

# Języki Skryptowe

Dokumentacja projektu "Ciągi bez zająknięć"

Robert Woźniak, grupa 3/6  
Politechnika Śląska  
Wydział Matematyki Stosowanej  
Informatyka

27 stycznia 2023

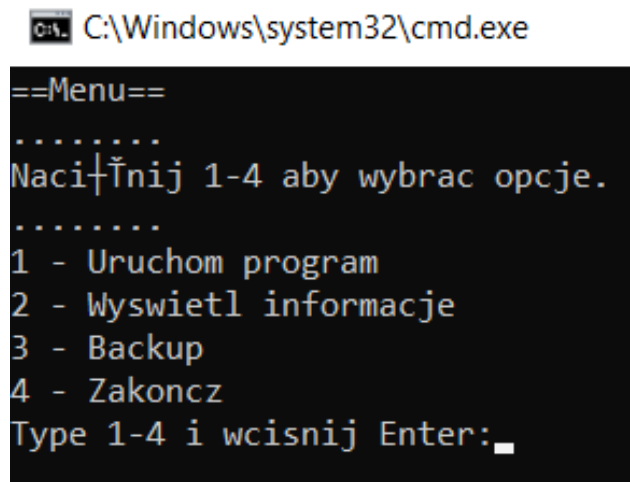
# Część I

## Opis programu

Program ma za zadanie znaleźć "ciąg bez zająknięć" (tj. ciąg, w którym żaden z podciągów się nie powtarza bezpośrednio po sobie o zadanej długości) oraz wskazać minimalną liczbę liter potrzebną do utworzenia takiego ciągu.

## Instrukcja obsługi

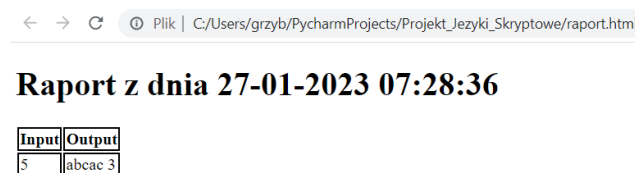
Aby uruchomić program należy włączyć skrypt menu.bat otwierający menu obsługi naszego programu. Po uruchomieniu wyświetli nam się tekst z instrukcją obsługi programu, wymagający podania przez użytkownika liczby w celu wykonania odpowiadającej mu funkcji. Możliwe wybory są następujące:



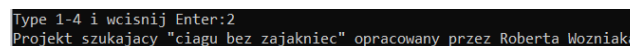
```
C:\Windows\system32\cmd.exe

==Menu==
.....
Naciśnij 1-4 aby wybrać opcje.
.....
1 - Uruchom program
2 - Wyświetl informacje
3 - Backup
4 - Zakończ
Type 1-4 i wcisnij Enter:
```

1. Uruchom program - Uruchamia program pobierając przy tym dane z katalogu input i tworzy raport.html, który jest następnie wyświetlany w domyślnej przeglądarce systemowej



2. Wyświetl informacje - Wypisuje na ekranie konsoli opis założeń programu



```
Type 1-4 i wcisnij Enter:2
Projekt szukający "ciągu bez zająknięć" opracowany przez Roberta Wozniaka
```

3. Backup - Tworzy kopię zapasową danych w katalogu backups zawierającą raport.html oraz zawartość folderów input i output

```
Type 1-4 i wciśnij Enter:3
-----
ROBOCOPY    ::    Robust File Copy for Windows
-----
Started : piątek, 27 stycznia 2023 07:36:33
Source   : C:\Users\grzyb\PycharmProjects\Projekt_Języki_Skryptowe\input\
Dest     : C:\Users\grzyb\PycharmProjects\Projekt_Języki_Skryptowe\backups\27.01.2023--7-36-33\input\
Files    : *.*
Options  : *.* /DCOPY:DA /COPY:DAT /R:1000000 /W:30
-----
100%      New Dir      1      C:\Users\grzyb\PycharmProjects\Projekt_Języki_Skryptowe\input\
New File      1      in.txt
-----
Total Copied Skipped Mismatch FAILED Extras
Dirs : 1 1 0 0 0 0
Files : 1 1 0 0 0 0
Bytes : 1 1 0 0 0 0
Times : 0:00:00 0:00:00 0:00:00 0:00:00 0:00:00
```

4. Zakoncz - Zamyka menu, kończąc tym samym program.

## Struktura danych programu

Program składa się z następującej struktury danych, wymaganych do prawidłowego uruchomienia aplikacji:

- menu.bat - Skrypt batch będący menu, którym uruchamia się program, wyświetla informacje o programie jak i tworzy kopie zapasową danych otrzymanych w wyniku wykonania tego programu
- main.py - Skrypt python zawierający główny program, pobierający plik wejściowy zawierający długość ciągu, który chcemy otrzymać i tworzący plik wyjścia zawierający wynik kalkulacji, który jest zgodny z założeniami projektu
- raport.py - Skrypt python pobierający dane z pliku wejścia oraz wyjścia i generujący plik raport.html zawierający raport wszystkich danych w postaci tabeli
- style.css - Plik kaskadowego arkusza stylu wykorzystywany podczas generowania raportu w celu stylizacji raportu, jak i ukazania danych w sposób właściwy
- Katalog input zawierający plik wejściowy in.txt Ponadto program w wyniku działania tworzy dodatkowo katalogi output, backups oraz plik raport.html, które nie są wymagane do prawidłowego uruchomienia aplikacji.

## Część II

### Opis działania

Do zrozumienia wykonywania programu potrzebna jest znajomość ciągu liczbowego, zwanego słowem Thue-Morse'a. Jest to binarny ciąg rekurencyjny zaczynający się od "0". Powstaje poprzez wprowadzenie alternatywy pomiędzy kolejnymi elementami kodu.

Do wykonania zadania wykorzystuje się pewną ciekawą własność słowa Thue-Morse'a. Ciąg, który otrzymamy możemy przekształcić zgodnie z tym ile "1" znajduje się pomiędzy "0". Jak się okazuje może to być tylko 0, 1 lub 2.

Przykład: dla 0110100110010110 jest to 2102012

Powstaje, zatem ciąg który nie ma, żadnych powtarzających się bez pośrednio po sobie podciągów. Oznacza to, że potrzebujemy maksymalnie 3 znaków by utworzyć taki ciąg. Problematyczna pozostaje kwestia ciągów o długości  $\leq 3$ , jednakże ze względu na to, że są to przypadki skrajne zostały uwzględnione jako specjalne przypadki.

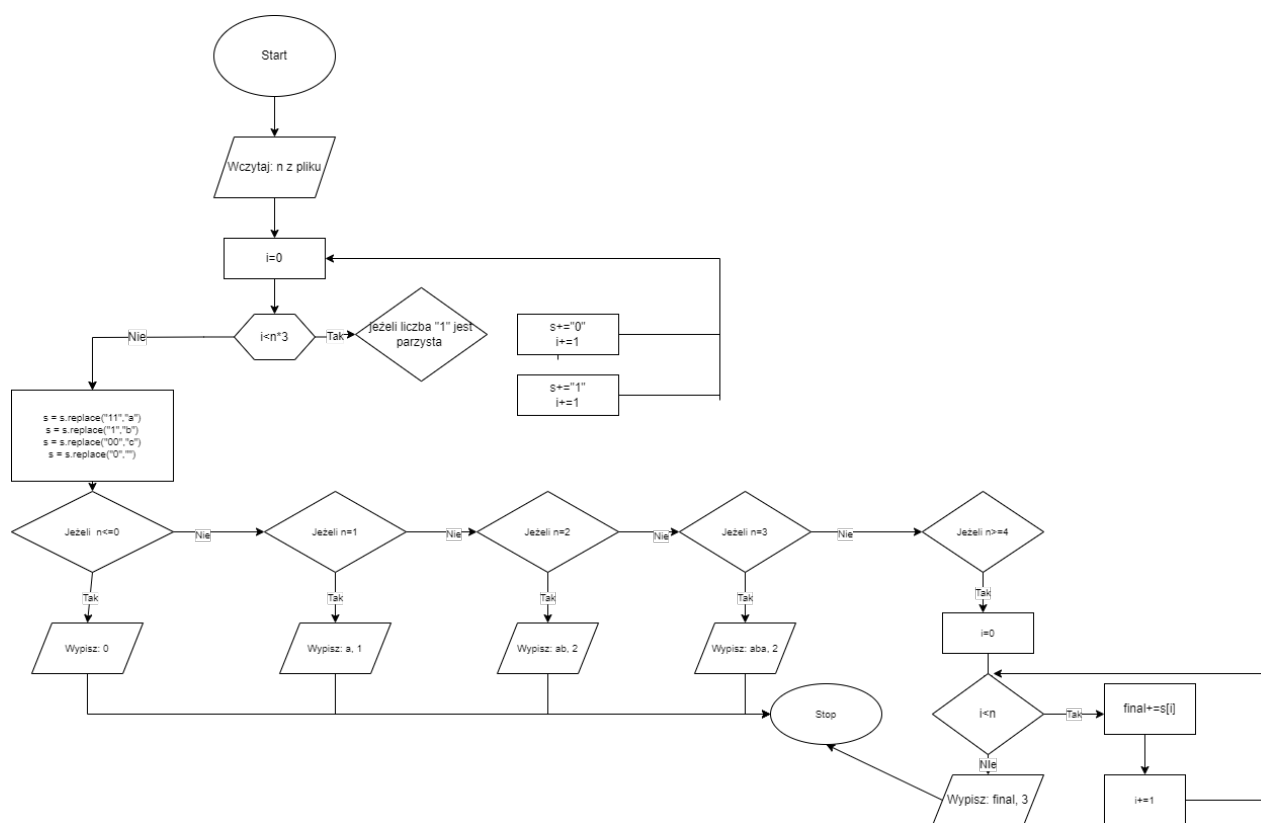
### Algorytmy

Słowo Thue-Morse'a generowane jest za pomocą pętli:

```
for i in range(int(n)*3):  
    if bin(i).count("1")%2==0:  
        s+="0"  
    else:  
        s+="1"
```

Tworzy ona bit po bicie, słowo Thue-Morse'a za pomocą sprawdzania czy liczba 1 w kolejnej liczbie zapisanej binarnie jest parzysta ("0"), nieparzysta ("1"), co tworzy negację kolejnych słów.

## Schemat blokowy



## Implementacja systemu

Skrypt menu.bat przy uruchomieniu, jak i po każdym odświeżeniu wypisuje instrukcje obsługi menu aplikacji. Uruchomienie programu z poziomu skryptu menu.bat powoduje uruchomienie main.py. Jeśli program znalazł wynik i zakończył się powodzeniem, tworzy on plik wynikowy do katalogu "output". Następnie uruchamiany jest skrypt raport.py, który pobiera dane odnośnie stylizowania strony z pliku "style.css" i tworzy plik raport.html w głównym katalogu projektu. Wybranie funkcji backup sprawdza czy istnieje w głównym katalogu plik raport.html, jeśli tak to tworzy katalog "backups" o ile takowy już nie istnieje, następnie tworzy katalog nazwany aktualną datą i godziną wewnątrz folderu "backups", który natomiast zawiera kopię całego katalogu "input", katalogu "output" oraz pliku raport.html.

Wykorzystane biblioteki i klasy oraz przykłady ich użycia

- os - ta biblioteka zawiera funkcje do pracy z systemem operacyjnym, takie jak usuwanie pliku (os.remove()) czy sprawdzanie, czy plik istnieje (exists()).]
- datetime - ta biblioteka zawiera klasy do pracy z datami i czasem, takie jak date i datetime. W tym programie służy do pobrania aktualnej daty i czasu.
- HTMLMakePrint - ta klasa zawiera metody do tworzenia i zapisywania pliku HTML. W konstruktorze tworzy plik HTML i zapisuje do niego nagłówki i styl CSS, a także tabelę z inputem i outputem.

Funkcje zawarte w raport.py

Klasa HTMLMakePrint zawiera tylko jedną funkcję - konstruktor (`__init__(self)`), który jest wywoływany automatycznie przy tworzeniu obiektu tej klasy.

Konstruktor (`__init__(self)`) pełni kilka ról:

- Sprawdza, czy plik "raport.html" istnieje i jeśli tak, to go usuwa.
- Pobiera aktualną datę i czas oraz zapisuje je w zmiennej self.fulldate.
- Otwiera plik "res/style.css" i zapisuje zawartość w zmiennej self.css.
- Tworzy plik "raport.html" i zapisuje do niego nagłówki HTML oraz styl CSS.
- Otwiera tabelę HTML i nagłówki tabeli.

Funkcja write() jest wbudowana w pythonie i jest to funkcja która pozwala na zapis do pliku, w tym przypadku pliku HTML, który został otwarty w konstruktorze.

## Testy

Przykład standardowy:

Input	Output
10	abcbabcb 3

Przykłady skrajne:

Input	Output	Input	Output
0	0	3	aba 2

## **Złożoność obliczeniowa**

Ten kod ma złożoność obliczeniową  $O(n)$ , gdzie  $n$  jest długością ciągu Thue-Morse. Pierwsza pętla `for` jest używana do iterowania przez wartości od 0 do  $(n*3) - 1$  i generowanie ciągu Thue-Morse.

# Pełen kod aplikacji

## main.py

```
1 import os.path
2
3 cd = os.path.dirname(os.path.abspath(r'C:\Users\grzyb\PycharmProjects\
    Projekt_Języki_Skryptowe\output\out.txt'))
4 open(os.path.join(cd,r'C:\Users\grzyb\PycharmProjects\
    Projekt_Języki_Skryptowe\output\out.txt'))
5 with open(r'C:\Users\grzyb\PycharmProjects\Projekt_Języki_Skryptowe\
    input\in.txt','r') as file:
6     n = file.read()
7     s = ""
8     final=""
9     for i in range(int(n)*3):
10         if bin(i).count(("1"))%2==0:
11             s+="0"
12         else:
13             s+="1"
14     s = s.replace("11","a")
15     s = s.replace("1","b")
16     s = s.replace("00","c")
17     s = s.replace("0","")
18     if(int(n)<=0):
19         print(0)
20         outputfile = open(r'C:\Users\grzyb\PycharmProjects\
            Projekt_Języki_Skryptowe\output\out.txt','w")
21         outputfile.write(str(0))
22     if(int(n)==1):
23         print("a")
24         print(1)
25         outputfile = open(r'C:\Users\grzyb\PycharmProjects\
            Projekt_Języki_Skryptowe\output\out.txt','w")
26         outputfile.write("a " + str(1))
27     if(int(n)==2):
28         print("ab")
29         print(2)
30         outputfile = open(r'C:\Users\grzyb\PycharmProjects\
            Projekt_Języki_Skryptowe\output\out.txt','w")
31         outputfile.write("ab " + str(2))
32     if(int(n)==3):
33         print("aba")
34         print(2)
35         outputfile = open(r'C:\Users\grzyb\PycharmProjects\
            Projekt_Języki_Skryptowe\output\out.txt','w")
36         outputfile.write("aba " + str(2))
37     if(int(n)>=4):
38         for i in range(int(n)):
39             final+=s[i]
40         print(final)
41         print(3)
42         outputfile = open(r'C:\Users\grzyb\PycharmProjects\
            Projekt_Języki_Skryptowe\output\out.txt','w")
```



```
43      outputfile.write(f"{final} " + str(3)).
```

---

## raport.py

```
1  import os
2  from datetime import datetime
3  from os.path import exists
4
5  class HTMLCreator:
6      def __init__(self):
7          if exists("raport.html"):
8              os.remove("raport.html")
9
10         now = datetime.now()
11         self.fulldate = now.strftime("%d-%m-%Y %H:%M:%S")
12         self.css = open("style.css", "r")
13         self.html = open("raport.html", "w")
14         self.html.write(f"""<!DOCTYPE html>
15             <html>
16             <head>
17                 <title>Raport z dnia {self.fulldate}</title>
18                 <style>
19                     {self.css.read()}
20                 </style>
21             </head>
22             <body>
23                 <div class="container">
24                     <h1>Raport z dnia {self.fulldate}</h1>
25                     <table>
26                         <tr>
27                             <th>Input</th>
28                             <th>Output</th>
29                         </tr>\n""")
30
31     raport = HTMLCreator()
32
33     raport.html.write("<tr>")
34     inputfile = open(r'C:\Users\grzyb\PycharmProjects\
35         Projekt_Języki_Skryptowe\input\in.txt', "r")
36     raport.html.write(f"<td>{inputfile.read()}</td>\n")
37
38     outputfile = open(r'C:\Users\grzyb\PycharmProjects\
39         Projekt_Języki_Skryptowe\output\out.txt', "r")
40     raport.html.write(f"<td>{outputfile.read()}</td>\n")
41
42     raport.html.write("</tr>\n")
43
44     raport.html.write("</table>")
45     raport.html.write("</div>")
46     raport.html.write("</body>")
47     raport.html.write("</html>")
48     raport.html.close()
```

---

## menu.bat

```
1 @echo off
2 cls
3 :menu
4 echo ==Menu==
5 echo .....
6 echo Nacisnij 1-4 aby wybrac opcje.
7 echo .....
8
9 echo 1 - Uruchom program
10 echo 2 - Wyszwietl informacje
11 echo 3 - Backup
12 echo 4 - Zakoncz
13
14 SET /p M=Type 1-4 i wcisnij Enter:
15
16 IF %M%==1 goto uruchom
17 IF %M%==2 goto info
18 IF %M%==3 goto backup
19 IF %M%==4 goto wyjscie
20
21 :uruchom
22
23 IF NOT EXIST output mkdir output
24 py main.py
25 py raport.py
26 start raport.html
27 echo
28
29
30 :info
31 echo Projekt szukajacy "ciagu bez zajakniec" opracowany przez Roberta
    Wozniaka
32 goto :menu
33
34 :backup
35 IF NOT EXIST backups mkdir backups
36 set name=%date%--%TIME:~1,7%
37 set name=%name:~1,7%
38 IF EXIST raport.html mkdir backups\%name%
39 robocopy input backups\%name%\input
40 robocopy output backups\%name%\output
41 copy raport.html backups\%name%\raport.html
42 goto :menu
43
44 :wyjscie
45 exit
46 PAUSE
```

---