

Języki Skryptowe

dokumentacja projektu
Zadanie 1 - "Kombinacje"
Konkurs Algorytmion 2013

Mariusz Wróbel, grupa 4/8

6 grudnia 2021

Spis treści

1	Część I	3
1.1	Opis programu	3
1.2	Instrukcja obsługi	3
1.3	Dodatkowe informacje	4
2	Część II	5
2.1	Opis działania	5
2.2	Schemat działania rekurencji	5
2.3	Algorytm wyszukiwania kombinacji	6
2.4	Implementacja	6
2.5	Test	7
3	Podsumowanie	9
3.1	Możliwości rozwijania programu	9
3.2	Pełen kod aplikacji	9

1.3 Dodatkowe informacje

- Program zaprzestanie wykonywania szukania jeżeli plik appIn.txt nie istnieje.
- Jeżeli argument nie będzie liczbą lub nie będzie liczbą naturalną program pominie analize owego argumentu.
- W tablicy końcowej argumenty wejściowe zostają wpisane tak jak były podane w pliku appIn.txt nie tak jak były analizowane przez program.
- Wersja pythona: 3.7.4

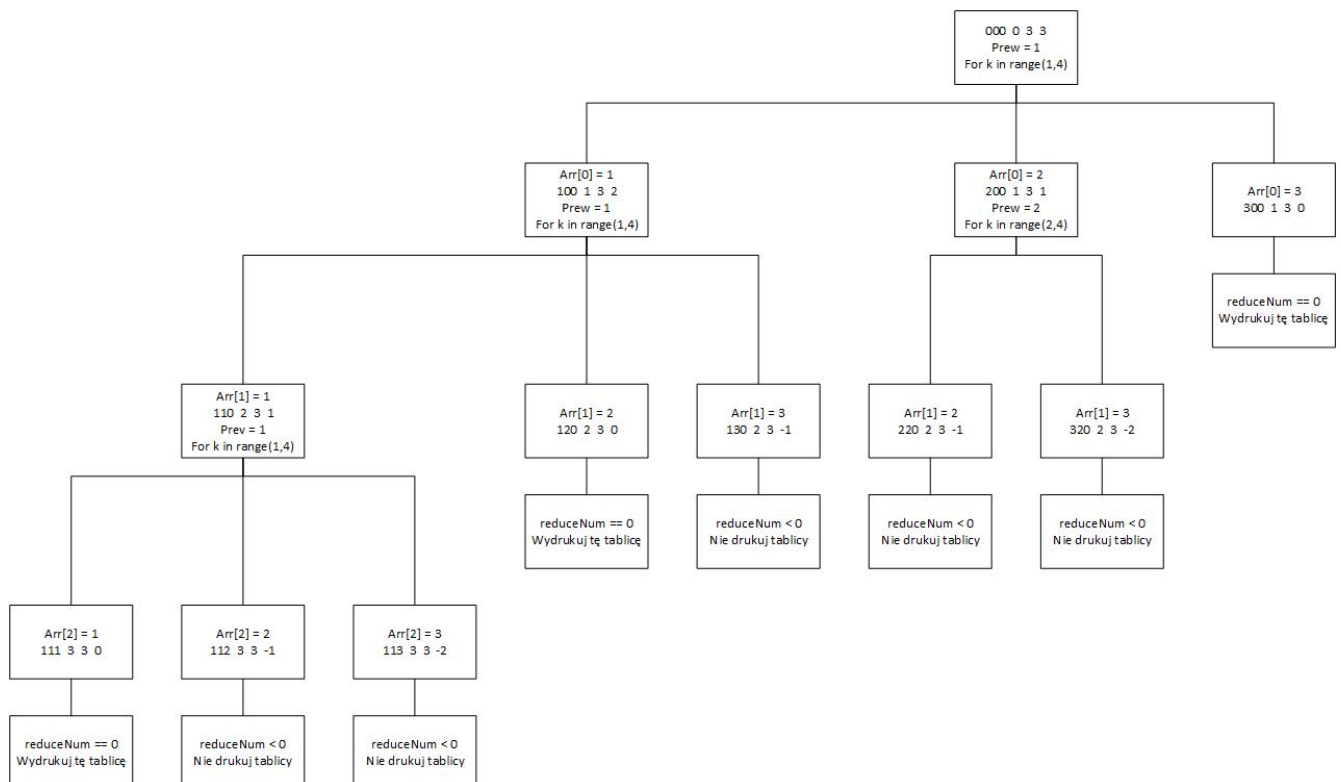
2 Część II

2.1 Opis działania

Jako pierwsze program tworzy tablicę zer która ma tyle elementów ile wynosi analizowana liczba. Następnie po raz pierwszy zostaje wywołana funkcja szukająca kombinacji jako argumenty przesyłamy naszą tablicę ,index który teraz wynosi 0, liczbę do której szukamy kombinacji oraz liczbę pomniejszoną, która teraz wynosi tyle samo. Jako pierwszą rzecz funkcja sprawdza czy liczba pomniejszona jest mniejsza od 0. Jeżeli jest to prawda to funkcja zostaje przerwana ponieważ suma liczb w tablicy jest większa od szukanej liczby. Następnie zostaje sprawdzone czy liczba pomniejszona jest równa 0. Jeżeli jest to prawda to zostaje zwrócona tablica ponieważ oznacza to ,że kombinacja została odnaleziona. Tablica zostaje wyświetlona bez zerowych elementów z uwagi na użycie zmiennej index która informuje który element był ostatnio zmieniany. Jeżeli dwa poprzednie warunki nie zostały spełnione, program zapisuje ostatni element. Za pomocą funkcji for wywołujemy ponownie funkcję szukającą kombinacji z tą różnicą, że tablica zawiera poprzednie kombinacje elementów, index zostaje zwiększony o 1, a od liczby pomniejszonej zostaje odjęty wartość ostatniego elementu.

2.2 Schemat działania rekurencji

Zobrazowany poniżej przykład działania rekurencji pokazuje proces szukania kombinacji dla liczby $n = 3$.



Rysunek 2: Schemat działania rekurencji dla $n = 3$

2.3 Algorytm wyszukiwania kombinacji

Algorithm 1 Algorytm wyszukiwania kombinacji.

Data: Dane wejściowe liczba n

Result: Wypisane kombinacje

def szukajKombinacji(arr, index, num, reducedNum)

if $reducedNum < 0$ **then**

 Kombinacja jest nieprawidłowa.

 Return

end

if $reducedNum == 0$ **then**

 Kombinacja została odnaleziona, wypisz tablicę do pliku łącznie z licznikiem

 Return

end

if $index == 0$ **then**

 prev = 1

else

 prev = arr[index - 1]

end

i = prev

for $i < num + 1$ **do**

 arr[index] = i

 szukajKombinacji(arr, index + 1, num, reducedNum - i)

 i += 1

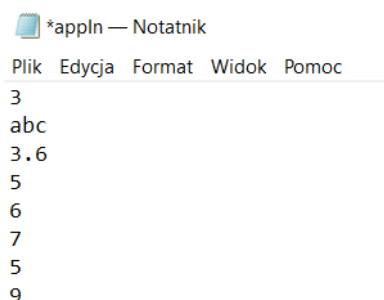
end

2.4 Implementacja

Program rozpoczynamy poprzez otwarcie aplikacji skrypt.bat. Aby uruchomić algorytm szukający należy wybrać odpowiednią opcję. Po wybraniu odpowiedniej opcji zostaje uruchomiony skrypt app.py który pobiera dane z pliku appIn.txt i je przetwarza. Po przetworzeniu efekt działania skryptu zostaje zapisany w pliku appOut.txt. Następnie zostaje uruchomiony skrypt web.py który pobiera dane z pliku appOut.txt i tworzy kod HTML który zostaje otwarty w przeglądarce internetowej.

2.5 Test

W pliku z danymi wejściowymi dodają liczby naturalne jak i również litery oraz liczby zmienneoprzecinkowe do przetestowania zabezpieczeń.



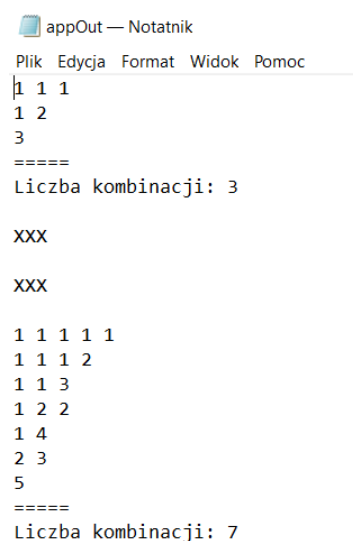
Rysunek 3: Zawartość pliku appIn.txt

Po otwarciu pliku skrypt.bat w menu wybieram opcję pierwszą która uruchamia skrypt app.py oraz web.py.



Rysunek 4: Widok menu z pliku skrypt.bat

Efekt działania skryptu app.py w surowej formie możemy zobaczyć w pliku appOut.txt



```

appOut — Notatnik
Plik  Edycja  Format  Widok  Pomoc
1 1 1
1 2
3
=====
Liczba kombinacji: 3

XXX

XXX

1 1 1 1 1
1 1 1 2
1 1 3
1 2 2
1 4
2 3
5
=====
Liczba kombinacji: 7

```

Rysunek 5: Zawartość pliku appOut.txt po uruchomieniu skryptu app.py

Po zapisaniu danych wyjściowych w pliku appOut.txt za pomocą skryptu web.py tworzymy kod HTML który wyświetla dane wyjściowe w postaci tabeli na stronie internetowej.

n = 3	1 1 1 1 2 3 ===== Liczba kombinacji: 3
n = abc	XXX
n = 3.6	XXX
n = 5	1 1 1 1 1 1 1 1 2 1 1 3 1 2 2 1 4 2 3 5 ===== Liczba kombinacji: 7

Rysunek 6: Efekt końcowy programu

Jak możemy zaobserwować, program poprawnie wypisał wszystkie kombinacje liczb naturalnych które sumują się do podanej liczby jednocześnie pominał analizę błędnych danych.

3 Podsumowanie

W projekcie udało się wykonać wszystkie założenia. Program wykonuje swoje zadanie, jest zabezpieczony przed brakiem poszczególnych plików lub błędnie wprowadzonbych danych. Efekt algorytmu szukającego zostaje poprawnie wyświetlony na stronie internetowej.

3.1 Możliwości rozwijania programu

Jedną z możliwości rozwinięcia programu jest dodanie interfejsu graficznego który zastąpiłby plik skrypt.bat lub polepszenie wyglądu strony internetowej.

3.2 Pełen kod aplikacji

app.py

```
1 c = 0
2
3 def szukajKombinacji(arr, index, num, reducedNum):
4     global c
5     if reducedNum < 0:
6         return
7
8     if reducedNum == 0:
9
10        for i in range(index):
11            file_write.write(str(arr[i]))
12            file_write.write(" ")
13            file_write.write('\n')
14            c += 1
15            return
16
17    if index == 0:
18        prev = 1
19    else:
20        prev = arr[index - 1]
21
22    for k in range(prev, num + 1):
23        arr[index] = k
24
25        szukajKombinacji(arr, index + 1, num, reducedNum - k)
26
27
28 def algorytmSzukajacy(n):
29     arr = [0] * n
30
31     szukajKombinacji(arr, 0, n, n)
32
33
34 try:
35     file_read = open("appIn.txt", 'r')
36
37     file_write = open("appOut.txt", 'w+')
38
```

```

39     for x in file_read:
40         try:
41             algorytmSzukajacy(int(x))
42             file_write.write("====")
43             file_write.write('\n')
44             file_write.write(f'Liczba kombinacji: {c}')
45             file_write.write('\n')
46             file_write.write('\n')
47             c = 0
48
49         except ValueError:
50             print("Bład danych wejsciowych!")
51             print("Bładny argument zostal pominiety.")
52             file_write.write("XXX")
53             file_write.write('\n')
54             file_write.write('\n')
55
56     file_read.close()
57     file_write.close()
58
59     print("Skrypt app.py wykonal sie poprawnie.")
60
61 except IOError:
62     print("Bład otwierania pliku appIn.txt")

```

web.py

```

1 def szukajKombinacji(arr, index, num, reducedNum):
2     if reducedNum < 0:
3         return
4
5     if reducedNum == 0:
6
7         for i in range(index):
8             file_write.write(str(arr[i]))
9             file_write.write(" ")
10            file_write.write('\n')
11            return
12
13     if index == 0:
14         prev = 1
15     else:
16         prev = arr[index - 1]
17
18     for k in range(prev, num + 1):
19         arr[index] = k
20
21         szukajKombinacji(arr, index + 1, num, reducedNum - k)
22
23
24 def algorytmSzukajacy(n):
25     arr = [0] * n
26
27     szukajKombinacji(arr, 0, n, n)
28

```

```

29
30 try:
31     file_read = open("appIn.txt", 'r')
32
33     file_write = open("appOut.txt", 'w')
34
35     for x in file_read:
36         try:
37             algorytmSzukajacy(int(x))
38             file_write.write('\n')
39
40         except ValueError:
41             print("Bład danych wejsciowych!")
42             print("Bledny argument zostal pominiety.")
43             file_write.write("---")
44             file_write.write('\n')
45             file_write.write('\n')
46
47     file_read.close()
48     file_write.close()
49
50     print("Skrypt app.py wykonal sie poprawnie.")
51
52 except IOError:
53     print("Bład otwierania pliku appIn.txt")

```

skrypt.bat

```

1 @echo off
2 setlocal enabledelayedexpansion
3
4 :menu
5 echo.=====
6 echo.    1. Start
7 echo.    2. Informacje
8 echo.    3. Backup
9 echo.    4. Zakoncz
10 echo.=====
11 set /p input=Wybierz opcje:
12 if %input% EQU 1 goto start
13 if %input% EQU 2 goto info
14 if %input% EQU 3 goto backup
15 if %input% EQU 4 (
16     exit
17 ) else (
18     cls
19     goto menu
20 )
21 :start
22 app.py
23 web.py
24 set /p c=Nacisnij enter
25 cls
26 goto menu
27 :info

```

```
28 echo.
29 echo Program rozbija liczbe naturalna na wszystkie mozliwe kombinacje
    sum ktore daja jako wynik podana liczbe.
30 echo Program pobiera dane wejsciowe z pliku appIn.txt.
31 echo Efekt pierwszego skryptu jest zapisywany w pliku appOut.txt ktory
    jest nastepnie zamieniany na strone internetowa.
32 echo.
33 echo Autor: Mariusz Wrobel
34 echo.
35 set /p c=Nacisnij enter
36 cls
37 goto menu
38
39 :backup
40 set czas=%time:~0,8%
41 set czas=%czas:~0,8%
42 set name=Backup-%date%-%czas%
43 mkdir backup\%name%
44 copy *.* "backup\%name%" > nul
45 echo.
46 echo Backup zostal zapisany w folderze backup pod nazwa: %name%
47 echo.
48 set /p c=Nacisnij enter
49 cls
50 goto menu
51 pause
```
