Języki Skryptowe

dokumentacja projektu 'Niepowtarzalne zadanie'

Michał Bober, grupa 3/5

10 stycznia 2023

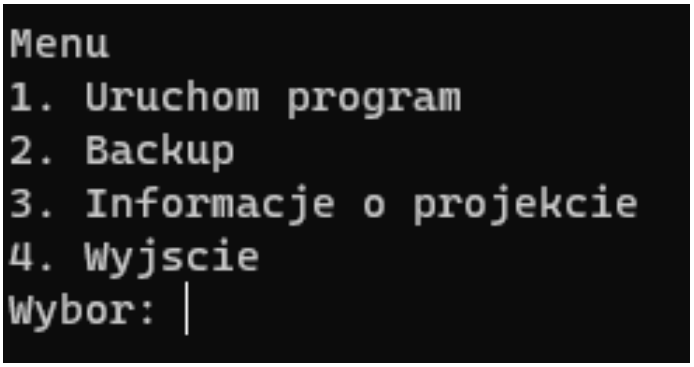
Część I

Opis programu

Projekt na podstawie zadania nr 2 z konkursu 'Algorytmion' z roku 2020.
Opis Algorytmu
Ciągiem niepowtarzalnym nazywamy taki ciąg (rozpatrujemy w tym zadaniu tylko ciągi skończone, tzn. posiadające skończoną liczbę elementów), w którym dowolny jego podciąg składający się z dowolnej liczby jego kolejnych elementów nie powtarza się bezpośrednio po sobie. Np. ciąg 1,2,3,2,3,1 nie jest ciągiem niepowtarzalnym, bo podciąg 2,3 występuje bezpośrednio po sobie. Podobnie ciąg 2,3,4,3,3,2,4 nie jest ciągiem niepowtarzalnym, bo podciąg 3 występuje bezpośrednio po sobie. Natomiast ciąg 1,2,3,2,1,3 jest ciągiem niepowtarzalnym, podobnie jak ciąg 3,2,4,3,2,1 (podciąg 3,2 występuje w tym ciągu dwa razy, jednak nie jest to bezpośrednie sąsiedztwo, bo rozdziela je element 4).

Instrukcja obsługi

Aby uruchomić program należy włączyć skrypt menu.bat otwierający menu obsługi naszego programu. Po uruchomieniu wyświetli nam się menu programu, wymagające podania przez użytkownika liczby w celu wykonania odpowiadającej mu funkcji.



Rysunek 1: Główne menu programu

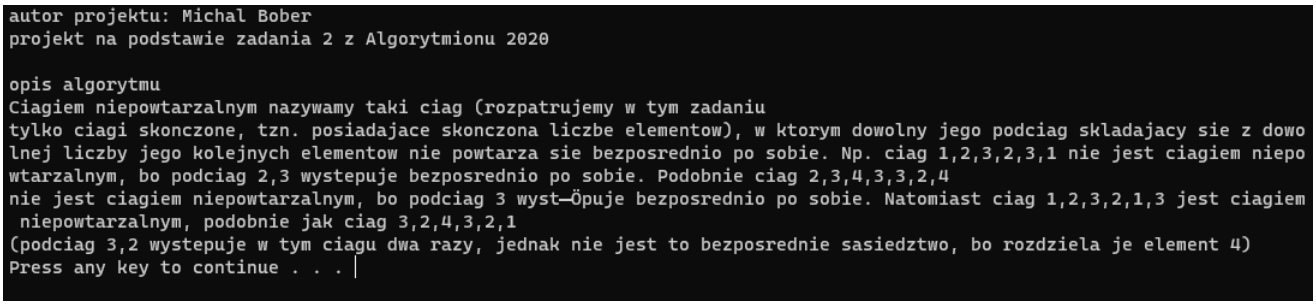
Możliwe wybory są następujące:

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

inputs	outputs
123231	nie jest niepowtarzalny 23
2343324	nie jest niepowtarzalny 3
123213	jest niepowtarzalny
324321	jest niepowtarzalny

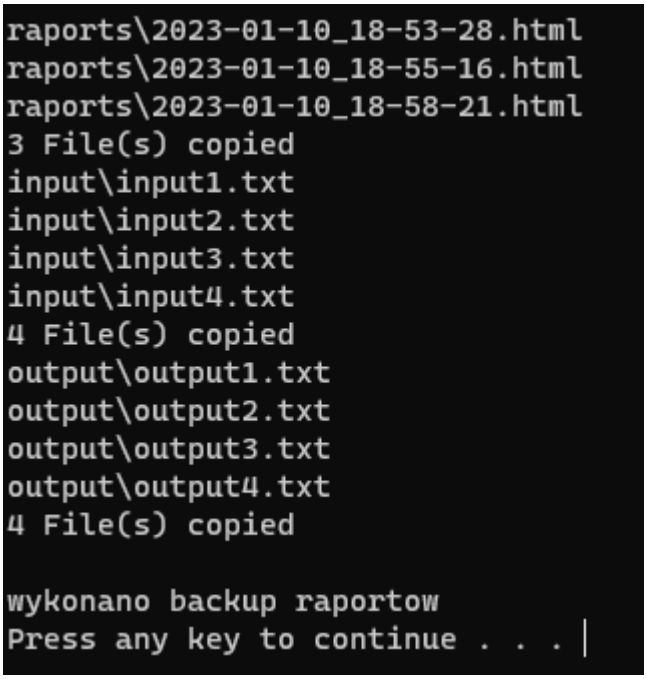
Rysunek 2: Przykładowy raport

2. Wyswietl informacje - Wypisuje na ekranie konsoli opis algorytmu



Rysunek 3: Informacje o programie

3. Backup - Tworzy kopię zapasową danych w katalogu backups raportow html



Rysunek 4: Komunikat o dokonanym backupie

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

Struktura danych programu

Program składa się z następującej struktury danych:
- menu.bat - Skrypt batch będący menu, którym uruchamia się program, może uruchomić skrypt python, wyświetlić informacje o projekcie i wykonać backup danych.
- main.py - Skrypt python zawierający główny program, i generator raportu html
- Katalog input zawierający wszystkie dane wejściowe do programu:
(można dodać własny plik o nazwie z kolejnym numerem gdzie podamy własne dane do sprawdzenia algorytmu)

$$input0.txt, input1.txt...inputX.txt \tag{1}$$

(tworzone w trakcie działania programu)
- Katalog output zawierający dane wyjściowe programu:
- Katalog raports zawierajacy wszystkie wygenerowane raporty:

$$Rok - miesiac - dzien - godzina.html \tag{2}$$

- Katalog backup zawierający kopię folderow: input, output, raports:

Część II

Opis działania

Skrypt menu.bat pobiera kolejno wszystkie nazwy plików wejściowych z katalogu input i przekazuje je wszystkie jako string do skryptu python który tworzy na podstawie tego listę z nazwami wszystkich plików. Następnie skrypt python wykonuje algorytm na każdym z danych z plików, który sprawdza czy input z danego pliku jest "ciągiem niepowtarzalnym".(Wytłumaczenie tego pojęcia w opisie projektu). Jeżeli tak to do kolejnych plików *output.txt* przekazywane jest 'jest niepowtarzalny' w przeciwnym przypadku 'nie jest niepowtarzalny x' gdzie x to ciąg który sie powtórzył.

Następnie na podstawie wszystkich plików input i output tworzony jest raport.html w któртm znajduje się tabelka z inputem z pliku i outputem na jego podstawie

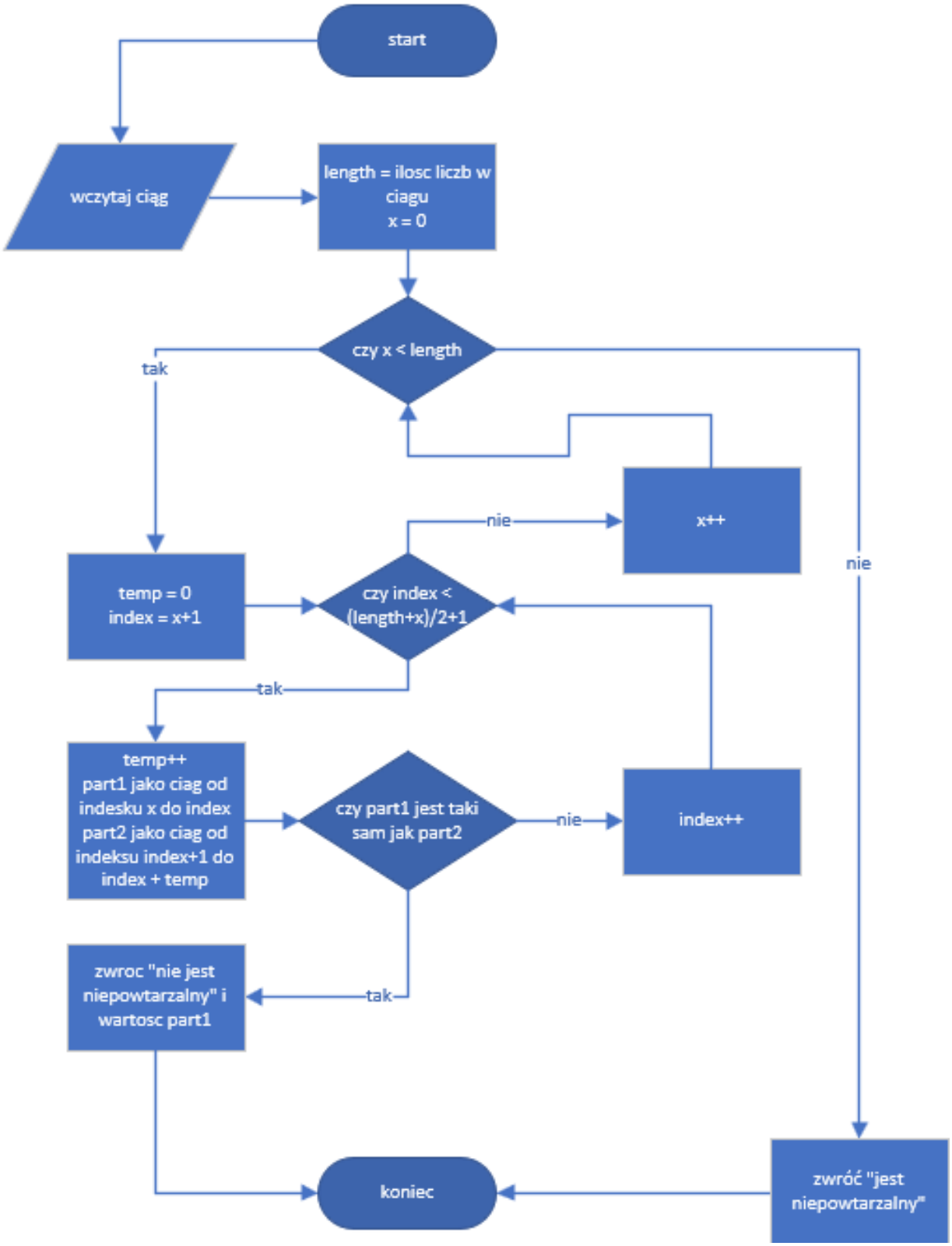
Ostatecznie uruchamiana jest domyślna przeglądarka użytkownika, w której wyświetla się utworzony *raport.html*

Algorytm

Data: Dane wejściowe plik *input*
Result: Dane wyjściowe plik *output*
Do *ciag* przypisz wartość *input*
Do *len* przypisz długość *ciag*

while $x < len$ **do**
  $temp = 0$
  $index = x+1$
  while $index < (len+x/2)+1$ **do**
    $temp{+}{+}$
    *part*1 to *ciag* od indexu x do indexu *index*
    *part*2 to *ciag* od indexu *index* do indexu *index*+*temp*
    if *part1* == *part2* **then**
      zwróć jako output 'nie jest niepowtarzalny *part*1
    end
  end
end
if *nic nie zostalo zwrocone* **then**
  zwróc jako output 'jest niepowtarzalny'
end

Algorithm 1: Algorytm sprawdzania czy ciąg z inputu jest 'ciągiem niepowtarzalnym'



Rysunek 5: Schemat blokowy algorytmu

Implementacja systemu

Wykorzystane biblioteki i przykłady ich użycia

```
import sys
from webbrowser import open as webopen
from os import path
from datetime import datetime
```

- webbrowser.open() as webopen()

```
webopen(file_path)
# otwiera w przegladarce plik o sciezce file_path
```

- os.path()

```
path.abspath(f"{file_name}.html")
#pobiera absolutna sciezke do danego pliku(pozniej wykorzystywane do
     otwierania w przegladarce pliku)
```

- sys

```
files = "".join(sys.argv[1:])[:-1].split(",")
#Pobiera argument przekazany do skryptu python z nazwami plikow i
    tworzy z niego liste wszystkich nazw plikow input
```

- datetime

```
file_name = datetime.now().strftime("%Y-%m-%d_%H-%M-%S")
#zwraca aktualna date z godzina w formacie prawidlowym do uzycia go
    jako nazwa pliku
```

Funkcje zawarte w main.py

- funkcja algorytm

```
# algorytm zwracajacy niepowtarzalnosc ciagu
def algorytm(ciag):
    # dlugosc ciagu
    length = len(ciag)
    # petla wykonujaca sie length razy
    for x in range(length):
        temp = 0
        # dla kazdej cyfry przejscie po wszystkich mozliwych ciagach laczonych
        for index in range(x+1,(length+x)//2+1):
            temp += 1
            part1 = ciag[x:index]
            part2 = ciag[index:index+temp]
            if part1 == part2:
                # jezeli wystepuja ciagi takie same obok siebie zwraca ze ciag jest powtarzalny i jakie liczby sa za to odpowiedzialne
                return f"nie jest niepowtarzalny {part1}"

    # w przeciwnym przypadku zwraca ze ciag jest niepowtarzalny
    return "jest niepowtarzalny"
```

- funkcja read file

```
# funkcja zczytujaca dane z pliku
def read_file(f_name):
    with open(f_name) as file:
        #zwraca dane z pliku
        return file.readlines()
```

- funkcja make html

```
# funkcja tworzaca raport html i otwierajaca go w oknie przegladarki
def make_html(inputs,outputs):
    # zmienne ze stringami zawierajace kod html
    table = "<table bgcolor='black'>\n"
    header = ["inputs","outputs"]
    table += "  <tr bgcolor='grey'>\n"
    # tworzenie kolumn naglowkowych
    for column in header:
        table += "    <th>{0}</th>\n".format(column.strip())
    table += "  </tr>\n"
    data = [[inputs[i],outputs[i]] for i in range(len(inputs))]
    # dla kazdego outputu
    for line in data:
        table += "  <tr bgcolor='grey'>\n"
        for column in line:
            table += "    <td>{0}</td>\n".format(column.strip())
        table += "  </tr>\n"
    table += "</table>"
    file_name = datetime.now().strftime("%Y-%m-%d_%H-%M-%S")
    with open(f"raports/{file_name}.html", "w") as file:
        file.writelines(table)
    # otwiera utworzony raport w domyslnej przegladarce
    webopen(path.abspath(f"raports/{file_name}.html"))
```

- funkcja generate io

```
# funkcja generujaca zczytujaca pliki input i wywolujaca nich
    algorytm oraz generujaca pliki output.txt
def generate_io():
    # zczytanie nazw plikow przekazanych jako argumenty
    files = "".join(sys.argv[1:])[:-1].split(",")
    inputs = []
    outputs = []
    # przejscie po wszystkich plikach i wykonanie na nich algorytmu
       i zapisanie do kolejno utworzonych plikow output
    for i,file in enumerate(files,1):
        ciag = read_file(f"input/{file}")[0]
        inputs.append(ciag)
        with open(f"output/output{i}.txt","w") as file:
            file.write(algorytm(ciag))
            outputs.append(algorytm(ciag))
    # zwraca pliki input i output na podstawie ktorych potem
       tworzony jest raport
    return [inputs, outputs]
```

Pełen kod aplikacji

main.py

```
1  import sys
2  from webbrowser import open as webopen
3  from os import path
4  from datetime import datetime
5
6  # algorytm zwracajacy niepowtarzalnosc ciagu
7  def algorytm(ciag):
8      length = len(ciag)
9      for x in range(length):
10         temp = 0
11         for index in range(x+1,(length+x)//2+1):
12             temp += 1
13             part1 = ciag[x:index]
14             part2 = ciag[index:index+temp]
15             if part1 == part2:
16                 return f"nie jest niepowtarzalny {part1}"
17
18     return "jest niepowtarzalny"
19 # funkcja zczytujaca dane z pliku
20 def read_file(f_name):
21     with open(f_name) as file:
22         return file.readlines()
23
24 # funkcja tworzaca raport html i otwierajaca go w oknie przegladarki
25 def make_html(inputs,outputs):
26     table = "<table bgcolor='black'>\n"
27     header = ["inputs","outputs"]
28     table += "  <tr bgcolor='grey'>\n"
29     for column in header:
30         table += "    <th>{0}</th>\n".format(column.strip())
31     table += "  </tr>\n"
32     data = [[inputs[i],outputs[i]] for i in range(len(inputs))]
33     for line in data:
34         table += "  <tr bgcolor='grey'>\n"
35         for column in line:
36             table += "    <td>{0}</td>\n".format(column.strip())
37         table += "  </tr>\n"
38     table += "</table>"
39     file_name = datetime.now().strftime("%Y-%m-%d_%H-%M-%S")
40     with open(f"{file_name}.html", "w") as file:
41         file.writelines(table)
42     webopen(path.abspath(f"{file_name}.html"))
43
44 # funkcja generujaca zczytujaca pliki input i wywolujaca nich algorytm
       oraz generujaca pliki output.txt
45 def generate_io():
46     files = "".join(sys.argv[1:])[:-1].split(",")
47     inputs = []
48     outputs = []
49     for i,file in enumerate(files,1):
```

```
50          ciag = read_file(f"input/{file}")[0]
51          inputs.append(ciag)
52          with open(f"output/output{i}.txt","w") as file:
53              file.write(algorytm(ciag))
54              outputs.append(algorytm(ciag))
55      return [inputs, outputs]
56
57
58 data = generate_io()
59 make_html(data[0],data[1])
```

---

menu.bat

```
1  @echo off
2  setlocal enabledelayedexpansion
3  :menu
4  cls
5  echo Menu
6  echo 1. Uruchom program
7  echo 2. Backup
8  echo 3. Informacje o projekcie
9  echo 4. Wyjscie
10 set /p select=Wybor:
11 IF %select%==1 GOTO run
12 IF %select%==2 GOTO backup
13 IF %select%==3 goto info
14 IF %select%==4 goto exit
15 goto menu
16
17 :run
18 cls
19 if not exist raports mkdir raports
20 if exist output (
21     rmdir /s /q output
22 )
23 mkdir output
24 set result=
25 for %%i in (input/) do (
26     set result=!result!%%i,
27 )
28 python main.py %result%
29 echo utworzono raport
30 pause
31 goto menu
32
33 :backup
34 cls
35 if exist backup (
36     rmdir /s /q backup
37 )
38 mkdir backup
39 xcopy raports backup\raports /e /i
40 xcopy input backup\input /e /i
```

```
41 xcopy output backup\output /e /i
42
43 echo:
44 echo wykonano backup raportow
45 pause
46 goto menu
47
48 :info
49 cls
50 echo autor projektu: Michal Bober
51 echo projekt na podstawie zadania 2 z Algorytmionu 2020
52 echo:
53 echo opis algorytmu
54 echo Ciagiem niepowtarzalnym nazywamy taki ciag (rozpatrujemy w tym
       zadaniu
55 echo tylko ciagi skonczone, tzn. posiadajace skonczona liczbe elementow)
       , w ktorym dowolny jego podciag skladajacy sie z dowolnej liczby jego
        kolejnych elementow nie powtarza sie bezposrednio po sobie. Np. ciag
        1,2,3,2,3,1 nie jest ciagiem niepowtarzalnym, bo podciag 2,3
       wystepuje bezposrednio po sobie. Podobnie ciag 2,3,4,3,3,2,4
56 echo nie jest ciagiem niepowtarzalnym, bo podciag 3 wystepuje
       bezposrednio po sobie. Natomiast ciag 1,2,3,2,1,3 jest ciagiem
       niepowtarzalnym, podobnie jak ciag 3,2,4,3,2,1
57 echo (podciag 3,2 wystepuje w tym ciagu dwa razy, jednak nie jest to
       bezposrednie sasiedztwo, bo rozdziela je element 4)
58 pause
59 goto menu
60
61 :exit
62 echo Koniec
63 pause
64 cls
```