

Dokumentacja Projektu

Języki Skryptowe

Paweł Pietraszko, Gr III F

Opis działania programu

Program szyfruje wszystkie pliki tekstowe z folder *tekst* do folderu *zaszyfrowane* i następnie deszyfruje do folderu *odszyfrowane* i na koniec zostaje utworzona strona w folderze *stronaHTML*. Strona internetowa jest sformatowana za pomocą pliku CSS.

Założenia i funkcjonalność programu oparta jest na poleceniu do zadania nr 1 - "Szyfrowanie wiadomości" z konkursu *Algorytmion* 2015.

ZADANIE 1 - "SZYFROWANIE WIADOMOŚCI"

Zadanie zaproponował: mgr Krzysztof Jarczewski, III LO im. S. Batorego w Chorzowie

Adam i Janek ustalili między sobą, że każdą wiadomość tekstową, będą szyfrować przy pomocy następującego sposobu:

- Każdą literę i znak interpunkcyjny należy zamienić na odpowiadającą liczbę kodu ASCII (liczba naturalna od 0 do 127).
- Pierwsza, tak powstała liczba, jest zamieniana na system dwójkowy.
- Kolejne liczby, wynikające z kodu ASCII, są zamieniane na system liczbowy, który jest równy powiększonej o dwa reszcie z dzielenia przez osiem poprzedniej liczby.
- Ilość cyfr zamienionej liczby, dla każdego systemu liczbowego, wynika z ilości cyfr zamiany liczby maksymalnej, czyli liczby 127 (dla systemu binarnego jest to siedem cyfr).

Twoim zadaniem jest napisanie programu, który szyfruje i deszyfruje wiadomości.

Szyfrowanie tekstu z pliku *tekst.txt* do pliku *szyfr.txt*

Deszyfrowanie tekstu z pliku *szyfr.txt* do pliku *odszyfrowane.txt*

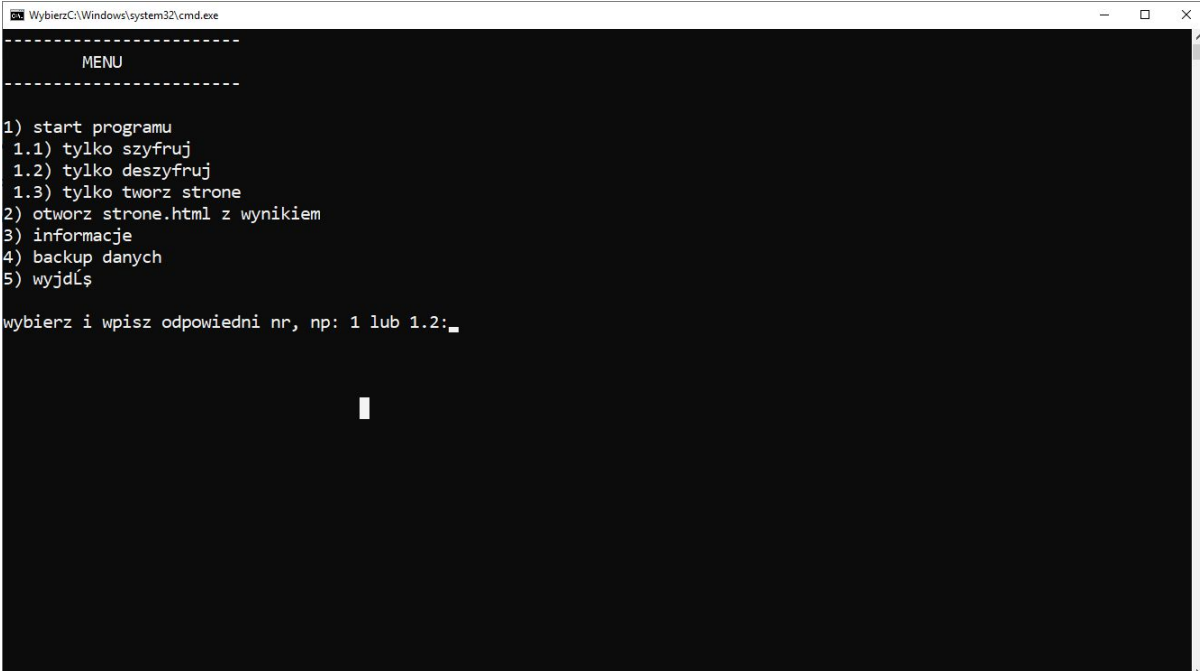
Przykład

| | | |
|-------|--------------|--|
| tekst | kod ASCII | system liczbowy: 2, $(66 \bmod 8)+2=4$, $(105 \bmod 8)+2=3$ |
| Bit | → 66 105 116 | → 1000010 1221 11022 |

Każda linijka podczas szyfrowania i deszyfrowania jest traktowana jako osobna wiadomość. Ma to znaczenie, gdyż każdy pierwszy znak szyfrowany jest w taki sam sposób, a na jego podstawie jego kodu ASCII wyliczany jest sposób szyfrowania kolejnego znaku.

Obsługa programu

Do obsługi funkcjonalności, jaką oferuje program, najlepiej jest wykorzystać proste menu *menu.bat*:



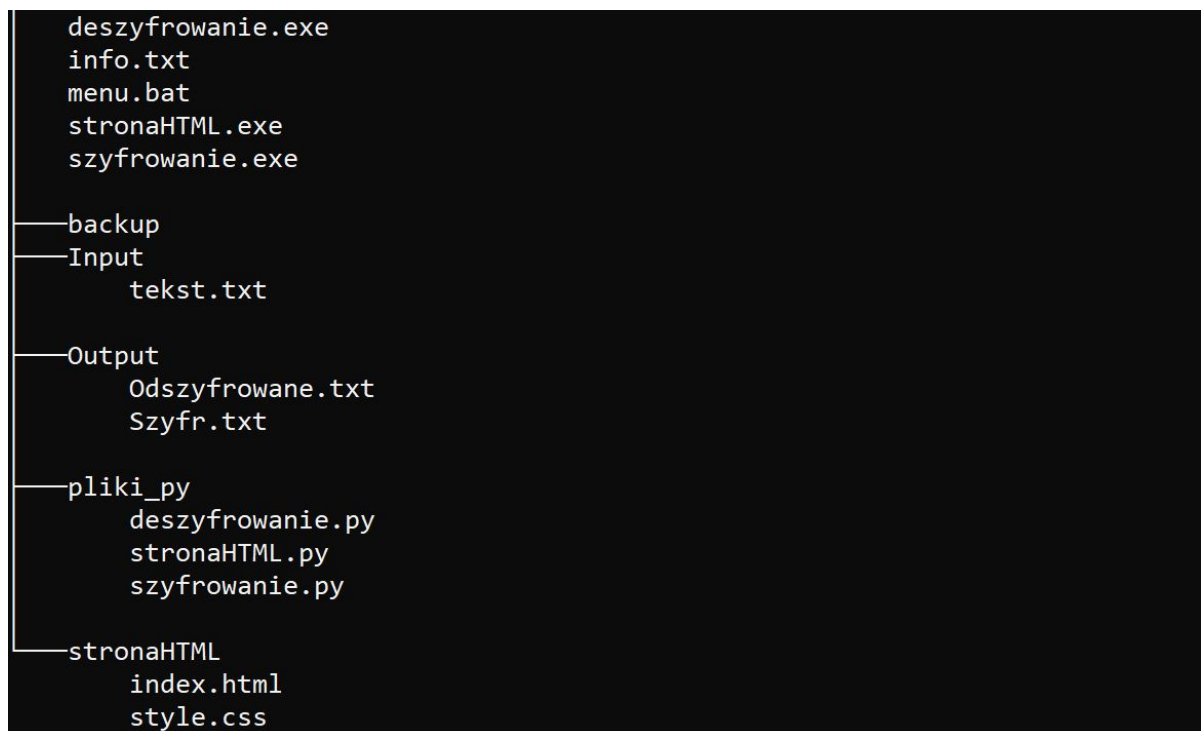
```
WybierzC:\Windows\system32\cmd.exe
-----
MENU
-----
1) start programu
  1.1) tylko szyfruj
  1.2) tylko deszyfruj
  1.3) tylko tworź stronę
2) otwórz stronę.html z wynikiem
3) informacje
4) backup danych
5) wyjdź

wybierz i wpisz odpowiedni nr, np: 1 lub 1.2:
|
```

- 1) Odpala kolejno: program szyfrujący, program deszyfrujący, program budujący stronę HTML i tworzy backup danych,
 - 1.1) Tylko szyfruje dane,
 - 1.2) Tylko deszyfruje dane,
 - 1.3) Tylko tworzy stronę HTML,
- 2) otwiera stronę HTML z wynikami w domyślnej przeglądarce, strona formatowana za pomocą pliku .css,
- 3) wyświetla krótki opis działania programu, tekst porabierany z pliku *info.txt*,
- 4) tworzy backup całego folderu (za wyjątkiem folderu *backup*) w folderze *backup/yyyy.mm.dd_hh.mm.ss (data_czas)*
- 5) zamyka program.

Każde polecenie zostaje opisane odpowiednim komunikatem, a następnie program oczekuje na wciśnięcie dowolnego klawisza, aby można było zapoznać się z komunikatem.

Struktura plików



Główny folder zawiera 3 głównie pliki:

- *szyfrowanie.py* (szyfruje pliki txt z folderu *tekst* do *zaszyfrowane*)
- *deszyfrowanie.py* (deszyfruje dane z pliki z folderu *zaszyfrowane* do *odszyfrowane*)
- *stronaHTML.py* (tworzy stronę w folderze *stronaHTML* z wynikami powyższych

poleceń)

oraz:

- *menu.bat* (, który był powyżej opisany)
- *info.txt* (zawierający krótką informację o działaniu programu i wywoływany z menu).

Zdecydowałem się podzielić główny program na 3 podprogramy, aby łatwiej było nimi zarządzać.

Backup - podfolder, w którym zapisywane są backupy (na zrzucie powyżej pusty).

tekst - podfolder, zawiera pliki tekstowe do zaszyfrowania

zaszyfrowane - podfolder, zawiera zaszyfrowane pliki

stronaHTML - podfolder, zawiera strony html z wynikami szyfrowania oraz plik css służący formatowaniu stron.

Implementacja

Szyfrowanie:

dane wejściowe:

`str` - ciąg znaków w postaci string

dane programu:

`sys=2` (system kodowania)

`tab=[]` (pusta tablica)

program:

```
for i=0; i<długość(str); i++:  
    str[i] zamień na liczbę całkowitą (ASCII)  
    ord(str[i]) na system o podstawie sys  
    dołącz zakodowany znak do tab  
    sys= ord(str[i])%8+2  
zwróć tab
```

`-ord()` - funkcja zamieniająca znak na liczbę całkowitą na podstawie tablicy ASCII

Zamiana liczby dziesiętnej na inny system:

Dane wejściowe:

`l` - liczba do zakodowania

`sys` - nowy system liczbowy

Dane programu:

`wynik` - pusta tablica

program:

```
powtarzaj póki l != 0:  
    dołącz do wynik l (mod sys)  
    l=(l-(l%sys))/sys  
odwróć tablicę wynik  
zwróć wynik
```

Deszyfrowanie:

Dane wejściowe:

str - ciąg znaków (liczb) rozdzielonych spacją

Dane programu:

wynik, **tab_str** - puste tablica

sys = 2 (system liczbowy za pomocą którego deszyfrujemy znak)

suma = 0

Program:

przypisz do **tab_str** zmienną **str** podzieloną co spację (" ")

for **i**=0; **i**<długość(**tab_str**);**i**++:

 for **j**=0; **j**<długość(**tab_str**[**i**]);**j**++:

suma = **tab_str**[**i**] zamieniony na postać dziesiętną (z systemu o podstawie

sys)

sys = (**suma** (mod 8))+2

 dołącz do **wynik** chr(**suma**)

zwróć **wynik**

-chr() - zamienia postać całkowitą na znak (na podstawie tablicy ASCII)

-zamiana na postać dziesiętną następuje przez pomnożenie tej liczby poprzez **sys** do potęgi proporcjonalnej do jej miejsca (licząc od prawej)

-pętla ze zmienną **j** ma zastosowanie w realnym programie (przechodzi po kolejnych znakach zmiennej **tab_str**[**i**] (tablicy 2-wymiarowej)

Kody z plików:

menu.bat:

```
@echo off
@chcp 1250
attr 0F
:menu
cls
echo -----
echo  MENU
echo -----
echo.
echo 1) start programu
echo  1.1) tylko szyfruj
echo  1.2) tylko deszyfruj
echo  1.3) tylko tworzy strone
echo 2) informacje
echo 3) backup danych
echo 4) wyjdź
echo.
set /p opcja=Wybierz i wpisz odpowiedni nr, np: 1 lub 1.2:
if %opcja%==1 goto start
if %opcja%==1.1 goto szyfrowanie
if %opcja%==1.2 goto deszyfrowanie
if %opcja%==1.3 goto strona
if %opcja%==2 goto info
if %opcja%==3 goto backup
if %opcja%==4 exit
goto error

:start
cls
for %%f in (tekst/*.txt) do python szyfrowanie.py tekst/%%f zaszyfrowane/%%f
echo zaszyfrowano dane

for %%f in (zaszyfrowane/*.txt) do python deszyfrowanie.py zaszyfrowane/%%f
odszyfrowane/%%f
echo zdeszyfrowano dane

for %%f in (zaszyfrowane/*.txt) do python stronaHTML.py zaszyfrowane/%%f
odszyfrowane/%%f stronaHTML/%%f
echo utworzono strone HTML z wynikami

pause >null
```

goto backup

:szyfrowanie

for %%f in (tekst/*.txt) do python szyfrowanie.py tekst/%%f zaszyfrowane/%%f

echo zaszyfrowano dane

pause >null

goto backup

:deszyfrowanie

for %%f in (zaszyfrowane/*.txt) do python deszyfrowanie.py zaszyfrowane/%%f

odszyfrowane/%%f

echo zdeszyfrowano dane

pause >null

goto backup

:strona

for %%f in (zaszyfrowane/*.txt) do python stronaHTML.py zaszyfrowane/%%f

odszyfrowane/%%f stronaHTML/%%f

echo utworzono strone HTML z wynikami

pause >null

goto backup

:info

cls

type info.txt

pause >null

goto menu

:backup

cls

if not exist backup md backup

set

datestr=%date:~-4,4%.%date:~-7,2%.%date:~-10,2%_%time:~0,2%.%time:~3,2%.%time:~6,2%

md backup/%datestr%

robocopy ..\projekt_sem .\backup/%datestr% /S /XD backup

cls

echo stworzono kopie plikow w folderze backup/%datestr%

pause >nul

goto menu

:error

echo wpisano nieprawidlowe dane, wybierz sposrod: {1, 1.1, 1.2, 1.3, 2, 3 ,4}

pause >null

goto menu

szyfrowanie.py:

```
import sys

def DecToAny(num,syst):
    mod=[]
    while num!=0:
        mod.append(num%syst)
        num=int((num-(num%syst))/syst)
    mod.reverse()
    return "".join(str(x) for x in mod)

def szyfrowanie(str):
    tab=[]
    syst=2
    for i in range(len(str)):
        tab.append(DecToAny(ord(str[i]),syst))
        syst=(ord(str[i])%8)+2
    return " ".join(tab)

def pelne_szyfrowanie(path_input, path_output):
    f_in=open(path_input,"r")
    f_out=open(path_output,"w")
    for l in f_in:
        f_out.write(szyfrowanie(l.rstrip())+'\n')
    f_in.close()
    f_out.close()

pelne_szyfrowanie(sys.argv[1],sys.argv[2])
```

deszyfrowanie.py:

```
import sys as s

def deszyfrowanie(str):
    tab_str=str.split(" ")
    wynik=[]
    sys=2
    for i in range(len(tab_str)):
        suma=0
        dl=len(tab_str[i])
        for j in range(dl):
            suma+=int(tab_str[i][dl-1-j])*pow(sys,j)
        sys=(suma%8)+2
        wynik.append(chr(suma))
    return "".join(wynik)

def pelne_deszyfrowanie(path_to_input, path_to_output):
    f_in = open(path_to_input,"r")
    open(path_to_output, 'w').close()
    f_out = open(path_to_output,"a")
    for l in f_in:
        f_out.write(deszyfrowanie(l.rstrip())+"\n")
    f_in.close()
    f_out.close()
pelne_deszyfrowanie(s.argv[1],s.argv[2])
```

stronaHTML.py:

```
def HTML_table_creator(file_1, file_2):
```

```
    table="<table class='table-fill'><tr><td class='text-left'>Zaszyfrowany tekst</td><td  
class='text-left'>odszyfrowany tekst</td></tr>\n"  
    for l, k in zip(file_1, file_2):  
        table = table+"<tr><td class='text-left'>",l,"</td><td class='text-left'>",k,"</td></tr>\n"  
    table=table+"</table>"  
    return table
```

```
def HTML_table(data_file1, data_file2,w_file):
```

```
    f_szyfr = open(data_file1,"r")  
    f_deszyfr = open(data_file2,"r")  
    open(w_file,"w").close()  
    f_zapis = open(w_file,"w")  
    f_zapis.write("<title>Wyniki</title>")  
    f_zapis.write("<div class='table-title'>")  
    f_zapis.write("<h3>Wyniki szyfrowania i deszyfrowania</h3>")  
    f_zapis.write("</div>")  
    f_zapis.write("<meta name='viewport' content='initial-scale=1.0; maximum-scale=1.0;  
width=device-width;'>\n")  
    f_zapis.write("<link rel='stylesheet' type='text/css' href='style.css'>\n")  
    f_zapis.write("<table class='table-fill'><tr><th class='text-left'>Zaszyfrowany tekst</th><th  
class='text-left'>Odszyfrowany tekst</th></tr>\n")  
    for l, k in zip(f_szyfr, f_deszyfr):  
        f_zapis.write("<tr><td class='text-left'>"+l+"</td><td class='text-left'>"+k+"</td></tr>\n")  
    f_zapis.write("</table>")  
    f_zapis.close()  
    f_szyfr.close()  
    f_deszyfr.close()
```

```
HTML_table("Output/Szyfr.txt", "Output/Odszyfrowane.txt", "stronaHTML/index.html")
```

