ZAKRES MATERIAŁU

I. **Entropia**
II. **Kodowania Huffmana a entropia**

I. Entropia

[DEF 01] (Entropia)
Niech X – zmienna losowa przyjmująca skończenie wiele wartości zgodnie z rozkładem prawdopodobieństwa p(X). Entropię tego rozkładu definiujemy

$$H(x) = -\sum_{i=1}^{n} p_i \log_2 p_i \quad (\text{dla } p_i \neq 0)$$

[Przykład 01] Entropia jednego rzutu symetryczną kostką do gry jest równa 1, ponieważ informację, że wypadło 1/2/3/4/5/6 oczek można przekazać za pomocą jednej jednostki informacji.

[Przykład 02] Entropia n niezależnych rzutów symetryczną kostką do gry jest równa n, ponieważ informację o n rzutach kostką można przekazać za pomocą n jednostek informacji.

[Przykład 03] X jest zmienną losową przyjmującą wartości ze zbioru A = {a, b, c, d} z prawdopodobieństwami p(a)=1/4, p(b)=1/2, p(c)=1/8, p(d)=1/8. Policz entropię związaną z wylosowaniem jednej wartości ze zbioru A.

$$H(x) = -\frac{1}{4}\log_2\frac{1}{4} - \frac{1}{2}\log_2\frac{1}{2} - \frac{1}{8}\log_2\frac{1}{8} - \frac{1}{8}\log_2\frac{1}{8} = \frac{1}{4}\cdot 2 + \frac{1}{2}\cdot 1 + \frac{1}{8}\cdot 3 + \frac{1}{8}\cdot 3 = \frac{7}{4}$$

Entropia wylosowania jednej wartości ze zbioru A wynosi 7/4.

Entropia składników systemu kryptograficznego.
Możemy policzyć

- entropię H(K), gdzie K jest zmienną losową (reprezentującą klucz) przyjmującą wartości zgodnie z rozkładem prawdopodobieństwa p_K,
- entropię H(P), gdzie P jest zmienną losową (reprezentującą przestrzeń tekstu jawnego),
- entropię H(C), gdzie C jest zmienną losową (reprezentującą przestrzeń tekstu zaszyfrowanego)

[Przykład 04] Niech $P = \{a, b, c\}$ będzie przestrzenią tekstów jawnych oraz niech prawdopodobieństwa a priori wystąpienia poszczególnych znaków tego zbioru będą równe $p_P(a) = 1/4$ i $p_P(b) = 1/2$, $p_P(c) = 1/4$. Entropię H(P) liczymy

$$H(P) = -\frac{1}{4}\log_2\frac{1}{4} - \frac{1}{2}\log_2\frac{1}{2} - \frac{1}{4}\log_2\frac{1}{4} = \frac{1}{4}\log_2 4 + \frac{1}{2}\log_2 2 + \frac{1}{4}\log_2 4 = \frac{1}{4}*2 + \frac{1}{2}*1 + \frac{1}{4}*2 = \frac{3}{2}$$

II. Kodowania Huffmana a entropia

[DEF 02] (kodowanie)
Niech X – zmienna losowa przyjmująca wartości ze skończonego zbioru zgodnie z rozkładem prawdopodobieństwa p(X).
Kodowanie zmiennej losowej X – przekształcenie $f : X \to \{0,1\}^*$.
$\{0,1\}^*$ - zbiór wszystkich skończonych ciągów 0-1

Niech $x_1, ..., x_n$ będzie ciągiem zdarzeń.
Kodowaniem f nazywamy (też) funkcję $f(x_1,...,x_n) = f(x_1)|...|f(x_n)$, gdzie
Znak | oznacza konkatenację ciągów.

[Własności] (kodowania f)

- Funkcji f powinna być **przekształceniem różnowartościowym**, ponieważ tylko w takim przypadku zaszyfrowany (skompresowany / zakodowany) tekst da się odszyfrować (zdekompresować / zdekodować).
- Funkcja f musi być **funkcją prefiksową** (inaczej mówimy **funkcją wolną od przedrostków**) tzn. nie może zdarzyć się sytuacja, że dla dwóch elementów $a, b \in X$ i ciągu $x \in \{0,1\}^*$ zachodzi $f(a) = f(b)|x$ (takie sytuacje powodowałyby niejasności przy dekodowaniu).

[Przykład 05] Niech eksperyment polega na n-krotnym rzucie monetą
Niech $X = \{O, R\}$ (O – orzeł, R - reszka)

- $f(O) = 01$, $f(R) = 0110$ - f nie jest funkcją prefiksową
  W takiej sytuacji (np. podczas rozkodowania ciągu 01101...) nie ma jasności jaki jest pierwszy znak (O czy R).
- $f(O) = 01$, $g(R) = 11$ - g jest funkcją prefiksową
  W takiej sytuacji zawsze mamy jednoznaczną odpowiedź, jaki jest kolejny znak (O czy R).

[DEF 03] (średnia długość kodowania elementu X)
Miara skuteczności kodowania f definiowana następująco

$$\ell(f) = \sum_{x \in X} p(x)\,|f(x)|$$

|y| -ozn. długość ciągu y.

- Dąży się do wyszukania takiego różnowartościowego kodowania f, które minimalizuje wartość $\ell(f)$.

[Algorytm] (kompresji i dekompresji Huffmana)

- Zachłanny algorytm Huffmana generuje kodowanie f, które
  - jest różnowartościowe
  - prefiksowe
  - spełnia nierówności $H(X) \le \ell(f) < H(X) + 1$ (entropia przybliża średnią długość kodowania Huffmana)

[Idea] (algorytmu kompresji Huffmana)
KOMPRESJA
(1) Wyznaczamy tablicę częstości wystąpień poszczególnych znaków w tekście i na jej podstawie określamy rozkład prawdopodobieństwa na zbiorze X możliwych znaków.
(2) Generujemy uporządkowaną niemalejąco (ze względu na wartość prawdopodobieństwa) listę (np. ArrayList) jednowęzłowych drzew, których węzły zawierają następujące pola (klucz (prawdopodobieństwo wystąpienia), znak (lub '_' w przypadku węzłów wewnętrznych), referencje do lewego i prawego poddrzewa)
(3) Na podstawie listy tworzymy drzewo Huffmana. W tym celu, w każdej iteracji bierzemy z listy dwa drzewa L1 i L2 o najmniejszych prawdopodobieństwach w korzeniach i łączymy je w jedno drzewo. Kluczem korzenia powstałego drzewa jest suma prawdopodobieństw korzeni łączonych drzew. Jego lewym poddrzewem jest L1, a prawym – L2. Nowe drzewo umieszczamy na liście w taki sposób, by nadal była uporządkowana niemalejąco. Proces kończymy w momencie uzyskania jednego drzewa, które ma tę własność, że węzły wewnętrzne zawierają jedynie prawdopodobieństwa, a liście dodatkowo znaki występujące w kompresowanym tekście.
(4) Generujemy kody Huffmana. Każdą lewą krawędź drzewa oznaczamy znakiem '0', a prawą – '1'. Czytamy każdą ze ścieżek drzewa zaczynając od korzenia i kończąc na liściu i zapisujemy występujące na ścieżce zera i

jedynki. Powstałe ciągi 0-1-kowe stanowią kody znaków występujących w liściach. Znaki i ich kody zapisujemy w strukturze HashMap (znak w roli klucza i kod jako obiekt).

(5) Kodujemy tekst korzystając z wygenerowanych kodów, a następnie każdy 0-1-kowy 8-elementowy podciąg zamieniamy na znak (zgodnie z tablicą kodów ASCII). Jeśli ostatni ciąg składa się z mniejszej niż osiem liczby znaków, dopełniamy go zerami.

(6) Do pliku wyjściowego wklejamy tablicę częstości wystąpień poszczególnych znaków plus tekst w postaci zakodowanej.

DEKOMPRESJA

(1) Na podstawie informacji zawartej w skompresowanym pliku generujemy tablicę częstości wystąpień znaków.

(2) Tworzymy listę drzew jednowęzłowych i na jej podstawie generujemy drzewo Huffmana.

(3) Zamieniamy zakodowany tekst na ciąg 0-1-kowy (każdy znak tekstu to wartość z zakresu 0-255, którą zapisujemy w postaci binarnej)

(4) Na podstawie tablicy częstości określamy ile znaków należy rozkodować. Czytamy kolejne zera i jedynki ciągu i na tej podstawie poruszamy się odpowiednio w lewo lub w prawo na drzewie Huffmana. Po rozkodowaniu fragmentu ciągu 0-1 (tj. po przypisaniu mu odpowiedniego znaku znajdującego się w liściu drzewa) rozpoczynamy dekodowanie kolejnego (startując na nowo od korzenia drzewa).

[Zadanie 01] Zastosuj algorytm kompresji i dekompresji Huffmana dla poniższego tekstu.

ala ma kota

KOMPRESJA

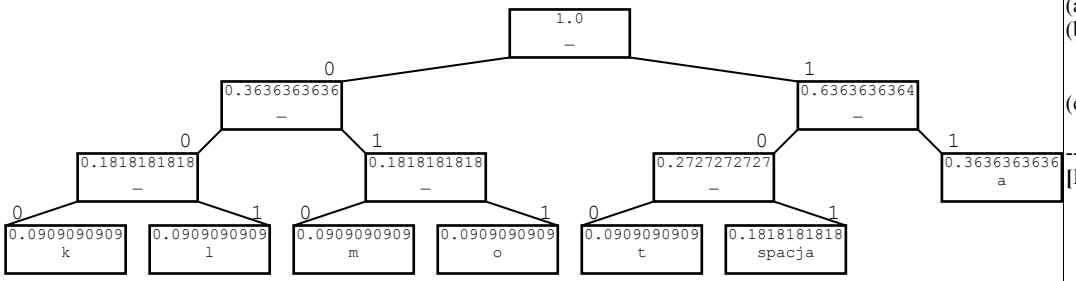
(1) tablica częstości wystąpień poszczególnych znaków w tekście

znak	a	l	spacja	m	k	o	t
częstość	4	1	2	1	1	1	1

(2) uporządkowana niemalejąco lista drzew jednowęzłowych

k	l	m	o	t	spacja	a
0.0909090909	0.0909090909	0.0909090909	0.0909090909	0.0909090909	0.1818181818	0.3636363636

(3) drzewo kodów Huffmana



(4) kody Huffmana

```
{spacja=101, o=011, k=000, m=010, a=11, t=100, l=001}
```

(5) zakodowany tekst z podziałem na podciągi 8-znakowe

```
a l  a _    m  a _    k  o    t  a
11001111  01010111  01000011  10011000        czerwone zera dopełniają ostatnią ósemkę
```

(6) zawartość skompresowanego pliku (tablica częstości i zakodowany tekst)

```
a-4,l-1,_-2,m-1,k-1,o-1,t-1
gUh9@
```

DEKOMPRESJA

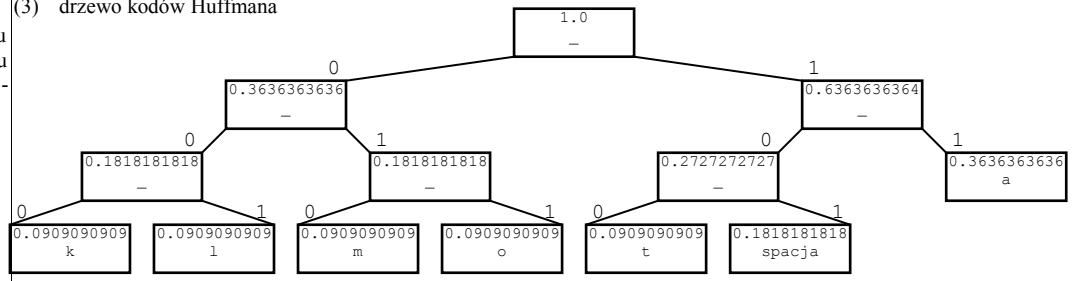
(1) tablica częstości wystąpień poszczególnych znaków w tekście

znak	a	l	spacja	m	k	o	t
częstość	4	1	2	1	1	1	1

(2) uporządkowana niemalejąco lista drzew jednowęzłowych

k	l	m	o	t	spacja	a
0.0909090909	0.0909090909	0.0909090909	0.0909090909	0.0909090909	0.1818181818	0.3636363636

(3) drzewo kodów Huffmana



(4) zakodowany tekst z przypisanymi znakami

```
11|001|11|101|010|11|101|000|011|100|11|000        czerwone zera dopełniają ostatnią ósemkę
a  l  a  _  m  a  _  k  o  t  a
```

[Zadanie 02] Zaimplementuj

(a) algorytm kompresji i dekompresji Huffmana. Dla ułatwienia, koduj 7-elementowe ciągi 0-1.

(b) algorytm wyznaczający entropię H(P), gdzie P jest zmienną losową (reprezentującą przestrzeń tekstu jawnego). Prawdopodobieństwa wystąpień poszczególnych znaków w tekstach polskich znajdują się w materiałach do jednych z wcześniejszych zajęć.

(c) algorytm wyznaczający średnią długość kodowania znaku dla konkretnego tekstu
Porównaj wartości z podpunktu (b) i (c) dla tekstu „ala ma kota"

[Przykład 06] Wyznacz średnią długość kodowania znaku algorytmem Huffmana dla tekstu "ala ma kota"

znak	a	l	spacja	m	k	o	t
częstość	4	1	2	1	1	1	1
p-stwo	0.3636363636	0.0909090909	0.1818181818	0.0909090909	0.0909090909	0.0909090909	0.0909090909
dł. kodu	2	3	3	3	3	3	3

$$\ell(f) = \sum_{x \in X} p(x)\,|f(x)|$$

$$\ell(f) =$$

$$= p('a')\,|f('a')| + p('l')\,|f('l')| + p('\_')\,|f('\_')| + p('m')'f('m')| + p('k')\,|f('k')| + p('o')\,|f('o')| + p('t')\,|f('t')| =$$

$$=0.3636363636*2+5(0.0909090909*3)+0.1818181818*3=$$

$$=0.7272727273+5*0.272727272+0.545454545=0.7272727273+1.36363636+0.545454545=2.636363632$$

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY
Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

---

[Zadanie 03] Skompresuj i zdekompresuj metodą Huffmana poniżej zapisane teksty. Podaj wyniki kolejnych etapów kompresji i dekompresji.

(a) W tekście zaszyfrowanym szukamy par identycznych fragmentów o długości co najmniej trzech znaków. Zapamiętujemy odległości między pierwszym znakiem pierwszego fragmentu i pierwszymi znakami kolejnych fragmentów. Oznaczamy te odległości odpowiednio przez d1, d2, ... Mając wartości d1, d2, ..., zakładamy, że m dzieli największy wspólny dzielnik tych liczb.

(b) ala ma kota

REALIZACJA

```
//-----------------------------------------------------------
Huffman h = new Huffman();

  System.out.println(h.kompresja(
   "W tekście zaszyfrowanym szukamy par identycznych fragmentów o długości co najmniej trzech znaków. Zapamiętujemy odległości m"+
   "iędzy pierwszym znakiem pierwszego fragmentu i pierwszymi znakami kolejnych fragmentów. Oznaczamy te odległości odpowiednio "+
   "przez d1, d2, .... Mając wartości d1, d2, ..., zakładamy, że m dzieli największy wspólny dzielnik tych liczb."));
  System.out.println(h.dekompresja(
   "0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,3,4,0,4,5,0,1,0,0,0,0,0,0,0,0,0,0,46,0,0,0,0,0,0,0,0,0,0,0,0,6,0,10,0,0,2,2,0,0,0,0,0,0,0,0,0,0,0,"+
   "0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,0,22,1,14,14,23,4,7,4,24,6,9,7,16,17,14,8,0,11,7,10,4,"+
   "0,11,0,15,21,0,0,0,0,0,\nŚ\"41}.Q-m>®wć1=|vQF7~@Ęi=hI{%ĄZ´§ÓlßQń¶ydlęl|ŻUf\\B6ŻJuFńrk!1$c¨=®;0¶}rńŃ¶?ts>ULĄu9}r6_ Ą'Żm>~śL5"+
   "kx«ig{-3r©9Yß\"$4\\§ts}WonśL3?87Ś+°=PXłfg}°ęo\"ą*&i-ßQŁN7jYBp~WÓląŻ:lWd¦eRfy<ó1lł©ęDÓŃ®{({ÓOu9^/©l;©!©0E1AŃ6ßlę?)|m7."+
   "W©l/^!]Whß0¨t/&k:t+ŁóvBg+i9$¤"));

  System.out.println(h.kompresja("ala ma kota"));
  System.out.println(h.dekompresja(
   "0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,"+
   "0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,"+
   "0,0,0,0,\ngUh9@"));

//-----------------------------------------------------------
```

WYNIKI

```
//-----------------------------------------------------------
-KOMPRESJA----------------------------------------------
(1)  lista drzew jednowęzłowych ze znakami i prawdopodobieństwami ich występowania w tekście
ą-0.002801120448179271 7,
ż-0.0028011204481792717,
M-0.0028011204481792717,
O-0.0028011204481792717,
W-0.0028011204481792717,
Z-0.0028011204481792717,
b-0.0028011204481792717,
1-0.005602240896358543 5,
2-0.0056022408963585435,
ę-0.008403361344537815,
ł-0.011204481792717087,
ó-0.011204481792717087,
f-0.011204481792717087,
h-0.011204481792717087,
u-0.011204481792717087,
ś-0.014005602240896359,
,-0.01680672268907563,
j-0.01680672268907563,
g-0.0196078431372549,
l-0.0196078431372549,
s-0.0196078431372549,
p-0.022408963585434174,
k-0.025210084033613446,
.-0.028011204481792718,
t-0.028011204481792718,
r-0.03081232492997199,
w-0.03081232492997199,
c-0.0392156862745098,
d-0.0392156862745098,
o-0.0392156862745098,
y-0.04201680672268908,
m-0.04481792717086835,
n-0.047619047619047616,
z-0.058823529411764705,
a-0.06162464985994398,
e-0.06442577030812324,
i-0.06722689075630252,
 -0.12885154061624648,

(2)  drzewo Huffmana przeczytane w porządku KLP
_-1.0, _-0.42577030812324934, _-0.1876750700280112, _-0.0896358543417367, _-0.04481792717086835,
p-0.022408963585434174, _-0.022408963585434174, ł-0.011204481792717087, ó-0.011204481792717087,
_-0.04481792717086835, _-0.022408963585434174, f-0.011204481792717087, h-0.011204481792717087,
_-0.022408963585434174, u-0.011204481792717087, _-0.011204481792717087, 2-0.0056022408963585435,
_-0.0056022408963585435, ą-0.0028011204481792717, ż-0.0028011204481792717, _-0.09803921568627451,
n-0.047619047619047616, _-0.05042016806722689, k-0.025210084033613446, _-0.025210084033613446,
_-0.011204481792717087, _-0.0056022408963585435, M-0.0028011204481792717, O-0.0028011204481792717,
_-0.0056022408963585435, W-0.0028011204481792717, Z-0.0028011204481792717, ś-0.014005602240896359,
_-0.2380952380952381, _-0.11484593837535015, _-0.056022408963585436, .-0.028011204481792718,
t-0.028011204481792718, z-0.058823529411764705, _-0.12324929971988796, a-0.06162464985994398,
_-0.06162464985994398, r-0.03081232492997199, w-0.03081232492997199,_-0.5742296918767507,
_-0.2605042016806722 3,  -0.12885154061624648,_-0.13165266106442575, e-0.06442577030812324,
i-0.06722689075630252, _-0.3137254901960784, _-0.1484593837535014, _-0.0700280112044818,
_-0.03361344537815126, ,-0.01680672268907563, j-0.01680672268907563, _-0.03641456582633053,
_-0.01680672268907563, ę-0.008403361344537815, _-0.008403361344537815, b-0.0028011204481792717,
1-0.0056022408963585435, g-0.0196078431372549, _-0.0784313725490196, c-0.0392156862745098,
d-0.0392156862745098, _-0.16526610644257703, _-0.0784313725490196, o-0.0392156862745098,
_-0.0392156862745098, l-0.0196078431372549, s-0.0196078431372549, _-0.08683473389355742,
y-0.04201680672268908, m-0.04481792717086835,

(3)  kody znaków wygenerowane na podstawie drzewa Huffmana
{O=00111001, k=00110, c=11010, h=000101, W=00111010, p=00000, 1=11001011,  =100, i=1011, a=0110,
ż=00011111, u=000110, ó=000011, t=01001, j=110001, s=111011, ś=001111, o=11100, d=11011, ł=000010,
Z=00111011, z=0101, w=01111, r=01110, 2=0001110, l=111010, g=110011, b=11001010, ę=1100100, ,=110000,
m=11111, M=00111000, f=000100, y=11110, ą=00011110, .=01000, n=0010, e=1010}

(4)  zakodowany tekst za pomocą kodów Huffmana (w postaci ciągu 0-1-kowego i ciągu znaków)
0011101010001001101000110001111110101011101010001010110111011010111110000100011101110001110110001011110
11111100111011010100011000110011011111111101000000001100111010010111101110100010010011111011010010100101
1110110100001011000001000111001101100111111101000100100100001101111100111001001101100001000011011001111
1000001111110101011100110101110010000100110110001111110010101110101100011000100101110010110101101000010 11
0001010010011000110000011011110100010000111011011000000011011111101111001000100100011011000110101111111 1
10100111001101111101010101100110000101110000111111010101110011111101111001001101101011111010000000101110
100111001111110110101111101111110001010010011000110101110101111110000000101110100111001111110110101101
011001111100100000100011100110110011111111010001001001000110100101110000000101110100111001111111101101011
11101111110111000101001001100011001101111110111000011011100111010101011000100101111011010000101100000100
01110011011001111111101000100100100001101111010001000011100101010010011011010010101101111111111010001 0011
0101001110011011111010101011001100001011100001111110101011100111001101100000111000111110111010110110010 1
01111100100000000111001011010010110011011110010111100001001101100011101100001000100001000010000100010000
11100001101100010001111011010100011110110011100100111100001111110101011100110111100101111000010011011000
11101100001000100001000010001100001000101011000110000010011011011011011111111110110000100000111111010100 1
11111001101101011011101011101010111000010011011000101111101111001000011011101101011111010001111111011000
00000011111010001011110100110110101101110101110100010101100110100010011111011010000101100111010101111010
01011100101001000
Ś"41}.Q-m>®wć1=|vQF7~@Ęi=hI{%ĄZ´§ÓlßQń¶ydlęl|ŻUf\B6ŻJuFńrk!1$c¨=®;0¶}
rńŃ¶?ts>ULĄu9}r6_ Ą'Żm>~śL5kx«ig{-3r©9Yß"$4\§ts}WonśL3?87Ś+°=PXłfg}°
ęo"ą*&ißQŁN7jYBp~WÓląŻ:lWd¦eRfy<ó1lł©ęDÓŃ®{({ÓOu9^/©l;©!©0E1AŃ6ßlę?)
|m7.W©l/^!]Whß0¨t/&k:t+ŁóvBg+i9$¤
-WYNIK KOMPRESJI--------------------------------------
(5)  wynik kompresji - tablica częstości i zakodowany tekst
```


KAPITAŁ LUDZKI NARODOWA STRATEGIA SPÓJNOŚCI
UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY
Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

```
0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,3,4,0,4,5,0,1,0,0,0,0,0,0,0,0,0,46,0,0,0,0,0,0,0,0,0,0,6,0,10,0,0,2,2,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,22,1,14,14,2
3,4,7,4,24,6,9,7,16,17,14,8,0,11,7,10,4,0,11,0,15,21,0,0,0,0,0,
Ś"41}.Qm>®wć1=|vQF7~@Ęi=hI{%ĄZ´§ÓlßQń¶ydlęl|ŻUf\B6ŻJuFńrk!1$c¨=®;0¶}r
ńŃ¶?ts>ULĄu9}r6_ Ą'Żm>~śL5kx«ig{-3r©9Yß"$4\§ts}WonśL3?87Ś+°=PXłfg}°ę
o"ą*&ißQŁN7jYBp~WÓląŻ:lWd¦eRfy<ó1lł©ęDÓŃ®{({ÓOu9^/©l;©!©0E1AŃ6ßlę?)|
m7.W©l/^!]Whß0¨t/&k:t+ŁóvBg+i9$¤

-DEKOMPRESJA-----------------------------------------
(1)  tablica częstości wystąpień znaków w tekście
0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,3,4,0,4,5,0,1,0,0,0,0,0,0,0,0,0,46,0,0,0,0,0,0,0,0,0,0,6,0,10,0,0,2,2,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,22,1,14,14,2
3,4,7,4,24,6,9,7,16,17,14,8,0,11,7,10,4,0,11,0,15,21,0,0,0,0,0,

(2)  lista drzew jednowęzłowych ze znakami i prawdopodobieństwami ich występowania w tekście
ą-0.0028011204481792717,
ż-0.0028011204481792717,
M-0.0028011204481792717,
O-0.0028011204481792717,
W-0.0028011204481792717,
Z-0.0028011204481792717,
b-0.0028011204481792717,
1-0.0056022408963585435,
2-0.0056022408963585435,
ę-0.008403361344537815,
ł-0.011204481792717087,
ó-0.011204481792717087,
f-0.011204481792717087,
h-0.011204481792717087,
u-0.011204481792717087,
ś-0.014005602240896359,
,-0.01680672268907563,
j-0.01680672268907563,
g-0.0196078431372549,
l-0.0196078431372549,
s-0.0196078431372549,
p-0.022408963585434174,
k-0.025210084033613446,
.-0.028011204481792718,
t-0.028011204481792718,
r-0.03081232492997199,
w-0.03081232492997199,
c-0.0392156862745098,
d-0.0392156862745098,
o-0.0392156862745098,
y-0.04201680672268908,
m-0.04481792717086835,
n-0.047619047619047616,
z-0.058823529411764705,
a-0.06162464985994398,
e-0.06442577030812324,
i-0.06722689075630252,
-0.12885154061624648,

(3)  drzewo Huffmana przeczytane w porządku KLP
_-1.0, _-0.42577030812324934, _-0.1876750700280112, _-0.0896358543417367, _-0.04481792717086835,
p-0.022408963585434174, _-0.022408963585434174, ł-0.011204481792717087, ó-0.011204481792717087,
_-0.04481792717086835, _-0.022408963585434174, f-0.011204481792717087, h-0.011204481792717087,
_-0.022408963585434174, u-0.011204481792717087, _-0.011204481792717087, 2-0.0056022408963585435,
_-0.0056022408963585435, ą-0.0028011204481792717, ż-0.0028011204481792717, _-0.09803921568627451,
n-0.047619047619047616, _-0.05042016806722689, k-0.025210084033613446, _-0.025210084033613446,
_-0.011204481792717087, _-0.0056022408963585435, M-0.0028011204481792717, O-0.0028011204481792717,
_-0.0056022408963585435, W-0.0028011204481792717, Z-0.0028011204481792717, ś-0.014005602240896359,
_-0.2380952380952381, _-0.11484593837535015, _-0.056022408963585436, .-0.028011204481792718,
t-0.028011204481792718, z-0.058823529411764705, _-0.12324929971988796, a-0.06162464985994398,
_-0.06162464985994398, r-0.03081232492997199, w-0.03081232492997199, _-0.5742296918767507,
_-0.26050420168067223,  -0.12885154061624648, _-0.13165266106442575, e-0.06442577030812324,
i-0.06722689075630252, _-0.3137254901960784, _-0.1484593837535014, _-0.0700280112044818,
_-0.03361344537815126, ,-0.01680672268907563, j-0.01680672268907563, _-0.03641456582633053,
_-0.01680672268907563, ę-0.008403361344537815, _-0.008403361344537815, b-0.0028011204481792717,
1-0.0056022408963585435, g-0.0196078431372549, _-0.0784313725490196, c-0.0392156862745098,
d-0.0392156862745098, _-0.16526610644257703, _-0.0784313725490196, o-0.0392156862745098,
_-0.0392156862745098, l-0.0196078431372549, s-0.0196078431372549, _-0.08683473389355742,
y-0.04201680672268908, m-0.04481792717086835,

(4)  zakodowany tekst za pomocą kodów Huffmana (w postaci ciągu 0-1-kowego)
00111010100010011010001100011111101010111010100010101101110110101111100001000111011100011110110001011110
1111110011101101010001100011001101111111110100000000110011101001011110111010001001001111011010010100101
11101101000010110000010001110011011001111111101000100100100001101111100111001001101100001000011011001111
10000111111010101110011010111001000010011011000111111001010111010110001100010010111001011010110100001011
00010100100110001100000110111101000100001110110110000000110111111011110010001001000110110001101011111111
1010011100110111111010101011001100001011100000111111010101110011111101111001001101101011111010000000101110
1001110011111110110101111110111111000101001001100011010111010111111000000010111010011100111111110110101101
0110011111001000001000111001101100111111110100010010010001101001011100000000101110100111001111111101101011
11101111110111000101001001100011001101111110111000011011100111010101011000100101111011010000101100000100
0111001101100111111111010001001001000011011110100010000111001010100100110110100101011011111111110100010011
010100111001101111110101010110011000010111000011111101010111001110011011000001110001111101110101101100101
01111100100000000111001011010010110011011110010111100001001101100011101100001000100001000010000100010000
11100001101100010001111011010100011110110011100100111100001111110101011100110111100101111000010011011000
111011000010001000010000100011000010001010110001100000100110110110110111111111101100001000001111110101001
1111100110110101101110101110101011100001001101100010111110111100100001101110110101111101000111111011000
000000111110100010111101001101101011011101011101000101011001101000100111111011010000101100111010101111010
010111001010010000000000

-WYNIK DEKOMPRESJI-----------------------------------
(5)  wynik dekompresji - rozkodowany tekst
W tekście zaszyfrowanym szukamy par identycznych fragmentów o długości co najmniej trzech znaków. Zapa-
miętujemy odległości między pierwszym znakiem pierwszego fragmentu i pierwszymi znakami kolejnych frag-
mentów. Oznaczamy te odległości odpowiednio przez d1, d2, .... Mając wartości d1, d2, ..., zakładamy, że
m dzieli największy wspólny dzielnik tych liczb.
-KONIEC----------------------------------------------

-KOMPRESJA-------------------------------------------
(1)  lista drzew jednowęzłowych ze znakami i prawdopodobieństwami ich występowania w tekście
<KLP>: k-0.09090909090909091,
<KLP>: l-0.09090909090909091,
<KLP>: m-0.09090909090909091,
<KLP>: o-0.09090909090909091,
<KLP>: t-0.09090909090909091,
<KLP>:  -0.18181818181818182,
<KLP>: a-0.36363636363636365,

(2)  drzewo Huffmana przeczytane w porządku KLP
<LKP>: _-1.0, _-0.36363636363636365, _-0.18181818181818182, k-0.09090909090909091, l-
0.09090909090909091,
_-0.18181818181818182, m-0.09090909090909091, o-0.09090909090909091, _-0.6363636363636364,
_-0.2727272727272727, t-0.09090909090909091,  -0.18181818181818182, a-0.36363636363636365,

(3)  kody znaków wygenerowane na podstawie drzewa Huffmana
{ =101, o=011, k=000, m=010, a=11, t=100, l=001}

(4)  zakodowany tekst za pomocą kodów Huffmana (w postaci ciągu 0-1-kowego i ciągu znaków)
11001111010101110100001110011
gUh9@
-WYNIK KOMPRESJI-------------------------------------
(5)  wynik kompresji - tablica częstości i zakodowany tekst
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,0,0,0,0,0,0,
0,0,0,1,1,1,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,
gUh9@

-DEKOMPRESJA-----------------------------------------
(1)  tablica częstości wystąpień znaków w tekście
```


KAPITAŁ LUDZKI NARODOWA STRATEGIA SPÓJNOŚCI
UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY
Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

```
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,0,0,0,0,0,0,
0,0,0,1,1,1,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,

(2)  lista drzew jednowęzłowych ze znakami i prawdopodobieństwami ich występowania w tekście
<KLP>: k-0.09090909090909091,
<KLP>: l-0.09090909090909091,
<KLP>: m-0.09090909090909091,
<KLP>: o-0.09090909090909091,
<KLP>: t-0.09090909090909091,
<KLP>:  -0.18181818181818182,
<KLP>: a-0.36363636363636365,

(3)  drzewo Huffmana przeczytane w porządku KLP
<KLP>: _-1.0, _-0.36363636363636365, _-0.18181818181818182, k-0.09090909090909091, l-
0.09090909090909091, _-0.18181818181818182, m-0.09090909090909091, o-0.09090909090909091, _-
0.6363636363636364,
_-0.2727272727272727, t-0.09090909090909091,  -0.18181818181818182, a-0.36363636363636365,

(4)  zakodowany tekst za pomocą kodów Huffmana (w postaci ciągu 0-1-kowego)
11001111010101110100001110011000000

-WYNIK DEKOMPRESJI------------------------------------
(5)  wynik dekompresji - rozkodowany tekst
ala ma kota
```


KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI
UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ SPOŁECZNY
Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego