

Języki Skrytpowe

dokumentacja projektu Szyfr Mendelejewa

Jakub Mrozek, grupa 3/6

3 stycznia 2023

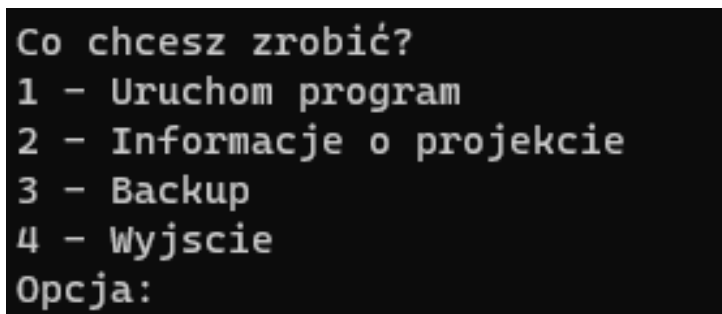
Część I

Opis programu

Program szyfruje podany w plikach wejściowych tekst w taki sposób, że zamiast kolejnych liter program wpisuje liczbę atomową pierwiastka, którego symbolem chemicznym są szyfrowane litery (wielkość liter nie jest uwzględniana). Odstęp między literami zastępuje symbol gwiazdki *, a odstęp między wyrazami - dwie gwiazdki **. Przykładowo, tekst SobOta rANo zaszyfrowany mógłby być ciągiem znaków 16*8*5*8*73**88*7*8 (siarka, tlen, bor, tlen tal; rad, azot, tlen). Niektóre teksty mogą być zaszyfrowane w różny sposób, na przykład wyraz nos zaszyfrowany może być na 7*8*16 (azot, tlen, siarka), 102*16 (nobel, siarka) lub 7*76 (azot, osm). Program bierze pod uwagę tylko jeden sposób - czyta on litery po kolei, kiedy napotka literę która nie jest znakiem pierwiastka, dodaje kolejną literę w wyrazie i sprawdza czy te dwie litery są znakiem pierwiastka, jeśli nie są wtedy program wypisuje komunikat o błędzie, w którym zawarte są litery nie będące symbolem żadnego pierwiastka.

Instrukcja obsługi

Aby uruchomić program należy włączyć skrypt *menu.bat*, który otwiera menu obsługi naszego programu. Po uruchomieniu wyświetli nam się menu programu w którym znajdują się 4 opcje.



```
Co chcesz zrobić?  
1 - Uruchom program  
2 - Informacje o projekcie  
3 - Backup  
4 - Wyjście  
Opcja:
```

Rysunek 1: Menu programu

1. **Uruchom program** - uruchamia program, który pobiera dane z plików tekstowych znajdujących się w folderze *input*, przetwarza je i wpisuje wynik do plików tekstowych znajdujących się w folderze *output*. Następnie program tworzy *raport.html* i otwiera go.

```
Opcja:
1
Skrypt został pomyślnie wykonany, otwieram wygenerowany raport...
Press any key to continue . . .
```

Rysunek 2: Komunikat po pomyślnej próbie uruchomienia programu

Raport	
Input	Output
SoB0Ta rANo	16*8*5*8*73**88*7*8
1414rak141 kr2424ab	88*19**19*88*5
algorytmion	Error: 'Go' nie jest pierwiastkiem.

Created by [Jakub Mrozek](#)

Rysunek 3: Raport programu

2. **Informacje o projekcie** - krótki opis działania programu

```
Opcja:
2
Autor projektu: Jakub Mrozek
-----
Program pobiera dane z plików "input/*.txt" i szyfruje zawarte tam słowa przy użyciu liczb atomowych pierwiastków z których skrótów składa się słowo. Pomiedzy każdą liczbą atomową wstawiona zostanie "*", a pomiedzy słowami "***".
-----
Przykład: SoB0Ta rANo - zostanie zaszyfrowane jako - 16*8*5*8*73**88*7*8. Niektórych słów nie da się zaszyfrować za pomocą programu (Nie sprawdza on wszystkich możliwych kombinacji, widząc pierwszą literę która jest symbolem pierwiastka zastępuje ją liczbą atomową, np. Hello - H = 1, El - nie istnieje).
-----
Cały projekt obudowany jest w menu, posiada ono cztery opcje: Uruchom, Opis, Backup oraz Wyjście. Główny program Pythona pobiera sobie pliki .txt z katalogu inputs, procesuje te dane i na wyjściu daje nam wynik, który zapisywany jest do plików "*.txt" w katalogu outputs. Dodatkowo tworzony jest "raport.html" generowany przez pythona. Backup robi kopie zapasową wszystkich inputów, outputów oraz "raport.html".
-----
Kliknij dowolny przycisk by wrócić do menu głównego.
Press any key to continue . . .
```

Rysunek 4: Informacje o projekcie

3. **Backup** - tworzy kopię zapasową plików (inputów, outputów oraz raportu) i zapisuje je w katalogu *Buckups*

```
Opcja:
3
input\1.txt
input\2.txt
input\3.txt
3 File(s) copied
output\1.txt
output\2.txt
output\3.txt
3 File(s) copied
D:styles.css
1 File(s) copied
D:raport.html
1 File(s) copied
Backup został pomyślnie wykonany, kliknij dowolny przycisk by powrócić do menu
Press any key to continue . . .
```

Rysunek 5: Tworzenie kopii zapasowej

4. **Wyjście** - wyłącza program

```
Opcja:
4
Zakończono działanie programu, kliknij dowolny przycisk by wyłączyć okno
Press any key to continue . . .
```

Rysunek 6: Wyjście z programu

Struktura danych programu

W skład struktury danych programu wchodzi następujące pliki, które są niezbędne do prawidłowego działania programu:

- *menu.bat* - skrypt Batch będący menu głównym, dzięki niemu uruchamiamy program i wyświetlamy informacje o jego założeniach i działaniu, jak również możemy stworzyć kopię zapasową danych otrzymanych w wyniku wykonania programu
- *info.bat* - skrypt Batch będący opisem programu, który otwiera się po wybraniu odpowiedniej opcji w menu głównym
- *main.py* - skrypt Python, który zawiera główny program. Pobiera on pliki wejściowe, znajdujące się w folderze *input*, zawierające zdania/słowa, które chcemy zaszyfrować i tworzy on pliki wyjściowe w folderze *output*, zawierające zaszyfrowane dane, bądź informację o błędzie w szyfrowaniu.
- *raport.py* - skrypt Python, który pobiera dane z plików wejściowych oraz wyjściowych i generuje plik *raport.html* zawierający raport danych w postaci tabeli.
- Folder *input* - zawierający wejściowe pliki tekstowe (*.txt)
- Folder *output* - zawierający wyjściowe pliki tekstowe (*.txt)
- *styles.css* - plik kaskadowego arkusza styli wykorzystywany podczas generowania raportu w celu jego stylizacji.

Program tworzy również folder *Backups* (jeśli taki nie istnieje), zawierający kopie zapasowe danych programu, po wybraniu odpowiedniej opcji w menu głównym.

Część II

Opis działania

Skrypt *menu.bat* włącza program *main.py*, który pobiera dane (pliki tekstowe) z folderu *input*. Następnie program formatuje dane zawarte w plikach usuwając wszelkie znaki nie będące literami i tworzy listę słów. Później aplikacja sprawdza poszczególne słowa z listy *words* i ich litery oraz zamienia je na liczbę atomową odpowiadającego im pierwiastka (pobiera tą informację z słownika *elements* zawierającego wszystkie pierwiastki i odpowiadające im liczby atomowe) i wpisuje ją do listy *code*. Jeśli litera nie odpowiada żadnemu symbolowi, wtedy program sprawdza czy kombinacja tej litery oraz następnej jest symbolem pierwiastka. Jeśli tak, wtedy litery te są zamieniane na liczbę atomową odpowiadającego im pierwiastka i liczba ta wpisywana jest do listy *code*. Jeśli nie, wtedy program wypisuje odpowiedni komunikat do pliku wyjściowego o tym, która kombinacja liter nie ma odpowiadającego jej symbolu pierwiastka. Jeśli jest to ostatnia litera w słowie i nie można sprawdzić kombinacji z następną literą, ponieważ nie istnieje, wtedy program wypisuje komunikat do pliku wyjściowego o tym, że nie ma takiego symbolu pierwiastka odpowiadającego podanej literze. Po przejściu przez słowo z listy *words*, program łączy zakodowane słowo z listy *code* dodając * po każdej liczbie atomowej i wpisuje całość do listy *result*. Po przejściu przez wszystkie słowa zawarte w pliku, główna funkcja łączy zawartość listy *result* dodając ** po każdym zakodowanym słowie i zwraca wynik. Następnie program tworzy pliki tekstowe i zapisuje w nich wyniki w folderze *output*.

Otrzymane wyniki następnie są pobierane przez *raport.py* i tworzy on *raport.html*, który zawiera tabelę z zawartością plików folderów *input* i *output*.

Następnie uruchamiana jest domyślna przeglądarka użytkownika, w której wyświetlany jest *raport.html*.

Algorytm

Data: Dane wejściowe plik *input*

Result: Dane wyjściowe plik *output*

Do funkcji *encrypt* przekaz dane z pliku *input*

for *word* **in** *words* **do**

while $i < \text{len}(\text{word})$ **do**

if *word*[*i*] **in** *elements* **then**

 Dodaj liczbę atomową odpowiadającego literze pierwiastka do *code*

i += 1

else

if $i < \text{len}(\text{word}) - 1$ **then**

 Dodaj do *symbol* następną literę **if** *symbol* **in** *elements* **then**

 Dodaj liczbę atomową odpowiadającego literom pierwiastka do *code*

i += 2

else

 Zwróć komunikat o braku odpowiedniego pierwiastka

end

else

 Zwróć komunikat o braku odpowiedniego pierwiastka

end

end

end

 Połącz *code* dodając * po każdej zaszyfrowanej literze Dodaj *code* do *result*

end

Połącz *result* dodając ** po każdym zaszyfrowanym słowie

Zwróć *result*

Implementacja systemu

Skrypt menu.bat po uruchomieniu i wybraniu opcji 1, następuje uruchomienie programu main.py. Program ten pobiera i zapisuje pliki wejściowe *input* do listy *filenames*, następnie przechodząc po pętli przez wszystkie pliki w liście, przekazuje pliki do funkcji *load_data*, która czytuje z nich dane i zwraca je. Następnie funkcja *encrypt* przyjmuje dane i szyfruje zawartość, zwracając na końcu wynik *result*, który przekazywany jest do funkcji *save_data*, która zapisuje zaszyfrowane dane do plików wyjściowych *output*. Wymagane jest aby stworzone były foldery *input* i *output*. Po wykonaniu skryptu uruchamiany zostaje raport.py, który tworzy raport w postaci pliku raport.html zawierający tabelę z danymi czytanyymi z plików *input* oraz *output*. Pobiera on także dane odnośnie stylizowania strony z pliku *styles.css*. Jeśli nie zostaną podane żadne pliki *input* program i tak stworzy pustą tabelę zawierającą jedynie nagłówki Input oraz Output.

Wybranie opcji *Backup* w menu głównym, spowoduje stworzenie folderu *Backups* (jeśli taki jeszcze nie istnieje), w nim zostanie stworzony folder nazwany odpowiednią datą i godziną kiedy została utworzona kopia zapasowa, a do niego przekopiowane zostaną foldery *input* i *output* oraz pliki *raport.html* i *styles.css*.

Wykorzystane biblioteki oraz przykłady

- OS

```
1 from os import walk
2 filenames = next(walk('input'), (None, None, []))[2]
3 //Tworzenie listy plikow input
```

Pełen kod aplikacji

main.py

```
1 from os import walk
2 filenames = next(walk('input'), (None, None, []))[2]
3
4
5 def encrypt(text):
6     elements = {
7         "H": 1, "He": 2, "Li": 3, "Be": 4, "B": 5, "C": 6, "N": 7, "O":
8         8, "F": 9, "Ne": 10,
9         "Na": 11, "Mg": 12, "Al": 13, "Si": 14, "P": 15, "S": 16, "Cl":
10        17, "Ar": 18, "K": 19, "Ca": 20,
11        "Sc": 21, "Ti": 22, "V": 23, "Cr": 24, "Mn": 25, "Fe": 26, "Co":
12        27, "Ni": 28, "Cu": 29, "Zn": 30,
13        "Ga": 31, "Ge": 32, "As": 33, "Se": 34, "Br": 35, "Kr": 36, "Rb":
14        37, "Sr": 38, "Y": 39, "Zr": 40,
15        "Nb": 41, "Mo": 42, "Tc": 43, "Ru": 44, "Rh": 45, "Pd": 46, "Ag":
16        47, "Cd": 48, "In": 49, "Sn": 50,
17        "Sb": 51, "Te": 52, "I": 53, "Xe": 54, "Cs": 55, "Ba": 56, "La":
18        57, "Ce": 58, "Pr": 59, "Nd": 60,
19        "Pm": 61, "Sm": 62, "Eu": 63, "Gd": 64, "Tb": 65, "Dy": 66, "Ho":
20        67, "Er": 68, "Tm": 69, "Yb": 70,
21        "Lu": 71, "Hf": 72, "Ta": 73, "W": 74, "Re": 75, "Os": 76, "Ir":
22        77, "Pt": 78, "Au": 79, "Hg": 80,
23        "Tl": 81, "Pb": 82, "Bi": 83, "Po": 84, "At": 85, "Rn": 86, "Fr":
24        87, "Ra": 88, "Ac": 89, "Th": 90,
25        "Pa": 91, "U": 92, "Np": 93, "Pu": 94, "Am": 95, "Cm": 96, "Bk":
26        97, "Cf": 98, "Es": 99, "Fm": 100,
27        "Md": 101, "No": 102, "Lr": 103, "Rf": 104, "Db": 105, "Sg":
28        106, "Bh": 107, "Hs": 108, "Mt": 109,
29        "Ds": 110, "Rg": 111, "Cn": 112, "Nh": 113, "Fl": 114, "Mc":
30        115, "Lv": 116, "Ts": 117, "Og": 118,
31    }
32
33    for i in text:
34        if not i.isalpha():
35            text = text.replace(i, '')
36    words = text.split()
37    result = []
38
39    for word in words:
40        code = []
```



```

29         i = 0
30         while i < len(word):
31             if word[i].upper() in elements:
32                 code.append(str(elements[word[i].upper()]))
33                 i += 1
34             else:
35                 if i < len(word) - 1:
36                     symbol = word[i:i + 2]
37                     if symbol.capitalize() in elements:
38                         code.append(str(elements[symbol.capitalize()]))
39                         i += 2
40                     else:
41                         return "Error: '" + word[i:i+2].capitalize() + "
                                ' nie jest pierwiastkiem."
42                 else:
43                     return "Error: '" + word[i] + "' nie jest symbolem
                                pierwiastka."
44         code = '*,'.join(code)
45         result.append(code)
46
47     result = '**',.join(result)
48
49     return result
50
51
52 def load_data(filename):
53     with open(f'input/{filename}', 'r') as file:
54         data = file.read()
55     return data
56
57
58 def save_data(data):
59     with open(f'output/{filename}', 'w') as file:
60         file.write(data)
61
62
63 for filename in filenames:
64     data = load_data(filename)
65     data = encrypt(data)
66     save_data(data)

```

raport.py

```

1 from os import walk
2 inputFiles = next(walk("input"), (None, None, []))[2]
3 outputFiles = next(walk("output"), (None, None, []))[2]
4
5 inputsList = []
6 outputsList = []
7
8 for filename in inputFiles:
9     with open(f"input/{filename}") as file:
10         lines = file.readlines()

```

```

11         result = "".join(lines).strip()
12         inputsList.append(result)
13
14     for filename in outputFiles:
15         with open(f"output/{filename}") as file:
16             lines = file.readlines()
17             result = "".join(lines).strip()
18             outputsList.append(result)
19
20     elementy = []
21     for element in range(0, len(inputsList)):
22         element = f"<tr> <th>{inputsList[element]} </th><th> {outputsList[
23             element]}</th> </tr>"
24         elementy.append(element)
25
26     file_html = open("raport.html", "w")
27     file_html.write('''
28     <html>
29     <head>
30         <title>Raport</title>
31         <link rel="stylesheet" href="styles.css">
32         <meta name="viewport" content="width=device-width, initial-scale
33             =1">
34         <link rel="preconnect" href="https://fonts.googleapis.com">
35         <link rel="preconnect" href="https://fonts.gstatic.com"
36             crossorigin>
37         <link href="https://fonts.googleapis.com/css2?family=Roboto:
38             wght@300;700&display=swap" rel="stylesheet">
39     </head>
40     <body>
41         <div class="container">
42             <h1>Raport</h1>
43             <table>
44                 <tr class="top-row">
45                     <th>Input</th>
46                     <th>Output</th>
47             </table>
48             ''')
49
50     for element in elementy:
51         file_html.write(element)
52
53     file_html.write('''
54     </table>
55     <footer>
56         <p>Created by <a href=https://github.com/Hunterlios>
57             Jakub Mrozek</a></p>
58     </footer>
59     </div>
60     </body>
61     </html>
62     ''')
63
64     file_html.close()

```

menu.bat

```
1 @echo off & chcp 65001
2
3 :start
4 cls
5
6 echo Co chcesz zrobic?
7 echo 1 - Uruchom program
8 echo 2 - Informacje o projekcie
9 echo 3 - Backup
10 echo 4 - Wyjscie
11 echo Opcja:
12
13 set /p whatapp=
14
15 if %whatapp%==1 (
16     goto 1
17 ) else if %whatapp%==2 (
18     goto 2
19 ) else if %whatapp%==3 (
20     goto 3
21 ) else if %whatapp%==4 (
22     goto 4
23 )
24
25 :1
26 python "main.py"
27 python "raport.py"
28 echo Skrypt zostal pomyslnie wykonany, otwieram wygenerowany raport...
29 "raport.html"
30 pause
31 goto start
32
33 :2
34 info.bat
35 goto start
36
37
38 :3
39 for /f %%a in ('powershell -Command "Get-Date -format
    yyyy_MM_dd__HH_mm_ss"') do set datetime=%%a
40 if not exist "Backups" mkdir Backups
41 cd Backups
42 mkdir "%datetime%"
43 cd %datetime%
44 mkdir input
45 mkdir output
46 cd ../
47 cd ../
48 xcopy /s input Backups\"%datetime%\input
49 xcopy /s output Backups\"%datetime%\output
50 xcopy styles.css Backups\"%datetime%"
51 xcopy raport.html Backups\"%datetime%"
52 echo Backup zostal pomyslnie wykonany, kliknij dowolny przycisk by
```

```
        powrocić do menu
53 pause
54 goto start
55
56 :4
57 echo Zakonczono dzialanie programu, kliknij dowolny przycisk by wylaczyc
    okno
58 pause
```

info.bat

```
1 @echo off
2 echo Autor projektu: Jakub Mrozek
3
4 echo -----
5
6 echo Program pobiera dane z plikow "input/*.txt" i szyfruje zawarte tam
    slowa przy uzyciu liczb atomowych pierwiastkow z ktorych skrotow
    sklada sie slowo. Pomiedzy kazda liczba atomowa wstawiona zostanie "*"
    ", a pomiedzy slowami "***".
7
8 echo -----
9
10 echo Przyklad: Sob0ta rANo - zostanie zaszyfrowane jako -
    16*8*5*8*73**88*7*8. Niektorych slow nie da sie zaszyfrowac za pomoca
    programu (Nie sprawdza on wszystkich mozliwych kombinacji, widzac
    pierwsza litere ktora jest symbolem pierwiastka zastepuje ja liczba
    atomowa, np. Hello - H = 1, El - nie istnieje).
11
12 echo -----
13
14 echo Caly projekt obudowany jest w menu, posiada ono cztery opcje:
    Uruchom, Opis, Backup oraz Wyjscie. Glowny program Pythona pobiera
    sobie pliki .txt z katalogu inputs, procesuje te dane i na wyjsciu
    daje nam wynik, ktory zapisywany jest do plikow "*.txt" w katalogu
    outputs. Dodatkowo tworzony jest "raport.html" generowany przez
    pythona. Backup robi kopie zapasowa wszystkich inputow, outputow oraz
    "raport.html".
15
16 echo -----
17 echo .
18 echo Kliknij dowolny przycisk by wrocic do menu glownego.
19 pause
20 menu.bat
```
