzebne elementy, umieścić należy na partycji C, tak żeby pełną ścieżką folderu była: C:/TicTacToe/. Grę możemy włączyć z poziomu załączonego Menu Konsolowego. Wystarczy, że uruchomimy plik KolkoKrzyzyk.bat i wpisujemy odpowiedni numer (w tym przypadku 2). Jeśli chcemy spersonalizować planszę, wpisać imiona graczy lub wybrać osobę, która rozpoczyna rozgrywkę, przechodzimy do Personalizacji (numer 1). Brakujący pakiet możemy zainstalować pod numerem 4, a z Menu wyjdziemy, jeśli wpisujemy numer 3.

Sama gra jest bardzo prosta i przyjemna w użyciu. Wskazany gracz zajmuje wybrane pole

Lewym Przyciskiem Myszy, po czym następuje ruch kolejnej osoby.

Dodatkowe informacje

Projekt stworzony przy użyciu interpretera Python 3.7. Do pełnego działania aplikacji wymagany jest pakiet pygame, który możemy zainstalować z poziomu Menu Konsolowego. Kolejne zaimportowane biblioteki to: time, sys, datetime.

Część II

Opis działania

Konsolowe Menu, napisane w języku batch, tworzy pliki tekstowe Kolor, Name1, Name2 i Zaczynajacy.txt (jesli takowe nie istnieją), a dodatkowo, w przypadku braku pliku Wyniki.html, tworzony również jest ten plik. Następnie, jeśli wybierzemy jakąkolwiek opcję personalizacji, pliki uzupełniane są przez odpowiednie dane. Gdy uruchamiamy grę, włącza się aplikacja pythonowa. Widzimy planszę w wybranym kolorze, a na tablicy wyników widoczne są spersonalizowane dane. Po każdej rundzie, tablica z wynikami jest aktualizowana, a wynik rozgrywki z wszelkimi potrzebnymi informacjami jest wysyłany do pliku Wyniki.html w estetycznej formie, dzięki włączonemu w aplikację skryptowi w języku Python. W pliku Wyniki.html tworzona jest historia rozgrywek.

Algorytm

Pseudokod (główna pętla):

```
Data: Dane wejściowe, to zawartości plików: Kolor.txt, Name1.txt, Name2.txt, Zaczynajacy.txt
Result: Dane wyjściowe, to informacje na temat przebiegłej rozgrywki, zapisane w pliku Wyniki.html
while True do
    for Każda akcja w programie do
        if Użytkownik kliknął przycisk wyjścia then
            Wydrukuj informację o poprawnym wyjściu z aplikacji. Zakończ działanie aplikacji Python, wróć do Konsolowego Menu;
        end
        if Użytkownik kliknął przycisk myszy then
            Sprawdź pole, a jeśli wolne i na planszy, to narysuj symbol.;
            if Ktoś wygrał then
                Zbierz informacje na temat rozgrywki i utwórz notatkę.
                Zmień symbol osoby zaczynającej na przeciwny.;
            end
            if Remis then
                Zbierz informację na temat rozgrywki i utwórz notatkę.
                Zmień symbol osoby zaczynającej na przeciwny.;
            end
        end
        Zapisz notatkę do pliku Wyniki.html;
    end
end
```

Algorithm 1: Algorytm drukowania informacji o liczbie parzystej/nieparzystej.

Historia rozgrywek

Zapis z rozegranych partii:

Wyniki rozgrywek Kolko i Krzyzyk by Artur Mendel

#####

Data:08/01/2021 12:37:30

Artur [O] vs Olga [X]

Zwyciezca: O

Zaczynajacy: O

Kolor planszy: blue

#####

#####

Data:08/01/2021 12:37:37

Artur [X] vs Olga [X]

Zwyciezca: O

Zaczynajacy: X

Kolor planszy: blue

#####

#####

Data:08/01/2021 12:37:48

Artur [O] vs Olga [X]

Zwyciezca: X

Zaczynajacy: O

Kolor planszy: blue

#####

#####

Data:08/01/2021 12:38:49

Random [O] vs Bartek [X]

Zwyciezca: X

Zaczynajacy: O

Kolor planszy: white

#####

Możliwe usprawnienia aplikacji

- a) Symbole O i X pojawiające się bez zbędnego tła
- b) Dodanie muzyki do gry
- c) Dodanie efektów dźwiękowych przy każdej akcji
- d) Sprawić, żeby folder nie musiał znajdować się na partycji C
- e) Dodanie ikonki do plików
- f) Estetyczne umieszczenie plików w katalogach
- g) Dodać czas trwania zakończonej rozgrywki w raporcie

Implementacja

Dane w plikach tekstowych są nadpisywane przy każdym uruchomieniu gry przez Menu.

```
1 echo Bialy > C:/TicTacToe/Kolor.txt
2 echo 0 > C:/TicTacToe/Zaczynajacy.txt
3 echo Player1 > C:/TicTacToe/Name1.txt
4 echo Player2 > C:/TicTacToe/Name2.txt
5
6 if not exist "C:/TicTacToe/Wyniki.html" (
```

Skrypt Batch tworzy również nagłówek do strony z wynikami.

```
1 echo
   -----
   >> C:/TicTacToe/Wyniki.html
2 echo ^<html^> >> C:/TicTacToe/Wyniki.html
3 echo ^<body^> >> C:/TicTacToe/Wyniki.html
4 echo ^<h1^>Wyniki rozgrywek Kolko i Krzyzyk by Artur Mendela: ^</h1^> >
   C:/TicTacToe/Wyniki.html
5 echo ^</body^> >> C:/TicTacToe/Wyniki.html
6 echo ^</html^> >> C:/TicTacToe/Wyniki.html
7 echo
   -----
   >> C:/TicTacToe/Wyniki.html
8 echo. >> C:/TicTacToe/Wyniki.html
9 )
```

Konsolowe Menu zbudowane jest głównie z komend If i Goto, które umożliwiają wybór odpowiedniej opcji i poruszanie się po poszczególnych sekcjach Menu.

```
1 :name1
2 cls
3 echo Imie gracza rozpoczynajacego
4 echo.
5 echo 1 - Wroc do glownego menu
6 echo 2 - Wyjscie
7 echo.
8 Set /p opcja4=Podaj imie gracza 1 lub wybierz opcje:
9
10 if %opcja4%==1 Goto menu
11 if %opcja4%==2 Goto exit
12 if %opcja4% NEQ 1 (
```

```

13  if %opcja4% NEQ 1 (
14  echo %opcja4% > C:/TicTacToe/Name1.txt
15  pause
16  Goto personalizacja
17  )
18  )
19  :name2
20  cls
21  echo Imie gracza drugiego
22  echo.
23  echo 1 - Wroc do glownego menu
24  echo 2 - Wyjscie
25  echo.
26  Set /p opcja5=Podaj imie gracza 2 lub wybierz opcje:
27  if %opcja5%==1 Goto menu
28  if %opcja5%==2 Goto exit
29  if %opcja5% NEQ 1 (
30  if %opcja5% NEQ 1 (
31  echo %opcja5% > C:/TicTacToe/Name2.txt
32  Goto personalizacja
33  )
34  )

```

Dane z plików pobierane są w aplikacji Pythonowej, po czym przypisywane są do odpowiednich zmiennych:

```

1  Fcolor = open("C:/TicTacToe/Kolor.txt")
2  FName1 = open("C:/TicTacToe/Name1.txt")
3  FName2 = open("C:/TicTacToe/Name2.txt")
4  FStartingPlayer = open("C:/TicTacToe/Zaczynajacy.txt")
5
6  color = Fcolor.read().replace(" ", "").replace("\n", "")
7  name1 = FName1.read().replace(" ", "").replace("\n", "")
8  name2 = FName2.read().replace(" ", "").replace("\n", "")
9  StartingPlayerSymbol = FStartingPlayer.read().replace(" ", "").replace("\n", "")
10
11 if StartingPlayerSymbol == "0":
12     SecondPlayerSymbol="X"
13     StartingPlayerSymbol="0"
14     print("Gracz startujacy: " + StartingPlayerSymbol)
15 else:
16     SecondPlayerSymbol="0"
17     StartingPlayerSymbol="X"
18     print("Gracz startujacy: " + StartingPlayerSymbol)

```

Tworzenie głównych ustawień i dodatkowych kolorów dla ułatwienia.

```

1  # wall and rules
2  WhoseTurn = StartingPlayerSymbol
3  TheWinner = None
4  Draw = False
5
6  # the board
7  Width = 400

```

```

8 Height = 400
9 Board = [[None] * 3, [None] * 3, [None] * 3]
10
11 # some colors variables to make it easier later
12 black = (0, 0, 0)
13 white = (255, 255, 255)
14 blue = (0, 191, 255)

```

Ustawienia GUI, m.in. pobieranie i skalowanie obrazów symboli i planszy.

```

1  # APP GUI settings
2
3  x = pg.init() # make able to use the pygame package
4  FPS = 30
5  TimeClock = pg.time.Clock()
6  MainDisplay = pg.display.set_mode((Width, Height + 180), 0, 32, 0, 0)
7  pg.display.set_caption("Tic Tac Toe Artur Mendela", "Tic Tac Toe Artur
   Mendela")
8
9  StartingBoard = pg.image.load("StartBoard.jpg")
10 ImageOfX = pg.image.load("ImageOfX.jpg")
11 ImageOf0 = pg.image.load("ImageOf0.jpg")
12
13 StartingBoard = pg.transform.scale(StartingBoard, (Width, Height + 180))
14 ImageOfX = pg.transform.scale(ImageOfX, (80, 80))
15 ImageOf0 = pg.transform.scale(ImageOf0, (80, 80))

```

Dla przykładu jedna z funkcji, tutaj tworząca planszę.

```

1 def CreateBoard():
2     global color
3     MainDisplay.blit(StartingBoard, (0, 0))
4     pg.display.update()
5     time.sleep(1)
6
7
8     if color == "blue":
9         MainDisplay.fill(blue)
10        print("Kolor planszy: " + color)
11    else:
12        print("Kolor planszy: " + color)
13        MainDisplay.fill(white)
14
15    # * draw a grid *
16
17    # vertical lines
18    pg.draw.line(MainDisplay, black, (Width / 3, 0), (Width / 3, Height), 6)
19    pg.draw.line(MainDisplay, black, (Width / 3 * 2, 0), (Width / 3 * 2, Height), 6)
20    pg.draw.line(MainDisplay, black, (0, 0), (0, Height), 6)
21    pg.draw.line(MainDisplay, black, (Width, 0), (Width, Height), 6)
22
23    # horizontal lines
24    pg.draw.line(MainDisplay, black, (0, 0), (Width, 0), 6)

```

```

25     pg.draw.line(MainDisplay, black, (0, Height / 3), (Width, Height /
26                    3), 6)
27     pg.draw.line(MainDisplay, black, (0, Height / 3 * 2), (Width, Height
28                    / 3 * 2), 6)
29     pg.draw.line(MainDisplay, black, (0, Height), (Width, Height), 6)
30
31     pg.display.update()
32     DrawAdditionalInfo()

```

Kolejna ciekawa funkcja sprawdzająca, czy rozgrywka się zakończyła - jeśli tak, to dodatkowo rysująca linię i zapisująca dane do odpowiednich zmiennych.

```

1  def WhetherWin():
2      global Board, TheWinner, Draw
3
4      # check if there is a horizontal winning line (czy ktos wygral w
5      # poziomie)
6
7      for Row in range(0, 3):
8          if ((Board[Row][0]) == Board[Row][1] == Board[Row][2]) and (
9              Board[Row][0] is not None):
10             TheWinner = Board[Row][0]
11             pg.draw.line(MainDisplay, (128, 0, 0), (0, (Row + 1) *
12                    Height / 3 - Height / 6),
13                    (Width, (Row + 1) * Height / 3 - Height / 6),
14                    4)
15
16             break
17
18      # check if there is a vertical winning line (czy ktos wygral w
19      # pionie)
20
21      for Column in range(0, 3):
22          if ((Board[0][Column] == Board[1][Column] == Board[2][Column])
23              and (Board[0][Column] is not None)):
24             TheWinner = Board[0][Column]
25             pg.draw.line(MainDisplay, (128, 0, 0), (Width / 3 * (Column
26                    + 1) - Width / 6, 0),
27                    (Width / 3 * (Column + 1) - Width / 6, Height),
28                    4)
29
30             break
31
32      # check if there is a diagonal winning line (czy ktos wygral w
33      # ukosie)
34
35      if ((Board[0][0] == Board[1][1] == Board[2][2]) and (Board[0][0] is
36          not None):
37          TheWinner = Board[0][0]
38          pg.draw.line(MainDisplay, (128, 0, 0), (0, 0), (Width, Height),
39                      5)
40
41      if ((Board[0][2] == Board[1][1] == Board[2][0]) and (Board[0][2] is
42          not None):
43          TheWinner = Board[0][2]
44          pg.draw.line(MainDisplay, (128, 0, 0), (Width, 0), (0, Height),
45                      5)

```

```
31     # there is no winner
32     if (all([all(Row) for Row in Board]) and TheWinner is None):
33         Draw = True
34
35     # DRAW DRAW
36     DrawAdditionalInfo()
```

Reszta funkcji przedstawiona jest w pełnym kodzie aplikacji, tuż poniżej.

Pełen kod aplikacji

```
1 import pygame as pg
2 import sys
3 import time
4 from datetime import datetime
5 from pygame.locals import *
6
7 # LOADING PERSONALIZATION SETTINGS FROM FILES @@@
8 Fcolor = open("C:/TicTacToe/Kolor.txt")
9 FName1 = open("C:/TicTacToe/Name1.txt")
10 FName2 = open("C:/TicTacToe/Name2.txt")
11 FStartingPlayer = open("C:/TicTacToe/Zaczynajacy.txt")
12
13 color = Fcolor.read().replace(" ", "").replace("\n", "")
14 name1 = FName1.read().replace(" ", "").replace("\n", "")
15 name2 = FName2.read().replace(" ", "").replace("\n", "")
16 StartingPlayerSymbol = FStartingPlayer.read().replace(" ", "").replace("\n", "")
17
18 if StartingPlayerSymbol == "0":
19     SecondPlayerSymbol="X"
20     StartingPlayerSymbol="0"
21     print("Gracz startujacy: " + StartingPlayerSymbol)
22 else:
23     SecondPlayerSymbol="0"
24     StartingPlayerSymbol="X"
25     print("Gracz startujacy: " + StartingPlayerSymbol)
26
27
28 #     *** Main settings ***
29
30 # wall and rules
31 WhoseTurn = StartingPlayerSymbol
32 TheWinner = None
33 Draw = False
34
35 # the board
36 Width = 400
37 Height = 400
38 Board = [[None] * 3, [None] * 3, [None] * 3]
39
40 # some colors variables to make it easier later
41 black = (0, 0, 0)
42 white = (255, 255, 255)
43 blue = (0, 191, 255)
44
45
46
47 # counting points to show a result
48
49 FirstPlayerPoints = 0
50 SecondPlayerPoints = 0
51
```

```

52 # APP GUI settings
53
54 x = pg.init() # make able to use the pygame package
55 FPS = 30
56 TimeClock = pg.time.Clock()
57 MainDisplay = pg.display.set_mode((Width, Height + 180), 0, 32, 0, 0)
58 pg.display.set_caption("Tic Tac Toe Artur Mendela", "Tic Tac Toe Artur
    Mendela")
59
60 StartingBoard = pg.image.load("StartBoard.jpg")
61 ImageOfX = pg.image.load("ImageOfX.jpg")
62 ImageOfO = pg.image.load("ImageOfO.jpg")
63
64 StartingBoard = pg.transform.scale(StartingBoard, (Width, Height + 180))
65 ImageOfX = pg.transform.scale(ImageOfX, (80, 80))
66 ImageOfO = pg.transform.scale(ImageOfO, (80, 80))
67
68
69 # *** several functions ***
70
71 def CreateBoard():
72     global color
73     MainDisplay.blit(StartingBoard, (0, 0))
74     pg.display.update()
75     time.sleep(1)
76
77
78     if color == "blue":
79         MainDisplay.fill(blue)
80         print("Kolor planszy: " + color)
81     else:
82         print("Kolor planszy: " + color)
83         MainDisplay.fill(white)
84
85 # * draw a grid *
86
87 # vertical lines
88 pg.draw.line(MainDisplay, black, (Width / 3, 0), (Width / 3, Height)
    , 6)
89 pg.draw.line(MainDisplay, black, (Width / 3 * 2, 0), (Width / 3 * 2,
    Height), 6)
90 pg.draw.line(MainDisplay, black, (0, 0), (0, Height), 6)
91 pg.draw.line(MainDisplay, black, (Width, 0), (Width, Height), 6)
92
93 # horizontal lines
94 pg.draw.line(MainDisplay, black, (0, 0), (Width, 0), 6)
95 pg.draw.line(MainDisplay, black, (0, Height / 3), (Width, Height /
    3), 6)
96 pg.draw.line(MainDisplay, black, (0, Height / 3 * 2), (Width, Height
    / 3 * 2), 6)
97 pg.draw.line(MainDisplay, black, (0, Height), (Width, Height), 6)
98
99 pg.display.update()
100 DrawAdditionalInfo()
101

```

```

102
103 def DrawAdditionalInfo():
104     global Draw, SecondPlayerSymbol, WhoseTurn, StartingPlayerSymbol
105
106     if TheWinner is None:
107         NewMessage = "It's your turn Mr. " + str(WhoseTurn)
108
109     else:
110         NewMessage = "You won, Mr. " + str(TheWinner)
111
112     if Draw:
113         NewMessage = "It's a DRAW, try again! "
114
115 #GUI SCORE ETC.
116
117     Font = pg.font.Font(None, 25)
118     MessageResult = Font.render(NewMessage, True, white)
119     Points = name1 + " [ " + StartingPlayerSymbol + " ] : " + str(
120         FirstPlayerPoints) + " | " + str(SecondPlayerPoints) + " : " + "
121         [ " + SecondPlayerSymbol + " ] " + name2
122     MessagePoints = Font.render(Points, True, black)
123
124     MessageResultSurround = MessageResult.get_rect(center=(Width / 2,
125         480 - 40))
126     MessagePointsSurround = MessagePoints.get_rect(center=(Width / 2,
127         580 - 50))
128
129     MainDisplay.fill((128, 0, 0), (0, 400, 400, 90))
130     MainDisplay.blit(MessageResult, MessageResultSurround)
131     MainDisplay.blit(MessagePoints, MessagePointsSurround)
132
133     pg.display.update()
134
135
136 def WhetherWin():
137     global Board, TheWinner, Draw
138
139     # check if there is a horizontal winning line (czy ktos wygral w
140     # poziomie)
141
142     for Row in range(0, 3):
143         if ((Board[Row][0]) == Board[Row][1] == Board[Row][2]) and (
144             Board[Row][0] is not None):
145             TheWinner = Board[Row][0]
146             pg.draw.line(MainDisplay, (128, 0, 0), (0, (Row + 1) *
147                 Height / 3 - Height / 6),
148                 (Width, (Row + 1) * Height / 3 - Height / 6),
149                 4)
150
151             break
152
153     # check if there is a vertical winning line (czy ktos wygral w
154     # pionie)
155
156     for Column in range(0, 3):
157         if ((Board[0][Column] == Board[1][Column] == Board[2][Column])

```



```

        and (Board[0][Column] is not None)):
148     TheWinner = Board[0][Column]
149     pg.draw.line(MainDisplay, (128, 0, 0), (Width / 3 * (Column
        + 1) - Width / 6, 0),
150                 (Width / 3 * (Column + 1) - Width / 6, Height),
                    4)
151     break
152
153     # check if there is a diagonal winning line (czy ma ukosie)
154     if ((Board[0][0] == Board[1][1] == Board[2][2]) and (Board[0][0] is
        not None):
155         TheWinner = Board[0][0]
156         pg.draw.line(MainDisplay, (128, 0, 0), (0, 0), (Width, Height),
            5)
157
158     if ((Board[0][2] == Board[1][1] == Board[2][0]) and (Board[0][2] is
        not None):
159         TheWinner = Board[0][2]
160         pg.draw.line(MainDisplay, (128, 0, 0), (Width, 0), (0, Height),
            5)
161
162     # there is no winner
163     if (all([all(Row) for Row in Board]) and TheWinner is None):
164         Draw = True
165
166     # DRAW DRAW
167     DrawAdditionalInfo()
168
169
170 def DrawSymbol(Row, Column):
171     global Board, WhoseTurn, XPos, YPos
172     # position X of new symbol image
173     if Row == 1:
174         YPos = 30
175
176     elif Row == 2:
177         YPos = Width / 3 + 30
178
179     elif Row == 3:
180         YPos = Width / 3 * 2 + 30
181     # position Y of new symbol image
182     if Column == 1:
183         XPos = 30
184
185     elif Column == 2:
186         XPos = Height / 3 + 30
187
188     elif Column == 3:
189         XPos = Height / 3 * 2 + 30
190
191     Board[Row - 1][Column - 1] = WhoseTurn
192
193     if (WhoseTurn == 'X'):
194         MainDisplay.blit(ImageOfX, (XPos, YPos))

```

```

195         WhoseTurn = 'O'
196     elif (WhoseTurn == 'O'):
197         MainDisplay.blit(ImageOfO, (XPos, YPos))
198         WhoseTurn = 'X'
199
200     DrawAdditionalInfo()
201     pg.display.update()
202
203
204 def CheckField():
205     # mouse position
206     global Column
207     global Row
208     pos = pg.mouse.get_pos()
209
210     # horizontal X
211     if (pos[0] < Width / 3):
212         Column = 1
213     elif (pos[0] < Width / 3 * 2):
214         Column = 2
215     elif (pos[0] < Width):
216         Column = 3
217     else:
218         Column = None
219
220     # vertical Y
221     if (pos[1] < Height / 3):
222         Row = 1
223     elif (pos[1] < Height / 3 * 2):
224         Row = 2
225     elif (pos[1] < Height):
226         Row = 3
227
228     else:
229         Row = None
230
231     print(WhoseTurn + " na pole: " + str(Row), str(Column))
232     if Row is not None and Column is not None and Board[Row - 1][Column
233         - 1] is None:
234         DrawSymbol(Row, Column)
235         WhetherWin()
236
237 CreateBoard()
238
239
240 def GameRestart():
241     global Board, WhoseTurn, TheWinner, Draw, SecondPlayerPoints,
242         FirstPlayerPoints, name1, name2, StartingPlayerSymbol,
243         SecondPlayerSymbol
244     time.sleep(3)
245     WhoseTurn = 'X'
246     Draw = False
247     if TheWinner == StartingPlayerSymbol:
248         FirstPlayerPoints += 1

```

```

247
248     elif TheWinner == SecondPlayerSymbol:
249         SecondPlayerPoints +=1
250
251     TheWinner = None
252     Board = [[None] * 3, [None] * 3, [None] * 3]
253     MainDisplay.fill(white)
254     CreateBoard()
255     pg.display.update()
256
257
258 while (True):
259     for Event in pg.event.get():
260         pg.display.update()
261         if Event.type == pg.QUIT:
262             print('Poprawne wyjście z gry.')
263             pg.quit()
264             sys.exit()
265
266         elif Event.type == pg.MOUSEBUTTONDOWN:
267             CheckField()
268             if (TheWinner or Draw):
269                 text = ''
270                 now = datetime.now()
271                 realtimeformat = now.strftime("%d/%m/%Y %H:%M:%S")
272                 if (TheWinner):
273                     text = '''<br> #
274                         ##### <br> <br>
275                         > Data: to''' + realtimeformat + "<br>" + name1 +\
276                         " [" + StartingPlayerSymbol + "] vs " + name2
277                         + " [" + SecondPlayerSymbol + "]" + " " + '''
278                     <br> Zwyciezca: ''' + str(TheWinner) + '''<br>
279                     Zaczynajacy: ''' +\
280                     StartingPlayerSymbol + ''' <br> Kolor planszy
281                     : ''' + color + '''
282
283                     <br> <br>
284                     #
285                     #####
286
287                     '''
288                 if (Draw):
289                     print("REMIS")
290                     text = ''' CZAREK ''' + str(TheWinner) + '''
291
292                     < html >
293                     < body >
294                     < h1 > Heading <
295                         / h1 >
296                     < / body >
297                     < / html >
298                     '''
299                     file = open("C:/TicTacToe/Wyniki.html", "a")
300                     file.write(text)
301                     file.close()
302                     GameRestart()

```

```
294         pg.display.update()
295
296 TimeClock.tick(FPS)
297 DrawAdditionalInfo()
```
