Laboratorium kryptograficzne dla gimnazjalistów 3

Projekt „Matematyka dla ciekawych świata”
Łukasz Mazurek

21.04.2016

1 Wczytywanie danych z pliku

Do tej pory wszystkie dane, z których korzystały nasze programy, wprowadzaliśmy bezpośrednio do kodu programu. Tym razem spróbujemy rozszyfrować dłuższą wiadomość zapisaną w pliku. Dlatego na dzisiejszych zajęciach nie będziemy już korzystać z online'owego interpretera Pythona `repl.it` — programy będziemy uruchamiać poleceniem `python3` dostępnym na komputerach w laboratoriach. Aby skorzystać z tego samego polecenia na Waszych komputerach domowych, wystarczy, że zainstalujecie interpreter Pythona dostępny za darmo do pobrania ze strony `https://www.python.org/downloads/` (zwróćcie uwagę, aby zainstalować wersję o numerze rozpoczynającym się od 3).

Aby rozpocząć pracę z naszym plikiem, otwórz terminal, stwórz w katalogu domowym podkatalog `krypto` i wejdź do tego katalogu poleceniami

```
mkdir krypto
cd krypto
```

Następnie otwórz edytor tekstu (np. Kate), stwórz w nim pusty plik i zapisz w naszym katalogu `krypto` pod nazwą `program.py`. Na koniec zapisz plik `szyfr.txt` pobrany ze strony `ciekawi.icm.edu.pl` (zakładka *Dla uczestników* → *Materiały - ćwiczenia*) również w tym samym katalogu. W tym momencie katalog `krypto` powinien zawierać dwa pliki: `program.py` i `szyfr.txt`. Aby się o tym przekonać, możesz użyć polecenia `ls`, które wypisuje zawartość katalogu, w którym się aktualnie znajdujemy:

```
$ ls
program.py      szyfr.txt
```

Teraz zmień zawartość pliku `program.py` na następującą:

```
plik = open('szyfr.txt', 'r')
tekst = plik.readline()
print(tekst)
```

Aby uruchomić napisany właśnie program, wpisz w konsoli polecenie `python3 program.py` i naciśnij Enter. Program powinien wypisać pierwszy wiersz pliku `szyfr.txt`. Co się stało?

- Polecenie z pierwszej linijki otwiera plik `szyfr.txt` i zapewnia dostęp do niego poprzez zmienną `plik`. Opcja `'r'` oznacza, że plik jest otwarty „do odczytu” (od angielskiego *read*).
- Druga komenda czyta pierwszy wiersz z pliku `szyfr.txt` i zapisuje go w zmiennej `tekst`.
- Ostatnie polecenie wypisuje zawartość zmiennej `tekst`.

Napiszemy teraz program, który zlicza liczbę liter A występujących w pliku `szyfr.txt`

```
plik = open('szyfr.txt', 'r')
tekst = plik.read()
ile_A = 0
for znak in tekst:
    if znak == 'A':
        ile_A = ile_A + 1
print(ile_A)
```

W powyższym kodzie:

- Wczytujemy całą zawartość pliku poleceniem `plik.read()` i zapisujemy w zmiennej `tekst`.
- Zmienna `ile_A` (początkowo równa 0) przechowuje aktualną liczbę zliczonych liter A.
- Za każdym razem, gdy trafimy w tekście na kolejną literę A, zwiększamy wartość zmiennej `ile_A` o jeden.

Zadanie 1 *Napisz program, który policzy, ile wielkich liter alfabetu łacińskiego występuje łącznie w pliku* `szyfr.txt`.

2 Zliczanie częstości występowania wszystkich liter alfabetu

Aby zliczyć wystąpienia litery A, potrzebowaliśmy zmiennej `ile_A`. Aby zliczyć wystąpienia każdej z 26 liter alfabetu z osobna, potrzebujemy 26 zmiennych. Możemy to jednak zrobić wygodniej, korzystając z 26-elementowej listy wypełnionej początkowo zerami:

```
plik = open('szyfr.txt', 'r')
tekst = plik.read()
ile = [0] * 26
for znak in tekst:
    if ord(znak) >= ord('A') and ord(znak) <= ord('Z'):
        poz = ord(znak) - ord('A')
        ile[poz] = ile[poz] + 1
print(ile)
```

W powyższym kodzie:

- Napis `[0] * 26` tworzy listę `[0, 0, ..., 0]` składającą się z 26 zer. Listę tę zapisujemy następnie w zmiennej `ile`.
- Dla każdego znaku z tekstu sprawdzamy, czy to wielka litera i jeśli tak, to obliczamy jej pozycję w alfabecie i zapisujemy w zmiennej `poz`.
- Zwiększamy wartość na pozycji `poz` w liście `ile`, czyli wartość zliczającą liczbę wystąpień znaku `znak`.

Po wykonaniu powyższego kodu w zmiennej `ile` znajduje się 26-elementowa lista zawierająca liczby wystąpień kolejnych liter alfabetu. Dane te najwygodniej jest przedstawić w formie wykresu.

3 Wizualizacja listy w formie wykresu

Do rysowania wykresów w Pythonie służą polecenia `plot` i `show`. Aby zwizualizować zawartość listy `ile` na wykresie, dopisz na końcu programu tworzącego tę listę linijki:

```
plot(ile)
xticks(range(26), ascii_uppercase)
show()
```

W powyższym kodzie:

- Pierwsza linijka mówi, że chcemy umieścić na wykresie serię danych z listy `ile`.
- Druga linijka sprawia, że kolejne wartości na osi X będą podpisane literami A, B, C, ..., Z zamiast liczbami 0, 1, ..., 25.
- Trzecia linijka to polecenie wyświetlenia wykresu na ekranie komputera.

Niektóre komendy, których użyliśmy pochodzą spoza zbioru podstawowych komend Pythona. Dlatego na początku programu musimy je najpierw zaimportować z odpowiednich bibliotek. W tym celu musimy dopisać na początku programu linijki:

```
from pylab import plot, show, xticks
from string import ascii_uppercase
```

W pierwszej linijce importujemy polecenia `plot`, `show` i `xticks` z biblioteki `pylab`, a w drugiej z biblioteki `string` importujemy stałą `ascii_uppercase`, która zawiera listę składającą się z kolejnych wielkich liter alfabetu łacińskiego.

Ostatecznie, nasz program powinien wyglądać następująco:

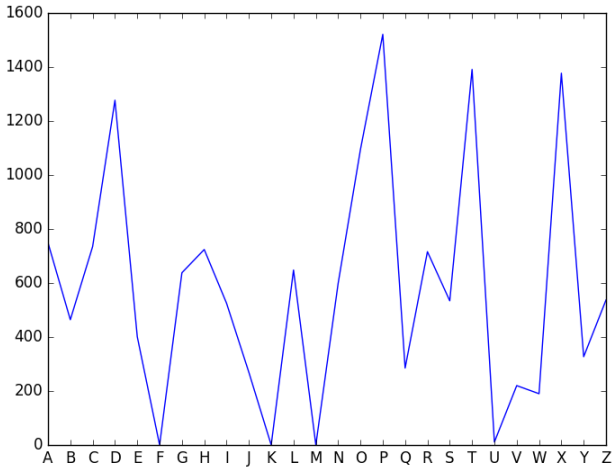
```
from pylab import plot, show, xticks
from string import ascii_uppercase

plik = open('szyfr.txt', 'r')
tekst = plik.read()
ile = [0] * 26
for znak in tekst:
    if ord(znak) >= ord('A') and ord(znak) <= ord('Z'):
        poz = ord(znak) - ord('A')
        ile[poz] = ile[poz] + 1

plot(ile)
xticks(range(26), ascii_uppercase)
show()
```

Po zapisaniu treści tego programu w pliku `program.py` i uruchomieniu go w konsoli poleceniem `python3 program.py` na ekranie powinien wyświetlić się wykres taki jak na rys. 1.

Jeśli zamiast wykresu Python wypisze komunikat o błędzie związanym z kodowaniem UTF-8, niezbędne może okazać się uprzednie wykonanie w konsoli poleceń:

```
export LC_ALL=pl_PL.UTF-8
export LANG=pl_PL.UTF-8
```

Służę one poprawnemu skonfigurowaniu bibliotek z których korzystamy z naszym kodowaniem znaków UTF-8.



Rysunek 1: Wykres częstości występowania poszczególnych liter w szyfrogramie

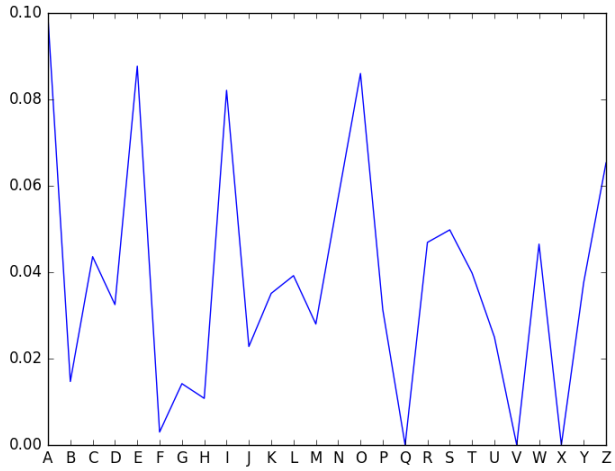
4 Porównanie wykresu z częstościami występowania liter w języku polskim

W pliku `czestosci_pl.txt` (dostępnym do pobrania ze strony `ciekawi.icm.edu.pl`, zakładka *Dla uczestników* → *Materiały - ćwiczenia*) znajduje się 26-elementowa lista częstości występowania poszczególnych liter w języku polskim. Dla uproszczenia zajmujemy się tutaj „bezogonkową" wersją polskiego alfabetu — kolejne wartości na tej liście odpowiadają kolejnym literom alfabetu ABCDEFGHIJKLMNOPQRSTUVWXYZ.

W kodzie naszego programu stwórz zmienną `czestosci` i zapisz do niej listę z pliku `czestosci_pl.txt`, a następnie narysuj wykres przedstawiający tę listę:

```
czestosci = [0.099, 0.0147, 0.0436, 0.0325, 0.0877, 0.003, 0.0142,
0.0108, 0.0821, 0.0228, 0.0351, 0.0392, 0.028, 0.0572, 0.086, 0.0313,
0, 0.0469, 0.0498, 0.0398, 0.025, 0, 0.0465, 0, 0.0376, 0.0653]
plot(czestosci)
xticks(range(26), ascii_uppercase)
show()
```

Powinieneś uzyskać wykres taki jak na rys. 2.



Rysunek 2: Wykres częstości występowania poszczególnych liter w języku polskim

Spróbujmy teraz umieścić oba wykresy na jednym rysunku. W tym celu należy dwa razy wykonać polecenie `plot`, oddzielnie dla każdej listy:

```
plot(czestosci)
plot(ile)
xticks(range(26), ascii_uppercase)
show()
```

Okaże się, że jeden z wykresów jest całkowicie niewidoczny. Dzieje się tak, ponieważ elementy listy `czestosci` nie przekraczają wartości 0.1, natomiast elementy listy `ile` sięgają liczby 1500. Dlatego wykres listy `czestosci` jest na tyle „niski" w porównaniu do wykresu listy `ile`, że praktycznie w całości pokrywa się z linią na poziomie 0.

Zadanie 2 *Przeskaluj listę* `ile`*, tak aby jej elementy odpowiadały* ***względnym*** *częstościom występowania poszczególnych liter w szyfrogramie. W tym celu policz liczbę wystąpień wszystkich wielkich liter alfabetu w szyfrogramie, a następnie podziel każdy element listy przez tę liczbę.*

Narysuj ponownie wykresy listy `ile` *(przeksalowanej) i listy* `czestosci` *na jednym rysunku. Tym razem wykresy powinny być podobnej wysokości.*

5 Przesuwanie wykresu

Jeśli poprawnie rozwiązałeś poprzednie zadanie, powinieneś otrzymać dwa wykresy o podobnej wysokości, jednak poszczególne „górki" i „dołki" powinny występować w różnych miejscach. W rzeczywistości tekst z pliku `szyfr.txt` został zaszyfrowany szyfrem Cezara, więc lista `ile` powinna odpowiadać cyklicznemu przesunięciu listy `czestosci`.

W Pythonie możemy w łatwy sposób przesunąć cyklicznie listę. Aby np. przesunąć cyklicznie listę o 3 pozycje w lewo możemy użyć konstrukcji:

```
lista[3:] + lista[:3]
```

Pierwsze wyrażenie zwraca fragment listy od elementu o indeksie 3 (włącznie) do końca, a drugie wyrażenie zwraca fragment listy od początku do elementu o indeksie 3 (wyłącznie). Znak dodawania pomiędzy wyrażeniami oznacza, że oba fragmenty zostają „sklejone" w jedną listę.

Zadanie 3 *Wiedząc, że szyfrogram z pliku* `szyfr.txt` *powstał przez zaszyfrowanie szyfrem Cezara pewnego tekstu napisanego w języku polskim, znajdź klucz, którym tekst został zaszyfrowany. W tym celu znajdź takie przesunięcie, żeby przesunięty wykres częstości występowania liter w szyfrogramie pokrywał się z wykresem częstości występowania liter w języku polskim. Dodatkowo, możesz sprawdzić, że odnalazłeś właściwy klucz odszyfrowując tekst szyfrogramu metodami poznanymi na poprzednich zajęciach.*

Wskazówka: w języku polskim nie występują litery Q, V i X.

6 Praca domowa nr 3

Rozwiązania zadań należy przesłać do czwartku 28 kwietnia do godz. 15^{59} na adres `gimnazjalisci.pracownia@icm.edu.pl` wpisując jako temat wiadomości `Gx PD3`, gdzie `x` to numer grupy, np. `G3 PD3` dla grupy G3, itd.

We wszystkich zadaniach będziemy korzystać z pliku `vig5.txt` (dostępny do pobrania ze strony `ciekawi.icm.edu.pl`, zakładka *Dla uczestników* → *Materiały - ćwiczenia*).

Zadanie domowe 1 *Napisz program, który wczyta tekst z pliku* `vig5.txt` *i wypisze na ekran co piąty znak tego tekstu, począwszy od pierwszego (liczymy wszystkie znaki, również spacje i znaki interpunkcyjne).*

Dla przykładu, początek pliku `vig5.txt` *wygląda tak:*

AAY IRNLMY OO DHUOJIPSEE AIPMIF WKT MTOSKKLV H AYKZIVE ARKI DKWOPTE.

Zatem Twój program powinien wypisać:

AR HPAF KHZADT...

Zadanie domowe 2 *Narysuj wykres częstości występowania poszczególnych liter w tekście będącym wynikiem zadania 1 (czyli w co piątym znaku pliku* `vig5.txt`*). Wiedząc, że wykres ten odpowiada przesuniętemu wykresowi częstości występowania liter w naszym języku (lista* `czestosci`*), znajdź to przesunięcie.*

Zadanie domowe 3 *Plik* `vig5.txt` *powstał w wyniku zaszyfrowania pewnego tekstu napisanego w języku polskim szyfrem Vigenere'a z pewnym pięcioliterowym kluczem. Znajdź ten klucz, a następnie rozszyfruj tekst.*

Wskazówka: szyfr Vigenere'a z pięcioliterowym kluczem możemy traktować jak 5 szyfrów Cezara z 5 różnymi kluczami. W zadaniu 2 odkryliśmy pierwszą literę klucza — znalezione przesunięcie to indeks w alfabecie tej litery. Aby znaleźć drugą literę klucza, przeprowadź analogiczną procedurę dla co piątej litery tekstu począwszy od drugiej, itd.