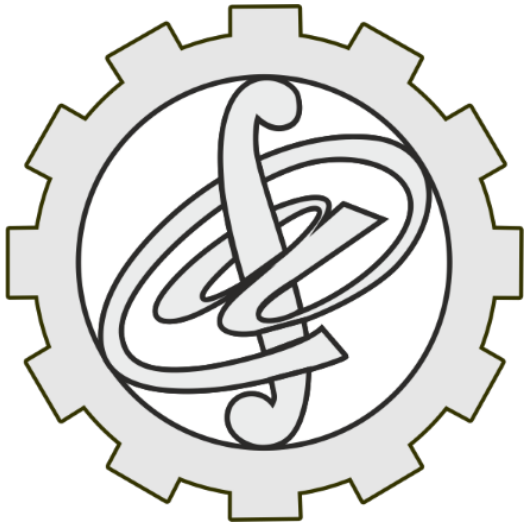
Języki Skryptowe

Dokumentacja projektu "Kółko i Krzyżyk" - Algorytmion 2014

Patryk Gamrat, grupa 2/4
Politechnika Śląska, Wydział Matematyki Stosowanej

22 stycznia 2024



Spis treści

1. Informacje wprowadzające

1.1. Założenia projektu

Głównym założeniem projektu bylo zaimplementowanie systemu, realizującego wybrane zadanie z konkursu Algorytmion. Dla wprowadzonych danych wejściowych realizowane są obliczenia a następie na podstawie wyników generowany jest raport. W projekcie korzystamy ze skryptów w języku Python oraz Batch, a zatem wykorzystujemy całą wiedzę zdobytą w ramach przedmiotu.

1.2. Opis programu

Program realizuje zadanie 2 z edycji 2014 konkursu Algorytmion - "Kółko i Krzyżyk". Zadanie to polega na odczytaniu z pliku tekstowego, które reprezentuje planszę 5x5 do gry w kółko i krzyżyk. Naszym celem jest obliczenie punktacji dla Kacpra umieszczającego "x" oraz dla Olka, który umieszcza "o". Punkty są liczone w zależności od ilości symboli w jednym wierszu kolumnie lub diagonali. Na koniec należy obliczyć sumę punktów i wyznaczyć zwycięzce.

Pełna treść zadania

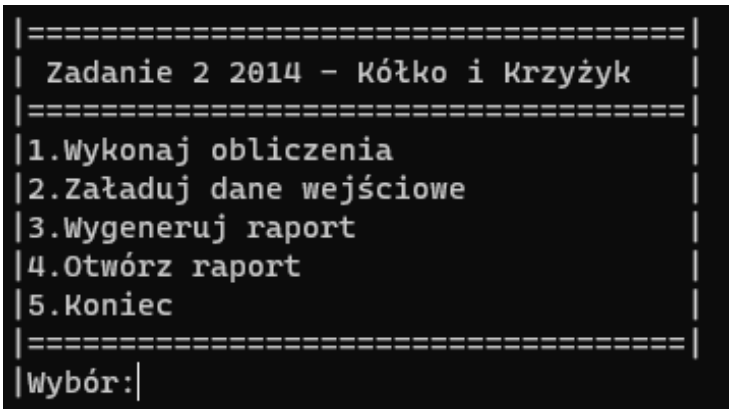
```
Kacper i Olek grają w "kółko i krzyżyk" na planszy 5x5, zaznaczając
na przemian pola (Kacper - krzyżyki, Olek - kółka). Umówili się, że
punkty będą liczyć po zakończeniu gry według poniższej reguły:

2 kółka lub krzyżyki w jednym wierszu, kolumnie lub diagonali - 1pkt,
3 kółka lub krzyżyki w jednym wierszu, kolumnie lub diagonali - 3pkt,
4 kółka lub krzyżyki w jednym wierszu, kolumnie lub diagonali - 7pkt,
5 kółka lub krzyżyki w jednym wierszu, kolumnie lub diagonali - 15pkt.

Twoim zadaniem jest odczytanie z pliku tekstowego danego układu
i poprawne przeliczenie punktów uzyskanych przez zawodników
i wyświetlenie ich na ekranie monitora.
```

```
Przykład.
x o x o x
o x o o o
x x x o x
o o x x o
o x x o o
punkty dla gracza umieszczającego "x": 1*7+4*3+5*1=24pkt
punkty dla gracza umieszczającego "o": 1*7+2*3+10*1=23pkt
wygrał Kacper.
Uwaga!
W pliku tekstowym gra.txt znajduje się pięć wierszy, w każdym
pięć znaków: "o" lub "x".
```

Realizacją częsci obliczeniowej tego zadania zajmuje się skrypt python. Pliki z układem planszy są przekazywane za pomocą skryptu batch, który działa jako menu użytkownika. Na podstawie wykonanych obliczeń, osobny skrypt generuje raport w formie pliku html, który zawiera liste odczytanych układów wraz z wynikami i zwycięzcą dla każdego z nich.

1.3. Instrukcja obsługi

Aby uruchomić projekt należy rozpakować dołączony plik. Następnie uruchamiamy skrypt batch nazwany **TicTacToe.bat**.



Zrzut 1: Menu

Po uruchomieniu skryptu, uzyskamy dostęp do menu. Opis poszczególnych opcji:
1. - Wykonuje skrypt python realizujący obliczenia na załadowanych danych wejściowych
2. - Wczytuje katalog z plikami .txt zawierającymi układy plansz
3. - Na podstawie wyników z katalogu *out* generuje raport
4. - Otwiera wygenerowany plik raport.html
5. - Zamyka konsolę

Aby wygenerować raport należy najpierw wybrać opcję **2.** oraz podać scieżkę do katalogu z plikami wejściowymi. Wraz z projektem dostarczony został katalog *in*. Zawiera on przykładowe pliki wejściowe z których można skorzystać.

Po załadowaniu danych możemy wybrać opcję **1.** aby wykonać obliczenia. Na ekranie konsoli wyświetli się komunikat z informacją o przetwarzanych plikach.
Ostatnim krokiem jest wygenerowanie raportu i otworzenie go za pomocą opcji **3.** oraz **4.**

Algorytmion 2014 Zadanie 2 - "Kółko i Krzyżyk"

Raport wykonanych obliczeń

Raport nr.1 (Plik out0.txt)

X	O	X	O	X
O	X	O	O	O
X	X	X	O	X
O	O	X	X	O
O	X	X	O	O

Punkty dla gracza umieszczającego "x":
0*15 + 1*7 + 4*3 + 5*1 = 24
Punkty dla gracza umieszczającego "o":
0*15 + 1*7 + 2*3 + 10*1 = 23
Wygrał Kacper

Zrzut 2: Wygenerowany raport

Aby program działał poprawnie nie można modyfikować plkiów wewnątrz katalogu *src*, należy również pamiętać aby katalog ten znajdował się w tym samym katalogu w którym znajduje się skrypt batch.

2. Realizacja projektu

2.1. Opis działania

Interakcja z użytkownikiem w programie jest ralizowana za pomocą skryptu batch. W zależności od opcji wybranej przez użytkownika ładowane są odpowiednie pliki lub uruchamiany konkretny skrypt python.

Poniżej znajduje się fragment kodu realizujący uruchamianie skryptu obliczeniowego na plikach w wybranym wcześniej katalogu.

```
:opcja1
if defined input (
  set /a i=0
  cd /d "!input!"
  if not exist "%~dp0out\" mkdir "%~dp0out"
  for /r %%x in (*.txt) do (
    py "%~dp0\src\SolveBoard.py" "%%x" "%~dp0out\out!i!.txt"
    if exist "%~dp0out\out!i!.txt" set /a i=!i!+1
  )
  cd /d "%~dp0"
  echo Wykonano obliczenia na plikach z katalogu %input%
) else (
  echo Nie załadowano katalogu z danymi wejściowymi
)
pause
goto :menu
```

Po wybraniu opcji **1.** w menu skrypt sprawdza czy został podany katalog z plikami wejściowymi, jeśli tak to skrypt przechodzi do tego katalogu. Za pomocą pętli dla wszystkich plików z rozszerzeniem .txt w katalogu wykonywany jest skrypt **SolveBoard.py**, który rozwiązuje układ planszy. Wyniki zapisywane są do katalogu *out*.

Skrypt **SolveBoard.py** wczytyuje plik wejściowy oraz plik do zapisu jako argumenty. Pierwszym krokiem jaki wykonuje kod, jest załadowanie układu planszy z pliku. Ponieważ skrypt batch nie zajmuje się walidacją poprawności pliku, dopiero w tym kroku sprawdzane jest czy podany układ planszy jest odpowiednio zapisany. W przypadku niepoprawnego pliku wejściowego, wyrzucany jest odpowiedni wyjątek.

```
def loadBoard(filename: str) -> list:
  loaded_board = []
  try:
    with open(filename, encoding='utf=8') as board_file:
      lines = board_file.readlines()
      lines = [l.strip() for l in lines]
      for line in lines:
        loaded_board.append(line.split(' '))

    # Sprawdzanie poprawności wczytania danych
    if len(loaded_board) != 5:
      raise IOError
    for l in range(5):
      if len(loaded_board[l]) != 5 or not all(s == 'x' or s
   == 'o' for s in loaded_board[l]):
        raise IOError
  except IOError:
    print(f"Plik {filename} jest niepoprawny!")

  print(f"Pomyślnie załadowano plik {filename} z planszą")
  return loaded_board
```

Wyniki działania skryptu zapisywane są w plikach **out.txt**. Poniżej znajduje się przykład takowego pliku

```
x o x o x o x o o o x x x o x o o x x o o x x o o
0*15 + 1*7 + 4*3 + 5*1 = 24
0*15 + 1*7 + 2*3 + 10*1 = 23
Wygrał Kacper
```

Po wybraniu odpowiedniej opcji w menu, pliki **out.txt** są przetwarzane przez skrypt **RaportGen.py**, który wstrzykuje dane z pliku do stworzonego wcześniej szabolnu dokumentu html. Poniżej znajduje się schemat prezentujący sposób działania programu

2.2. Algorytm obliczający wyniki

Poniżej znajduje się pseudokod algorytmu, który oblicza ilość punktów dla obu graczy.

```
Data: Tablica tab z układem planszy
Result: Ilość punktów dla obu graczy
Zainicjuj tablicę trójwymiarową usedPatterns wartościami false
punktyKacper = 0
punktyOlek = 0
foreach symbol in tab do
    foreach kierunek do
        n = 0
        while Przesuwając się w danym kierunku napotykamy
          aktualny symbol do
            n += 1
            Ustaw wartość usedPatterns sprawdzanych symboli i dla
              obecnego kierunku na True
        end
        Przelicz n na punkty według punktacji z treści zadania
        if symbol = 'x' then
            Dodaj punkty do punktyKacper
        end
        else
            Dodaj punkty do punktyOlek
        end
    end
end
```

Najbardziej kluczowym elementem algorytmu jest tablica trójwymiarowa **usedPatterns[5][5][4]**. Pierwsze dwa indeksy są pozycję na planszy 5x5, natomiast ostatni określa o jaki kierunek chodzi (0 - wiersze, 1 - kolumny, 2 - diagonale lewe, 3 - diagonale prawe). W tablicy przechowujemy informację, czy dany symbol był już wykorzystany w wybranym kierunku.

Implementacja powyższego algorytmu w skrypcie python:

```
board = loadBoard(input_file)
if board is not None:
  used_symbols = [[[False for _ in range(4)] for _ in range
   (5)] for _ in range(5)]
  kacper_points = [0 for _ in range(4)]
  olek_points = [0 for _ in range(4)]
  for i in range(5):
    for j in range(5):
      symbol = board[i][j]

      # Wiersze
      n = 0
      while (j + n < 5) and (board[i][j + n] == symbol) and (
   used_symbols[i][j + n][0] == False):
        used_symbols[i][j + n][0] = True
        n += 1
      if symbol == 'x':
        addPoints(n, kacper_points)
      else:
        addPoints(n, olek_points)

      # Kolumny
      n = 0
        while (i + n < 5) and (board[i + n][j] == symbol) and
    (used_symbols[i + n][j][1] == False):
          used_symbols[i + n][j][1] = True
          n += 1
      if symbol == 'x':
        addPoints(n, kacper_points)
      else:
        addPoints(n, olek_points)

      # Diagonale lewe
      n = 0
      while (i + n < 5) and (j - n >= 0) and (board[i + n][j
   - n] == symbol) and (used_symbols[i + n][j - n][2] ==
   False):
        used_symbols[i + n][j - n][2] = True
        n += 1
      if symbol == 'x':
        addPoints(n, kacper_points)
      else:
        addPoints(n, olek_points)

```

```
40        # Diagonale prawe
41        n = 0
42        while (i + n < 5) and (j + n < 5) and (board[i + n][j +
      n] == symbol) and (used_symbols[i + n][j + n][3] == False
      ):
43          used_symbols[i + n][j + n][3] = True
44          n += 1
45        if symbol == 'x':
46          addPoints(n, kacper_points)
47        else:
48          addPoints(n, olek_points)
```

2.3. Analiza wyników

Sprawdzenie poprawności wyników dla przykładowych układów planszy
Przykład 1
Dla przykładu z treści zadania

```
x o x o x
o x o o o
x x x o x
o o x x o
o x x o o
```

Otrzymujemy wyniki

```
punkty dla gracza umieszczającego "x": 0*15 + 1*7 + 4*3 + 5*1 = 24pkt
punkty dla gracza umieszczającego "o": 0*15 + 1*7 + 2*3 + 10*1 = 23pkt
Wygrał Kacper
```

Wynik jest zgodny z wynikiem podanym w treści zadania

Przykład 2
Dla pliku wejściowego

```
x o x o o
x o x o o
x o x o x
x o o x o
o o x x x
```

Otrzymujemy wyniki

```
punkty dla gracza umieszczającego "x": 0*15 + 1*7 + 4*3 + 1*1 = 20pkt
punkty dla gracza umieszczającego "o": 1*15 + 1*7 + 1*3 + 10*1 = 35pkt
Wygrał Olek
```

Zweryfikujmy wynik i policzmy punkty Kacpra

```
 Wiersze:
 x o x o o
 x o x o o
 x o x o x
 x o o x o
 o o x x x
 1*3pkt
 Kolumny:
 x o x o o
 x o x o o
 x o x o x
 x o o x o
 o o x x x
 1*7+1*3+1*1=11pkt
 Diagonale lewe:
 x o x o o
 x o x o o
 x o x o x
 x o o x o
 o o x x x
 1*3=3pkt
 Diagonale prawe:
 x o x o o
 x o x o o
 x o x o x
 x o o x o
 o o x x x
 1*3=3pkt
 Suma: 3 + 11 + 3 + 3 = 20pkt
```

Ilosć punktów Kacpra jest zgodna z ilością punktów obliczoną przez algorytm. Teraz obliczymy punkty Olka.

```
Wiersze:
x o x o o
x o x o o
x o x o x
x o o x o
o o x x x
4*1=4pkt
Kolumny:
x o x o o
x o x o o
x o x o x
x o o x o
o o x x x
1*15+1*3+1*1=19pkt
Diagonale lewe:
x o x o o
x o x o o
x o x o x
x o o x o
o o x x x
1*7+2*1=9pkt
Diagonale prawe:
x o x o o
x o x o o
x o x o x
x o o x o
o o x x x
3*1=3pkt
Suma: 4 + 19 + 9 + 3 = 35pkt
```

Ilość punktów Olka również jest zgodna z oczekiwanym wynikiem. A zatem wyniki obliczane przez algorytm są poprawne.

3. Kod źródłowy aplikacji

3.1. Skrypt Batch

```
echo offsetlocal EnableDelayedExpansionchcp 1250
:menu
  cls
  echo ^|====================================^|
  echo ^| Zadanie 2 2014 - Kółko i Krzyżyk   ^|
  echo ^|====================================^|
  echo ^|1.Wykonaj obliczenia                ^|
  echo ^|2.Załaduj dane wejściowe            ^|
  echo ^|3.Wygeneruj raport                  ^|
  echo ^|4.Otwórz raport                     ^|
  echo ^|5.Koniec                            ^|
  echo ^|====================================^|
  set /p choice=^|Wybór:
  if "%choice%"=="1" ( goto :opcja1 )
  if "%choice%"=="2" ( goto :opcja2 )
  if "%choice%"=="3" ( goto :opcja3 )
  if "%choice%"=="4" ( goto :opcja4 )
  if "%choice%"=="5" ( goto :eof) else ( goto :menu )

:opcja1
  if defined input (
    set /a i=0
    cd /d "!input!"
    if not exist "%~dp0out\" mkdir "%~dp0out"
    for /r %%x in (*.txt) do (
      py "%~dp0\src\SolveBoard.py" "%%x" "%~dp0out\out!i!.txt
   "
      if exist "%~dp0out\out!i!.txt" set /a i=!i!+1
    )
    cd /d "%~dp0"
    echo Wykonano obliczenia na plikach z katalogu %input%
    ) else (
      echo Nie załadowano katalogu z danymi wejściowymi
    )
  pause
  goto :menu
:opcja2
  set /p input=Podaj scieżkę katalogu z danymi wejściowymi:
  if exist !input! (
    echo Pomyślnie załadowano katalog %input%
```

```
  ) else (
    echo Nie udało się załadować katalogu %input%
    set input=
  )
  pause
  goto :menu
:opcja3
  py "%~dp0\src\RaportGen.py" "%~dp0out" "raport.html"
  pause
  goto :menu
:opcja4
  if exist raport.html (
    echo Otwieram plik raport.html
    start raport.html
  ) else (
    echo Nie znaleziono raportu
  )
  pause
  goto :menu
```

3.2. Skrypt Python realizujący obliczenia

```
import sys


# Ładuje planszę
def loadBoard(filename: str) -> list:
    loaded_board = []
    try:
        with open(filename, encoding='utf=8') as board_file:
            lines = board_file.readlines()
            lines = [l.strip() for l in lines]
            for line in lines:
                loaded_board.append(line.split(' '))

        # Sprawdzanie poprawności wczytania danych
        if len(loaded_board) != 5:
            raise IOError
        for l in range(5):
            if len(loaded_board[l]) != 5 or not all(s == 'x'
   or s == 'o' for s in loaded_board[l]):
                raise IOError
    except IOError:
        print(f"Plik {filename} jest niepoprawny!")
```

```
22
23      print(f"Pomyślnie załadowano plik {filename} z planszą")
24      return loaded_board
25
26
27  def addPoints(count: int, points: list):
28      match count:
29          # 1 punkt
30          case 2:
31              points[0] += 1
32          # 3 punkty
33          case 3:
34              points[1] += 1
35          # 7 punktow
36          case 4:
37              points[2] += 1
38          # 15 punktow
39          case 5:
40              points[3] += 1
41
42
43  if __name__ == '__main__':
44      board = None
45      try:
46          input_file = sys.argv[1]
47          output_file = sys.argv[2]
48      except IndexError:
49          raise SystemExit("Nie podano scieżki do plików!")
50
51      board = loadBoard(input_file)
52      if board is not None:
53          used_symbols = [[[False for _ in range(4)] for _ in
    range(5)] for _ in range(5)]
54          kacper_points = [0 for _ in range(4)]
55          olek_points = [0 for _ in range(4)]
56          for i in range(5):
57              for j in range(5):
58                  symbol = board[i][j]
59
60                  # Wiersze
61                  n = 0
62                  while (j + n < 5) and (board[i][j + n] ==
    symbol) and (used_symbols[i][j + n][0] == False):
63                      used_symbols[i][j + n][0] = True
64                      n += 1
```

```
                if symbol == 'x':
                    addPoints(n, kacper_points)
                else:
                    addPoints(n, olek_points)

                # Kolumny
                n = 0
                while (i + n < 5) and (board[i + n][j] == symbol) and (used_symbols[i + n][j][1] == False):
                    used_symbols[i + n][j][1] = True
                    n += 1
                if symbol == 'x':
                    addPoints(n, kacper_points)
                else:
                    addPoints(n, olek_points)

                # Diagonale lewe
                n = 0
                while (i + n < 5) and (j - n >= 0) and (board[i + n][j - n] == symbol) and (used_symbols[i + n][j - n][2] == False):
                    used_symbols[i + n][j - n][2] = True
                    n += 1
                if symbol == 'x':
                    addPoints(n, kacper_points)
                else:
                    addPoints(n, olek_points)

                # Diagonale prawe
                n = 0
                while (i + n < 5) and (j + n < 5) and (board[i + n][j + n] == symbol) and (used_symbols[i + n][j + n][3] == False):
                    used_symbols[i + n][j + n][3] = True
                    n += 1
                if symbol == 'x':
                    addPoints(n, kacper_points)
                else:
                    addPoints(n, olek_points)

        # Sumowanie punktów
        sumKacper = kacper_points[3] * 15 + kacper_points[2] * 7 + kacper_points[1] * 3 + kacper_points[0] * 1
        sumOlek = olek_points[3] * 15 + olek_points[2] * 7 + olek_points[1] * 3 + olek_points[0] * 1
```

```
103
104         # Wypisanie wyników
105         try:
106             with open(output_file, "w", encoding='utf-8') as
    out_file:
107                 for row in board:
108                     print(*row, end=' ', file=out_file)
109                 print(f"\n{kacper_points[3]}*15 + {
    kacper_points[2]}*7 + {kacper_points[1]}*3 + {
    kacper_points[0]}*1 = {sumKacper}pkt", file=out_file)
110                 print(f"{olek_points[3]}*15 + {olek_points
    [2]}*7 + {olek_points[1]}*3 + {olek_points[0]}*1 = {
    sumOlek}pkt", file=out_file)
111                 if sumKacper > sumOlek:
112                     print("Wygrał Kacper", file=out_file)
113                 elif sumOlek > sumKacper:
114                     print("Wygrał Olek", file=out_file)
115                 else:
116                     print("Jest remis", file=out_file)
117             print(f"Zapisano wynik do pliku {output_file}")
118         except IOError:
119             print(f"Nie udało się zapisać do pliku {
    output_file}!")
```

3.3. Skrypt Python realizujący generowanie raportu

```
import sys
import os


# Tworzy tablice z planszą
def create_table(symbols: str) -> str:
    table = '''<table class="board"><tr>'''
    for j, symbol in enumerate(symbols):
        if j % 5 == 0 and j != 0:
            table += "</tr><tr>"
        if symbol == 'x':
            player = "kacper"
        else:
            player = "olek"
        table += f'''<td class="{player}">{symbol}</td>'''
    table += "</table>"
    return table


if __name__ == '__main__':
    try:
        input_dir = sys.argv[1]
        output_file = sys.argv[2]
    except IndexError:
        raise SystemExit("Nie podano scieżki do plików!")

    # Odczytywanie plików wejściowych i tworzenie raportów
    raport_sections = []
    try:
        for i, filename in enumerate(os.listdir(input_dir)):
            if filename.endswith(".txt"):
                file = os.path.join(input_dir, filename)
                with open(file, encoding="utf8") as raport:
                    lines = raport.read().split('\n')
                    board = lines[0].split()
                    score_kacper = lines[1]
                    score_olek = lines[2]
                    result = lines[3]

                    section_header = f'''
<hr><section><div class="left">
<h3>Raport nr.{i+1} <span style="font-weight: normal">(Plik {
    filename})</span></h3>'''
```

```
                    section_table = create_table(board)
                    section_text = f'''
</div><div class="right">
<div class="score"><span class="kacper">Punkty dla gracza umieszczającego "x"</span>:<br>{score_kacper}</div>
<div class="score"><span class="olek">Punkty dla gracza umieszczającego "o"</span>:<br>{score_olek}</div>
<div class="result">{result}</div></div>
</section>'''
                    raport_sections.append(section_header + section_table + section_text)
    except IOError:
        raise SystemExit("Błąd podczas odczytywania plików z raportami!")

    # Zapisywanie raportów do pliku .html
    try:
        with open(output_file, "w", encoding="utf=8") as html:
            html.write('''<!DOCTYPE html>
<html lang="pl">
<head>
<meta charset="UTF-8">
<title>Zadanie 2 2014 - Kółko i Krzyżyk</title>
<meta name="viewport" content="width=device-width,initial-scale=1">
<style>
body {
background-color: #ffffcc;
text-align: center;
font-size: 1.1em
}
header h1 {
font-weight: bold;
color: red
}
h3 {
font-size: 1.5em
}
hr {
border: none;
height: 2px;
background-color: black
}
section {
```

```
display: flex;
align-items: center;
justify-content: space-around
}
.right {
text-align: left
}
table {
font-size: 2em
}
th ,td {
text-align: center;
font-size: 2em;
border: 3px solid black;
width: 2em
}
.kacper {
color: blue
}
.olek {
color: green
}
.score {
text-align: center;
font-size: 1.5em
}
.score span {
font-weight: bold
}
.result {
text-align: center;
font-size: 1.5em;
font-weight: bold
}
</style>
</head>
<body>
<header>
<h1>Algorytmion 2014 Zadanie 2 - "Kółko i Krzyżyk"</h1>
<h2>Raport wykonanych obliczeń</h2>
</header>''')
            for section in raport_sections:
                html.write(section)
            html.write('''
</body>
```

```
127 </html>''')
128         print(f"Utworzono raport w pliku {output_file}")
129     except IOError:
130         raise SystemExit("Błąd podczas tworzenia raportu .
    html!")
```