# Szyfry Klasyczne – Kółko Informatyczne 27 marca 2012

Materiały na kółko znajdziesz na stronie

<http://www.mimuw.edu.pl/~amn/20marca/>

Są tam trzy nowe pliki: enigma.xlsx, enigma.cpp, zagadka.txt.

Odszyfrujemy szyfrogram z poprzednich zajęć na dwa sposoby. Długość hasła-klucza odczytaliśmy ostatnim razem: ma ono sześć liter. Jedną z nich już znamy:

**\_ \_ T \_ \_ \_**

Jakie są pozostałe litery?

## Metoda I: metoda dopasowywania

W pliku enigma.xlsx znajdziesz arkusz kalkulacyjny, za pomocą możesz łatwo wypróbowywać swoje hipotezy. Arkusz pomaga wyświetlając częstotliwość występowania liter w naszym tekście i średnią częstotliwość występowania liter w języku angielskim.

Zaczekaj, aż razem dokładnie omówimy działanie arkusza.

## Metoda II: analiza statystyczna

Do tej analizy potrzebne nam będzie pojęcie **odchylenia standardowego**.

Szczegóły na tablicy…

**Ćwiczenie**. Poniżej przedstawione są wyniki dwóch skoczków podczas treningu.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Adam Małysz | 105,5 | 104 | 101 | 103,5 | 107 | 103 |
| Kamil Stoch | 106 | 102 | 99 | 108,5 | 103,5 | 105 |

1. Który z nich dalej skakał?
2. Który z nich uzyskał lepszy średni wynik?
3. Który z nich prezentuje stabilniejszą formę?
4. Którego wziąłbyś na zawody indywidualne? Którego na drużynowe?

**Ćwiczenie.** Artylerzyście kazano pięciokrotnie strzelić do celu A, B oraz C. Zmierzono, jak daleko od celu padły jego strzały:

|  |  |  |  |
| --- | --- | --- | --- |
| **Strzał** | **Cel A** | **Cel B** | **Cel C** |
| **1** | 2,3 | 1,5 | 1,2 |
| **2** | 2,5 | 1,7 | 1,4 |
| **3** | 1,7 | 2,6 | 0,8 |
| **4** | 2,8 | 2,3 | 0,9 |
| **5** | 3,1 | 2,2 | 1,1 |

1. Oblicz średnią odległość strzałów od celu A, B i C.
2. Oblicz odchylenie standardowe dla wszystkich strzałów. Użyj osobnych średnich dla celu A, celu B i celu C.
3. Który cel leży najbliżej? Który najdalej?

**Ćwiczenie.** W tabeli przedstawiono częstości występowania liter w języku angielskim i częstości ich wystąpień w analizowanym tekście. Jak obliczyć odchylenie standardowe?

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **a** | **b** | **c** | **d** | **e** | **f** | **g** | **h** | **i** | **j** | **k** | **l** | **…** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 0,0% | 1,9% | 7,5% | 0,0% | 0,0% | 11,3% | 0,0% | 9,4% | 1,9% | 1,9% | 3,8% | 1,9% | 3,8% |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 8,1% | 1,4% | 2,7% | 4,2% | 12,7% | 2,2% | 2,0% | 6,0% | 6,9% | 0,1% | 0,7% | 4,0% | 2,4% |  |  |  |  |  |  |  |  |  |  |  |  |  |

## Zadania

1. Napisz program, który odczytuje ze standardowego wejścia liczbę **n** i ciąg n liczb, liczy średnią arytmetyczną dla tego ciągu i odchylenie standardowe.
2. Zanalizuj załączony program (linijka po linijce). Być może będziesz musiał objaśnić jego działanie pozostałym.
3. Zmodyfikuj ten program tak, by za jego pomocą można było odszyfrować tekst z pliku zagadka.txt.

## Zadania domowe

1. If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3, 5, 6 and 9. The sum of these multiples is 23.

Find the sum of all the multiples of 3 or 5 below 1000.

1. A Pythagorean triplet is a set of three natural numbers, *a* <*b* <*c*, for which,

*a*2 + *b*2 = *c*2

For example, 32 + 42 = 9 + 16 = 25 = 52. There exists exactly one Pythagorean triplet for which *a* + *b* + *c* = 1000. Find the product *abc*.

#include<iostream>  
#include<string>  
#include<math.h>  
using namespace std;

string kryptogram =

"hyxvkazrnxnoffyxneffufkrmfyxnerztqunrjtrjotkaisufuxvufhyxw"  
 "ulrzxvywvfxwiofkxhzhsthrbomthqsihkxhjufzgkzhsebkntcwmlktkv"  
 "gxessthrjotatraafphrktvfnwgnrvbkntvlghxerrghzwsemcyilntwzh"  
 "ijgszjcrfhgccjmeanxllxryqfghkmbvwxudsrmlotkrlmzhsnkizcvvww" "krjrgxufhyxejmwebwzrokbstotkaijiodhrjdwjmvochpxwoazfgichcj" "bkttvzlaotvdrxxusetqkofkxkg";

const int d = 6; // dlugosc hasla

int l = 0; // numer aktualnie analizowanej litery hasła

char c; // sprawdzana litera do klucza

int freq[26]; // to będziemy liczyć: częstotliwość występowania liter

float odch[26]; // to będziemy liczyć odchylenia standardowe z populacji  
 // dla wszystkich możliwych podstawień c

char najlepsza\_litera[d]; // znalezione litery z najmniejszymi odchyleniami

float letter\_freq[26] = {

8.167, // a  
 1.492, // b  
 2.782, // c  
 4.253, // d  
 12.702, // e  
 2.228, // f  
 2.015, // g  
 6.094, // h  
 6.966, // i  
 0.153, // j  
 0.772, // k  
 4.025, // l   
 2.406, // m  
 6.749, // n  
 7.507, // o  
 1.929, // p  
 0.095, // q  
 5.987, // r  
 6.327, // s  
 9.056, // t  
 2.758, // u  
 0.978, // v  
 2.360, // w  
 0.150, // x  
 1.974, // y  
 0.074 // z  
};

int main()  
{  
 // ustalamy wartość l (przebiega liczby od 0 do d-1)  
 for (l = 0; l < d; l++) {  
 // ustalamy wartość c (przebiega litery od 'a' do 'z')  
 // jest to testowana właśnie litera do hasła

for (c = 'a'; c <= 'z'; c++) {  
 // dla ustalonego l i c obliczamy częstotliwość występowania  
 // liter w ciągu (bierzemy co l-tą literę, przesuwamy o c w lewo  
 // i zliczamy wystąpienia)

for (int i = 0; i < 26; i++) {  
 freq[i] = 0;  
 }  
 for (int i = l; i < kryptogram.length(); i += d) {  
 // bierzemy litery kryptogramu co d, zaczynamy od l-tej char k = kryptogram[i];  
 k = k - c;  
 if (k < 0) { k = k + 26; }  
 freq[k]++;  
 }  
 // liczymy odchylenie standardowe z populacji :)  
 // (wypełniamy jedno miejsce tabeli odch)

int t = 0;  
 for (int i = 0; i < 26; i++) {  
 t = t + freq[i];  
 }

float od = 0;  
 for (int i = 0; i < 26; i++) {

float f = 100.0 \* (float) freq[i] / (float) t;  
 od = od + (f - letter\_freq[i]) \* (f - letter\_freq[i]);

}

od = sqrt(od/ (float) t);  
 odch[c - 'a'] = od;

}

// szukamy litery z najmniejszym odchyleniem

float min = 100000000;  
 int i\_min = -1;

for (int i = 0; i < 26; i++) {  
 if (odch[i] < min) {  
 i\_min = i; min = odch[i];  
 }  
 }  
 // wypisujemy tę literę  
 cout << "Litera nr " << l << " hasła to: " << (char) (i\_min + 'a')   
 << "\n";  
 }  
 return 0;  
}