# Ochrona Danych

# Laboratorium 5

## 1. Użyj tekstów jawnych do wykonania analizy statystycznej

Użyłem skryptu Powershell-a to wykonania analizy statystycznej. Skrypt wczytuje do zmiennej zawartość pliku poleceniem *[System.IO.File]::ReadAllBytes()* i grupuje pojedyncze bajty poleceniem *Group-Object*. Oprócz samego znaku pokazana jest też jego wartość kodowa w kolumnie ‘Kod’. Użyłem sortowania według ilości wystąpień.

|  |  |  |  |
| --- | --- | --- | --- |
| Tekst\_jawny\_0.txt | szyfrogram\_f\_0.txt | Tekst\_jawny\_1.txt | szyfrogram\_f\_1.txt |
| Znak Kod Ilosc  ---- --- -----  32 2424  a 97 1139  e 101 1102  i 105 1092  o 111 957  z 122 750  n 110 649  y 121 641  t 116 574  w 119 553  s 115 540  c 99 502  m 109 463  r 114 458  d 100 432  p 112 356  k 107 338  ³ 179 323  l 108 296  j 106 266  , 44 236  u 117 235  . 46 199  ¿ 191 179  g 103 178  ê 234 177  b 98 170  ¹ 185 132  156 115  h 104 95  - 45 95  ó 243 91  æ 230 75  ... 13 55  ... 10 55  T 84 41  N 78 32  ? 63 28  Z 90 24  f 102 21  P 80 20  B 66 5  G 71 4  I 73 4  H 72 4  E 69 3  R 82 3  U 85 2  ! 33 2  L 76 2  1 49 2  \t 9 1  140 1  F 70 1  ¯ 175 1  ( 40 1  0 48 1  2 50 1  ) 41 1  7 55 1  £ 163 1 | Znak Kod Ilosc  ---- --- -----  X 88 1350  " 34 642  ð 240 619  ¾ 190 608  Ù 217 493  ¨ 168 412  /t 9 366  16 339  » 187 312  150 306  f 102 304  Ì 204 281  c 99 248  Ñ 209 233  137 218  154 197  Æ 198 193  Õ 213 174  ... 10 169  17 137  ª 170 133  z 122 123  ó 243 114  + 43 112  W 87 99  B 66 96  4 89  H 72 67  18 65  6 54  å 229 53  19 47  ` 96 41  ] 93 27  ò 242 25  5 25  25 15  21 14  ¤ 164 13  × 215 12  6 54 12  d 100 11  r 114 11  ù 249 11  è 232 8  Ý 221 8  § 167 8  a 97 7  V 86 7  141 5  æ 230 5  þ 254 4  - 45 3  22 3  s 115 3  ) 41 2  2 1  T 84 1  . 46 1  P 80 1  15 1  20 1 | Znak Kod Ilosc  ---- --- -----  32 5113  a 97 2765  i 105 2386  e 101 2313  o 111 2062  z 122 1773  n 110 1592  c 99 1159  w 119 1157  r 114 1149  s 115 1144  y 121 1137  t 116 1075  m 109 1025  d 100 905  k 107 874  ³ 179 866  p 112 777  j 106 653  u 117 625  l 108 568  , 44 507  ê 234 424  b 98 389  g 103 367  . 46 352  ¹ 185 351  h 104 257  ¿ 191 254  156 231  ó 243 221  æ 230 135  ... 10 107  ... 13 107  150 90  P 80 60  f 102 59  N 78 44  ñ 241 40  W 87 39  133 11  U 85 10  A 65 10  L 76 10  E 69 8  I 73 8  R 82 6  2 50 3  0 48 3  132 3  ¯ 175 3  148 3  " 34 2  ! 33 1  - 45 1  F 70 1  ; 59 1  H 72 1  9 57 1  G 71 1  1 49 1  ( 40 1  ) 41 1 | Znak Kod Ilosc  ---- --- -----  X 88 2818  ¾ 190 1564  " 34 1324  ð 240 1309  Ù 217 1149  ¨ 168 942  16 939  f 102 651  \t 9 651  137 648  » 187 631  Ì 204 619  150 612  c 99 586  Ñ 209 564  154 514  ... 10 455  Æ 198 422  17 397  ó 243 362  Õ 213 360  ª 170 271  B 66 231  18 219  + 43 218  z 122 201  W 87 199  4 153  å 229 146  6 142  H 72 117  ` 96 69  5 47  ò 242 47  r 114 34  Ö 214 28  ¤ 164 27  ] 93 26  6 54 24  × 215 22  a 97 21  25 20  s 115 18  - 45 17  V 86 15  Ý 221 12  141 12  è 232 11  ê 234 9  d 100 7  ù 249 7  § 167 6  æ 230 4  15 4  4 52 4  21 4  T 84 3  31 3  E 69 3  P 80 2  . 46 2  Ó 211 2  þ 254 2  / 47 1  139 1  142 1  29 1  ) 41 1  26 1  2 1  19 1  30 1  A 65 1 |

Ciekawostką jest przesunięcie w trzecim od końca wierszu. Znak występuje tylko raz i ma kod 9. Po spojrzeniu w tablicę kodów ASCI dowiadujemy, że jest to znak tabulacji. To tłumaczy to przesunięcie.

Zobaczmy czy częstości znaków są pradopodobne. Poniżej tabela z najczęstszymi literami występującymi w polskich tekstach (według [Wikipedii](https://pl.wikipedia.org/wiki/Alfabet_polski#Cz%C4%99sto%C5%9B%C4%87_wyst%C4%99powania_liter)):

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **a** | 8,91% | | **i** | 8,21% | | **o** | 7,75% | | **e** | 7,66% | | **z** | 5,64% | | **n** | 5,52% | | **r** | 4,69% | | **w** | 4,65% | | **s** | 4,32% | | **t** | 3,98% | | **c** | 3,96% | | **y** | 3,76% | | **k** | 3,51% | | **d** | 3,25% | | **p** | 3,13% | | |  |  | | --- | --- | | **m** | 2,80% | | **u** | 2,50% | | **j** | 2,28% | | **l** | 2,10% | | **ł** | 1,82% | | **b** | 1,47% | | **g** | 1,42% | | **ę** | 1,11% | | **h** | 1,08% | | **ą** | 0,99% | | **ó** | 0,85% | | **ż** | 0,83% | | **ś** | 0,66% | | **ć** | 0,40% | | **f** | 0,30% | | |  |  | | --- | --- | | **ń** | 0,20% | | **q** | 0,14% | | **ź** | 0,06% | | **v** | 0,04% | | **x** | 0,02% | |

Zobaczmy 10 najczęściej występujących znaków z powyższego źródła i zestawmy je z tekstami jawnymi (z pominięciem spacji):

|  |  |
| --- | --- |
| Wikipedia | aioeznrwst |
| Tekst\_jawny\_0.txt | aeioznytws |
| Tekst\_jawny\_1.txt | aieozncwrs |

Wyniki są prawdopodobne – najpopularniejszym znakiem jest **a**. Pozostałe znaki są identyczne i różni je jedynie nieznacznie kolejność. Jedynie różnice wydają się powiększać dla ostatnich czterech znaków, gdzie nie wszędzie mamy znaki **c**,**r** i **y**, ale jak spojrzymy na pełną tabele z Wikipedii to widzimy, że znaki **c**, **y** znajdują się na pozycjach 11 i 12.

Teraz przygotujemy tekst odszyfrowany. Na początek wpiszemy tablicę do odkowania na podstawie 10 najpopularniejszych znaków. Zaczynamy od pierwszego tekstu.

|  |  |
| --- | --- |
| Tekst\_jawny\_0.txt | szyfrogram\_f\_0.txt |
| Znak Kod Ilosc  ---- --- -----  32 2424  a 97 1139  e 101 1102  i 105 1092  o 111 957  z 122 750  n 110 649  y 121 641  t 116 574  w 119 553 | Znak Kod Ilosc  ---- --- -----  X 88 1350  " 34 642  ð 240 619  ¾ 190 608  Ù 217 493  ¨ 168 412  /t 9 366  16 339  » 187 312  150 306 |

Przykład zmiennej typu słownikowego w Powershell:

$alfabet = New-Object -TypeName 'System.Collections.Generic.Dictionary[[char], [char]]'

$alfabet = @{

[char]'X' = ' ';

[char]'"' = 'a';

[char]240 = 'e';

[char]190 = 'i';

[char]217 = 'o';

[char]168 = 'z';

[char]9 = 'n';

[char]16 = 'y';

[char]187 = 't';

[char]150 = 'w';

}

Zobaczmy jak wygląda nasz tekst po takim przetłumaczeniu. Znaki przetłumaczone są zaznaczone na biało, pozostały tekst na szaro (próba A1):

sšo‰o t–iec ta+ cacofoÕync HfaiÑšaec –e\_ ‰ozcofn \_ t–fae‰Ñza \_ –o Æofaec fic \_eÑyoz Ýet–eHcn \_ó\_ WÕatšo a ÑÕi–eBo yi\_fi\_yae\_tze \_et–ª WnHcn Æo Æ‰ot–ó \_nfa ÑoÌzešiÕa šoaÌiz ×iHyae zišoaÌznec ót–ifaiyae iy–eyn a Æ‰oB‰icofiyae ecat\_az 6i otaecyiHÌae BoÑzay ziÌzyaecn yiÑifi` tnByi yi 6aeca+z d šaeÑn Bo oÑWao‰\_ª –o yife– \_iš ta+ –ó ftznt–šo zeÆtó\_eª iÕe cn Æ‰ze\_n\_ecnª –o Æ‰znÕeÌ\_ a yit HÌa\_By\_zò\_\_ × Æo‰z\_Ñšó \_ tšfa–ofi Ñof\_ÑÌiª i zf‰iÌi\_\_Ì ta+ Ño ]ociª t–fae‰Ñza \_ \_eHÕa Ño šoaÌi Ñn\_ó‰ó yae zcaeya+ ÑeÌnz\_aª –o ‰\_Wª \_iš Æot–iyofaeHzò\_Ñf‰\_Ìa ta+ a Wez tofi fntzeÑ z ÆocaetzÌzeyaizò\_\_ \_o–\_f \_et–ec ta+ zio\_n`ª \_e ÆotzeÑ

Trudno wyłapać prawidłowe słowa, widać dziwną sekwencje wyrazów **aiyae**. To nam sugeruje, że słownik, którego użyliśmy nie jest do końca poprawny.

Analogicznie, dla tekstu drugiego (próba B1):

þ\_oBeša 4neci‰»ócþ \_ \_§A êoåÑan szwcÌzaš \_E\_\_ª\_E\_Eò\_\_óWecw ‰ozca\_a` aÕ–i‰na–wcnw Æ‰ziWeiB zÑa‰ziaª Ìza»ace c–iÑwª BÑw ‰ziÌzwce»–oH`ª c š–\_‰i\_ \_w\_icwª \_i»– ÑaÕiša oÑ na»zwÌå oÌzišecaaª a Ìza»ace z ‰aÌ\_e na»zi\_ ÑoÌeišÕeci\_ na–ó‰wz èÕaÌziBo åe»–o‰ea Æo–oÌzwa »e+ c znanwc nac šei‰ónšó e Ìzw Www \_ašeiH cwÑa‰zineaª š–\_‰wÌå za\_HÌei ÕóW ónešne+Ìei coBo ‰aÑwšaÕnei zceine` Õo»w ÕóÑzeª š‰a\_óª Hcea–a\_ \_a –i Æw–anea nei ca cea‰wBoÑni\_ oÑÆoceiÑzeª aÕi o–cei‰a\_\_ oni Ñ‰oB+ Ño »nóÌea ‰ozca\_aaª coB\_ÌwÌå »–anoce` Ño»šonai za\_+Ìei ócw»ó e nei cnei\_ Æ‰zw\_icn\_ ‰oz‰wcš+z §o oÌzwce»–iª o–cei‰a\_\_ ÆoÕi Ño Ñw»šó»\_eª BÑw\_ zÑaa na ÆoÑanw –ica– co\_i Ww` naci– –wÕiª eÕó \_i»– Ñw»šó–an–\_czò\_è‰oBa ÆoÆ‰aceaa »e+ z ša\_Ñwc Æ‰zi\_iÌåanwc šeÕoci–‰icz \_ei cea »e+ \_ó\_ zaW\_\_Ìzwce zaš‰+–aceª aÕi aBoÑnwc »ÆaÑšeic

Także widać dziwne, niewystępujące w polskim języku sekwencje liter, np. **aceaa**.

## 2. Podejmij kolejne kroki prowadzące do deszyfracji tekstu

Pomyślałem, że warto by było się upewnić, że słowa które widzimy mają dokładnie tyle znaków ile w oryginalnym tekście. Czyli powinniśmy znać pozycję wszystkich znaków określających granicę słów: spacji, tabulacji i znaków przejścia do nowej linii. O ile spacje można założyć, że znajdują się tam gdzie powinny – występują ok. dwa razy częściej niż najpopularniejsza litera ‘a’ – to ze znakami nowego wiersza i tabulajcją już tak nie jest. Uznałem, że najłtawiej będzie znaleźć przejście do nowego wiersza. W systemach Windows jest ono realizowane za pomocą dwuch znaków CR (Carriage Return) o kodzie 13 oraz LF (Line Feed) o kodzie 10. Napisałem więc kolejny skrypt, tym razem pokazujący najczęściej występujące kombinacje dwóch znaków występujących koło siebie:

|  |  |
| --- | --- |
| Tekst\_jawny\_0.txt | Tekst\_jawny\_1.txt |
| Name Count  ---- -----  ie 439  a 217  S 182  Z 163  cz 162  zy 133  st 133  Po 130  za 128  em 123  Sz 121  ow 111  ta 100  Zi 96  mi 95  - 94  ia 93  ak 71  an 64  ym 61  ... 55  - 52  k 49  da 39  ró 29  wr 21  us 16  n 15  Pu 12 | Name Count  ---- -----  ni 670  a 584  S 469  cz 371  za 282  zy 257  Sz 247  an 174  B 158  ka 147  ak 137  er 122  ym 121  U 117  ... 107  k 79  da 79  we 59  Lo 47  ik 45  iw 42  og 33  Bo 32  gi 27  um 25  su 24  n 21  Un 21  rs 11 |

Niestety okazało się to mało skuteczne. Powershell zapisuje oba znaki oraz ich kombinację, jako …, prawdopodonie dlatego, aby zmieścić je w tabeli. Nie zaimplementowałem wyświetlania kodów danych sekwencji dwóch znaków, ale poprzez porównanie częstości z pierwszą tabelą możemy zauważyć, gdzie są nasze kombinacje CR LF. W pierwszym tekście ta kombinacja występuje 55 razy, w drugim 107. Jest to odpowiednio 21 i 15 najpopularniejszy dwuznak. Daleko niżej niż się spodziewałem.

Ale z drugiej strony to dzięki tym tabelom wiemy teraz, że nasze wątpliwości dotyczące długości wyrazów były nie do końca uzasadnione. Widzimy że znaków CR LF jest o wiele mniej niż spacji, dlatego też zdecydowana większość słów z naszego nie do końca odszyfrowanego tekstu będzie miała taką samą długość jak oryginalny tekst.

W związku z tym, że możemy się spodziewać przygotowane słowniki są w miare dobre, a mimo to nie możemy jeszcze nic z nich odczytać, to proponuję rozważyć dwa podejścia:

1. Ręcznie modyfikowanie słownika aż nie pojawią się jakieś rozpoznawalne wyrazy.
2. Automatyczne modyfikowanie słownika i przygotowanie odkodowanych tekstów. Posortowanie ich według jakości.

Co do metody 1. jest trudne jeśli nie niemożliwe opisanie jakimi kryteriami kieruje się ludzki umysł przy próbie wyłapania właściwych słów i modyfikacji słownika. Ręczna modyfikacja nie da nam też gwarancji, że nie zatoczymy kółka, czyli że nie będziemy powtarzać poprzednich konfiguracji słownika. Nie wymaga ona natomiast żadnej implementacji.

A metoda 2. wymaga oczywiście przygotowania algorytmu, ale można go tak zaprojektować, aby testowane słowniki się nie powtarzały. Możemy wybrać listę najpopularniejszych polskich słów (np. tutaj: [Najpopularniejsze słowa 1-1000 wersja Jerzego Kazojcia](https://pl.wiktionary.org/wiki/Indeks:Polski_-_Najpopularniejsze_s%C5%82owa_1-1000_wersja_Jerzego_Kazojcia)) i sprawdzać, czy przetłumaczony tekst zawiera któreś słowa z tej listy. Im więcej słów będzie pasować tym większy wynik punktowy przyznamy danemu tekstowi.

Za każdą literę w pełni odgadniętym słowie możemy dodać np. 3 punkty. W przypadku niepełnych trafień sprawa się komplikuje, gdyż może się zdarzyć, że nasze nie wpełni przetłumaczone słowo – nazwijmy je „wzór” – będzie pasować do kilku z najpopularniejszych. Wtedy proponuję przyznać 1 punkt za każdą trafioną literę. Założymy, że nasze wzór to słowo, które ma najwięcej trafionych liter.

Poniżej poglądowy algorytm przedstawiony graficznie obrazujący co się dzieje dla każdego słowa z przetłumaczonego tekstu. Gdy przejdziemy przez wszystkie słowa z listy 1000 to wybierzemy najwyższą wartość z punktacji dla danego słowa przetłumaczonego tekstu. Gdy podczas sprawdzania warunek (ilość punktów = długość słowa) będzie prawdziwy to kończymy pętlę. Ten warunek oczywiście oznacza, że nasze przetłumaczone słowo istnieje w słowniku i nie ma wtedy potrzeby sprawdzania kolejnych słów.

