

Kriptografija i sigurnost mreža 2023.

2. domaća zadaća

Doris Đivanović

Zadatak 1

Vigenereovom širom iz otvorenog teksta na hrvatskom jeziku dobiven je šifrat:

RZZRU	VXCAM	ZEAXH	EXGTB	SMYUX	NUVYI	AWMIO	ZQFHW	
OVJHI	IIAYB	ZTUVX	XDWTI	GOYZU	JZFHT	ZOACF	MPEHO	
RXGVA	XZPOS	JGHJG	RHYDE	RFPPD	XKTPY	NHGSJ	BZOUN	
ZZIYI	NTFHN	KCXRN	HGVPZ	HTXED	GSERA	HZNZG	AATZF	KTP

Odredite najprije **duljinu ključne riječi**, potom samu **ključnu riječ**, te **dekriptirajte šifrat**.

Duljina ključne riječi

Prva metoda

Duljinu ključne riječi najprije sam pokušala odrediti pomoću tzv. **Kasiskijevog testa**. Vrlo brzim pregledom šifrata uočila sam da je jedini trigram koji se u šifratu pojavljuje dva puta **UVX**. Četvrto slovo jednog i drugog trigrama ne podudaraju se, pa zaključujem da u šifratu sigurno nema segmenata duljine veće od 3 koji se pojavljuju dva puta. Trigram UVX prvi se put pojavljuje na poziciji 5, a drugi put na poziciji 53, pa razlika njihovih pozicija iznosi $53 - 5 = 48$. Broj 48 možemo faktorizirati na sljedeće načine: $48 = 2^4 \cdot 3 = 16 \cdot 3 = 8 \cdot 6$. Među višekratnicima ove razlike mogu pronaći neke kandidate za duljinu ključne riječi, pa bih mogla pretpostaviti da je duljina ključne riječi $m = 3$ (iako mi je to sumnjivo), ili možda $m = 6$.

Druga metoda

Pribjegavam drugoj metodi koja za svaki kandidat m za duljinu ključne riječi dani šifrat zapisuje stupac po stupac u (krnju) matricu dimenzija $m \times \lceil m/n \rceil$, gdje je n duljina šifrata. Time šifrat dijeli u m podnizova, koji su pojedinačno, ako je m uistinu duljina ključne riječi, šifrirani Cezarovom šifrom s nekim ključem. Također, tada **indeks koinkidencije** svakog od podnizova mora biti **jako blizu 0.064** (≈ 0.064 ima tekst pisan na govornom jeziku, a i njegov monoalfabetški šifrat).

Napisala sam sljedeći program koji taj postupak radi za $m = 1, 2, \dots, 10$. Za svaki m kreira se i na ekran ispisuje matrica. "Višak u matricama" popunila sam znakom *. Zatim se za svaki podniz z_1, \dots, z_m računa indeks koinkidencije $I_c(z_i)$ po formuli i ispisuje se na ekran. Radi lakše provjere koji je m vjerojatnije duljina ključne riječi, za svaki dobiveni indeks računala sam i ispisala njegovo odstupanje od broja 0.064 - u kodu nazvano **GREŠKA**.

```
1 #include <iostream>
2 #include <string>
3 #include <vector>
4 #include <cmath>
5
6 using namespace std;
7
```

```

8 int main (void)
9 {
10     string Cyphertext = string("RZZRUVXCAMZEAXHEXGTBSMYUXNUVYIAWMIOZQFHW") +
11                             string("OVJHIIIIAYBZTUVXXDWTIGOYZUJZFHTZOACFMPEHO") +
12                             string("RXGVAXZPOSJGHJGRHYDERFPDXKTPYNHGSJBZOUN") +
13                             string("ZZIYINTFHNK CXRNHGVPZHTXEDGSERAHZNZGAATZFKTP");
14
15     int n = Cyphertext.size();    /// n je duljina zadanog sifrata
16
17     /// slova medunarodne abecede redom 0 - 25
18     vector<char> slovo = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K',
19                          'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V',
20                          'W', 'X', 'Y', 'Z'};
21
22     cout << "Radit cu metodu napada s matricom za m = 1, 2, ... , 10." << "\n\n";
23
24     for(int m = 1; m <= 10; m++)
25     {
26         cout << "m = " << m << "\n\n";
27
28         /// Matrica ima m redaka i ceiling(n/m) stupaca
29         const int broj_redaka = m;
30         const int broj_stupaca = ceil((double)n/(double)m);
31
32         char** matrica = (char**)malloc(broj_redaka*sizeof(char*));
33         for(int i = 0; i < broj_redaka; i++)
34             matrica[i] = (char*)malloc(broj_stupaca*sizeof(char));
35
36         /// Matricu stupac po stupac punim redom elementima iz sifrata
37         for(int j = 0; j < broj_stupaca; j++)
38             for(int i = 0; i < broj_redaka; i++)
39             {
40                 if(j*broj_redaka + i < n)
41                     matrica[i][j] = Cyphertext[j*broj_redaka + i];
42                 else
43                     /// EVENTUALNI VISAK ELEMENATA MATRICE POSTAVIM NA *
44                     matrica[i][j] = '*';
45             }
46
47         /// Ispis matrice
48         cout << "Matrica: \n\n";
49         for(int i = 0; i < broj_redaka; i++)
50         {
51             for(int j = 0; j < broj_stupaca; j++)
52             {
53                 cout << matrica[i][j];
54             }
55             cout << endl << endl;
56         }
57
58         vector<double> indeks(broj_redaka, 0.0);
59         int duljina_podniza, frekvencija;
60
61         /// Racunanje indeksa koincidencije za svaki redak matrice
62         for(int redak = 0; redak < broj_redaka; redak++)
63         {
64             if(matrica[redak][broj_stupaca - 1] != '*')
65                 duljina_podniza = broj_stupaca;
66             else
67                 duljina_podniza = broj_stupaca - 1;
68
69             for(int i = 0; i < 26; i++)
70             {

```

```

71         frekvencija = 0;
72
73         for(int l = 0; l < duljina_podniza; l++)
74         {
75             if(matrica[redak][l] == slovo[i])
76                 frekvencija++;
77         }
78
79         indeks[redak] += frekvencija*(frekvencija - 1);
80     }
81
82     indeks[redak] = indeks[redak]/(duljina_podniza*(duljina_podniza - 1));
83 }
84
85 /// Ispis svih indeksa koineidencije za odredeni broj m
86 cout << "Indeksi koineidencije za podnizove: \n\n";
87 for(int i = 0; i < indeks.size(); i++)
88 {
89     cout << "Podniz z_" << i + 1 << " : ";
90     cout << "I_c = " << indeks[i];
91     cout << " GRESKA: " << abs(indeks[i]-0.064) << endl << endl;
92 }
93 cout << endl << "-----" << endl << endl;
94
95 for(int i = 0; i < broj_redaka; i++)
96     free matrica[i];
97 free matrica;
98 }
99
100 return 0;
101 }

```

Druga metoda - analiza rezultata

Gledajući output dobiven na ekranu, čini mi se da su najmanje greške za $m = 3$ i $m = 6$, ali još ne mogu zaključiti koja je sigurno duljina ključne riječi:

```

C:\Users\Djivo1\Desktop\VigenereDuljina.exe

Indeksi koineidencije za podnizove:
Podniz z_1 : I_c = 0.0532972  GRESKA: 0.0107028
Podniz z_2 : I_c = 0.041358  GRESKA: 0.022642

-----

m = 3
Matrica:
RRXMAETMXVAIQWJIYTXWGZZTAMHXAPJJHEPXPJHOZYTNXHPTDEHZAFP
ZUCZXXBYNYWOFHIBUXTOUFZCPOGXOGGYRPKYGBUZIPKRGZXGRZGTK*
ZVAEHGSUUIMZHVIAZVDIYJHOFERVZSHRDFDTNSZNINHCNVHESANAZT*
Indeksi koineidencije za podnizove:
Podniz z_1 : I_c = 0.0518519  GRESKA: 0.0121481
Podniz z_2 : I_c = 0.0587002  GRESKA: 0.00529979
Podniz z_3 : I_c = 0.0545073  GRESKA: 0.00949266

-----

m = 4
Matrica:
RUAAXSXYMQ0IYUDGUHAPRA0HHRDPGZZIHGXGHDRAK

```

```
C:\Users\Djivo1\Desktop\VigenereDuljina.exe
Podniz z_5 : I_c = 0.0342742  GRESKA: 0.0297258

-----

m = 6
Matrica:
RXATXAQJYXGZAHAJHPPJZTXPDHAP
ZCXBWFWHBXOFCOXGYPYBZFRZGZT*
ZAHSUMHIZDYHFRZHDDNZIHNHSNZ*
RMEMVIWITWZTMXPJEXHOYNHTEZF*
UZXXYY00IUTUZPG0GRKGUIKGXRGK*
VEGUIZVAVIJOEVSFRFTSNNCVAAT*
Indeksi koincidencije za podnizove:
Podniz z_1 : I_c = 0.0793651  GRESKA: 0.0153651
Podniz z_2 : I_c = 0.0541311  GRESKA: 0.00986895
Podniz z_3 : I_c = 0.0940171  GRESKA: 0.0300171
Podniz z_4 : I_c = 0.039886  GRESKA: 0.024114
Podniz z_5 : I_c = 0.0769231  GRESKA: 0.0129231
Podniz z_6 : I_c = 0.0569801  GRESKA: 0.00701994
```

Ključna riječ

U svrhu određivanja same ključne riječi, napisala sam još jedan program. On nam omogućava da za fiksiranu, odabranu duljinu ključne riječi m odredimo ključnu riječ. Program najprije fiksira m . Zatim kreira i ispisuje na ekran gore spomenutu matricu samo za tu duljinu. Zatim, postupkom koji smo radili na predavanju, pretpostavljajući da je **otvoreni tekst pisan na hrvatskom jeziku**, računamo zajedničke indekse koincidencije svakog od podnizova z_1, \dots, z_m , tj. svakog retka matrice $j = 1, \dots, m$, i slučajnog teksta na hrv. jeziku, i time za svaki redak računamo ključ k_j za njegovu Cezarovu šifru. Na kraju, spajanjem tih ključeva u jedan, tj. u jednu riječ, dobivamo i ispisujemo na ekran ključnu riječ duljine m za Vigenereovu šifru. Također, prolazeći po retcima matrice, dešifriramo svaki redak njegovim ključem i dešifrirane elemente spremamo na odgovarajuća mjesta. Dešifriranu matricu ispisujemo kao jedan string prolazeći po stupcima i time dobivamo otvoreni tekst za tu ključnu riječ.

```
1 #include <iostream>
2 #include <string>
3 #include <vector>
4 #include <cmath>
5 #include <map>
6
7 using namespace std;
8
9 int main (void)
10 {
11     string Cyphertext = string("RZZRUVXCAMZEAXHEXGTBSMYUXNUVYIAWMIOZQFHW") +
12                             string("OVJHIIIAVBZTUVXXDWTIGOYZUJZFHTZOACFMPEHO") +
13                             string("RXGVAXZPOSJGHJGRHYDERFPDXKTPYNHGSJBZOUN") +
14                             string("ZZIYINTFHNK CXRNHGVPZHTXEDGSEAHZNZGAATZFKTP");
15
16     int n = Cyphertext.size(); /// n je duljina zadanog sifrata
17
18     /// slova medunarodne abecede redom 0 - 25
19     vector<char> slovo = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K',
20                          'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V',
```

```

21         'W', 'X', 'Y', 'Z'};
22
23     /// frekvencije slova u HRVATSKOM jeziku redom
24     vector<double> p = {0.115, 0.015, 0.028, 0.037, 0.084, 0.003, 0.016, 0.008,
25                        0.098, 0.051, 0.036, 0.033, 0.031, 0.066, 0.090, 0.029,
26                        0.000, 0.054, 0.056, 0.048, 0.043, 0.035, 0.000, 0.000,
27                        0.000, 0.023};
28
29     /// Matrica ima m = 6 redaka i ceiling(n/6) stupaca
30     const int m = 6;
31     const int broj_stupaca = ceil((double)n/(double)m);
32
33     char** matrica = (char**)malloc(m*sizeof(char*));
34     for(int i = 0; i < m; i++)
35         matrica[i] = (char*)malloc(broj_stupaca*sizeof(char));
36
37     /// Matricu stupac po stupac punim redom elementima iz sifrata
38     for(int j = 0; j < broj_stupaca; j++)
39         for(int i = 0; i < m; i++)
40         {
41             if(j*m + i < n)
42                 matrica[i][j] = Cyphertext[j*m + i];
43             else
44                 /// EVENTUALNI VISAK ELEMENATA MATRICE POSTAVIM NA *
45                 matrica[i][j] = '*';
46         }
47
48     /// Ispis matrice
49     cout << "Matrica: \n\n";
50     for(int i = 0; i < m; i++)
51     {
52         for(int j = 0; j < broj_stupaca; j++)
53         {
54             cout << matrica[i][j];
55         }
56         cout << endl << endl;
57     }
58
59     vector<int> k (m);    /// ovdje cu redom spremiti slova kljuca (tj. odg. brojeve)
60
61     /// Trazenje k_j za svaki j
62     for(int j = 0; j < m; j++)
63     {
64         int duljina_podniza;
65         if(matrica[j][broj_stupaca - 1] != '*')
66             duljina_podniza = broj_stupaca;
67         else
68             duljina_podniza = broj_stupaca - 1;
69
70         vector<int> frekvencija(26, 0);
71         for(int i = 0; i < 26; i++)
72             for(int l = 0; l < duljina_podniza; l++)
73             {
74                 if(matrica[j][l] == slovo[i])
75                     frekvencija[i]++;
76             }
77
78         /// Racunanje M_0 - M_25 za fiksirani j
79         vector<double> M (26, 0.0);
80         for(int g = 0; g < 26; g++)
81         {
82             for(int i = 0; i < 26; i++)
83             {

```

```

84         int mod = (i - g) % 26;
85         if(mod < 0) mod = 26 + mod;
86
87         M[g] += p[i]*frekvencija[mod];
88     }
89
90     M[g] = M[g]/duljina_podniza;
91 }
92
93 /// Trazenje maksimalnog M za fiksirani j
94 double max = M[0];
95 int h = 0;
96 for(int g = 1; g < 26; g++)
97 {
98     if(M[g] > max)
99     {
100         max = M[g];
101         h = g;
102     }
103 }
104
105 if(h == 0) k[j] = 0;
106 else k[j] = 26 - h;    /// racunanje k_j
107
108 cout << "Za j = " << j + 1 << " imamo h = " << h;
109 cout << ", M_" << h << " = " << max;
110 cout << ", pa je k_" << j + 1 << " = " << k[j] << "; \n\n";
111 }
112
113 cout << "Kljucna rijec: ";
114 for(int j = 0; j < m; j++)
115     cout << slovo[k[j]];
116 cout << endl << endl;
117
118 /// Korespondencija brojeva i slova - drugi smjer
119 map<char, int> mapping = {{'A', 0}, {'B', 1}, {'C', 2}, {'D', 3}, {'E', 4},
120                          {'F', 5}, {'G', 6}, {'H', 7}, {'I', 8}, {'J', 9},
121                          {'K', 10}, {'L', 11}, {'M', 12}, {'N', 13},
122                          {'O', 14}, {'P', 15}, {'Q', 16}, {'R', 17},
123                          {'S', 18}, {'T', 19}, {'U', 20}, {'V', 21},
124                          {'W', 22}, {'X', 23}, {'Y', 24}, {'Z', 25}};
125
126 /// Dekriptiranje
127 for(int j = 0; j < m; j++)
128     for(int i = 0; i < broj_stupaca; i++)
129     {
130         if(matrica[j][i] != '*')
131         {
132             /// od svakog broja u j-tom retku oduzimam k_j mod 26
133             int mod = (mapping[matrica[j][i]] - k[j]) % 26;
134             if(mod < 0) mod = 26 + mod;
135
136             /// u istu matricu pohranjujem odgovarajuca dekriptirana slova
137             matrica[j][i] = slovo[mod];
138         }
139     }
140
141 /// Ispisujem dekriptiranu matricu kao jedan string, stupac po stupac
142 cout << "Otvoreni tekst: \n\n";
143 for(int j = 0; j < broj_stupaca; j++)
144     for(int i = 0; i < m; i++)
145     {
146         if(j*m + i >= n) break;    /// dosli smo do kraja teksta

```

```

147         cout << matrica[i][j];
148     }
149     cout << endl;
150
151     for(int i = 0; i < m; i++)
152         free matrica[i];
153     free matrica;
154
155     return 0;
156 }

```

Uočimo da nam ovaj program može poslužiti kao konačan alat za određivanje duljine ključne riječi. Jednostavno, mijenjanjem parametra m , možemo vidjeti za koji će dobivena ključna riječ biti smisljena (iako to ne mora biti), ali možemo vidjeti i za koji će m otvoreni tekst biti smisljena poruka, što nam je sigurna potvrda. Uvrštavanjem $m = 3$ u gornji program, dobiva se da je ključna riječ = PGZ, a otvoreni tekst uopće nije smislen. Međutim, uvrštavanjem $m = 6$, dobiva se da je ključna riječ = POŽEGA i dobiva se smisljeni otvoreni tekst:

```

Matrica:
RXATXAQJYXGZAHJHPPJZTXPDHAP
ZCXBNWFHBX0FC0XGYBYBZFRZGZT*
ZAHSUMHIZDYHFRZHDNDZIHNSNZ*
RMEMVIWITWZTMXPJEXHOYNHTEZF*
UZXYV00IUTUZPG0GRKGUIKGXRGK*
VEGUIZVAVIJ0EVSFRFTSNNCVAAT*
Za j = 1 imamo h = 11, M_11 = 0.064, pa je k_1 = 15;
Za j = 2 imamo h = 12, M_12 = 0.0541852, pa je k_2 = 14;
Za j = 3 imamo h = 1, M_1 = 0.0775556, pa je k_3 = 25;
Za j = 4 imamo h = 22, M_22 = 0.0572963, pa je k_4 = 4;
Za j = 5 imamo h = 20, M_20 = 0.0754074, pa je k_5 = 6;
Za j = 6 imamo h = 0, M_0 = 0.0598519, pa je k_6 = 0;
Ključna rijec: POŽEGA
Otvoreni tekst:
CLANOVI OBITELJI ARGENTI SU IZVRSILI NEIZBRISIV UTJECAJ NA POVIJESNI RAZVOJ KRIPTOLOGIJE. SASTAVLJALI SU SIFARSKE ALFABETE TAKO DA SU NAKON KLJUCNE RIJECI DODAVALI PREOSTALA SLOVA ALFABETA
Process returned 0 (0x0)   execution time : 0.220 s
Press any key to continue.

```

Dakle, zaključujemo da je:

- duljina ključne riječi = 6,
- ključna riječ = POŽEGA,
- otvoreni tekst glasi:

CLANOVI OBITELJI ARGENTI SU IZVRSILI NEIZBRISIV UTJECAJ NA POVIJESNI RAZVOJ KRIPTOLOGIJE. SASTAVLJALI SU SIFARSKE ALFABETE TAKO DA SU NAKON KLJUCNE RIJECI DODAVALI PREOSTALA SLOVA ALFABETA.

Zadatak 2

Šifrirajte otvoreni tekst

PARKER HITT

pomoću **Playfair**ove šifre s ključnom riječi CRYPTOGRAPHY.

Rješenje

Matrica slova dimenzije 5×5 uzimajući u obzir ključnu riječ:

C	R	Y	P	T
O	G	A	H	B
D	E	F	IJ	K
L	M	N	Q	S
U	V	W	X	Z

Rastav otvorenog teksta na bigrame:

PA RK ER HI TT

Bigram od istih slova rastavlja se s X, broj slova do parnog nadopunjuje se s X:

PA RK ER HI TX TX

Šifriranje:

- P i A nasuprotni vrhovi pravokutnika $\Rightarrow PA \rightarrow YH$,
- R i K nasuprotni vrhovi pravokutnika $\Rightarrow RK \rightarrow TE$,
- E i R u istom stupcu $\Rightarrow ER \rightarrow MG$,
- H i I u istom stupcu $\Rightarrow HI \rightarrow IQ$,
- T i X nasuprotni vrhovi pravokutnika $\Rightarrow TX \rightarrow PZ$.

Šifrat:

YHTEMG IQPZPZ

Zadatak 3

Odredite ključ K u **Hillovoj šifri** ako je poznato da je $m = 2$, te da otvorenom tekstu

HEBERN

odgovara šifrat

RVBHCF.

Rješenje

Budući da je $m = 2$, Hillovom se šifrom dva uzastopna slova u otvorenom tekstu zamjenjuju s dva uzastopna slova u šifratu. Stoga, kako znamo da se otvoreni tekst HEBERN šifrira u RVBHCF, uz tablicu korespondencije slova međunarodne abecede i skupa ostataka modulo 26, možemo zaključiti da za traženi ključ K vrijedi:

$$e_K(7, 4) = (17, 21),$$

$$e_K(1, 4) = (1, 7),$$

$$e_K(17, 13) = (2, 5).$$

Dakle, imamo tri para uređenih parova

$$(x_{1j}, x_{2j}) \quad \text{i} \quad (y_{1j}, y_{2j}) \quad \text{t. d.} \quad e_K(x_{1j}, x_{2j}) = (y_{1j}, y_{2j}).$$

Budući da je $m = 2$, za analizu su nam dovoljna dva uređena para, npr. $(7, 4)$ i $(17, 13)$. Iz gornjih dviju jednadžbi te definicije Hillovog kriptosustava dobivamo sljedeću matričnu jednadžbu (**sve operacije su mod 26**):

$$\begin{vmatrix} 7 & 4 \\ 17 & 13 \end{vmatrix} K = \begin{vmatrix} 17 & 21 \\ 2 & 5 \end{vmatrix}.$$

Ako je matrica X invertibilna, ključ K možemo dobiti na sljedeći način:

$$K = X^{-1}Y.$$

Znamo da je matrica A u \mathbb{Z}_{26} invertibilna akko joj je determinanta invertibilna u \mathbb{Z}_{26} , tj. ako je $(\det A, 26) = 1$, i tada je

$$A^{-1} = (\det A)^{-1} \begin{vmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{vmatrix}.$$

Računamo

$$\det X = \det \begin{vmatrix} 7 & 4 \\ 17 & 13 \end{vmatrix} = 7 \cdot 13 - 17 \cdot 4 = 91 - 68 = 23 \bmod 26 = 23.$$

Sada iz tablice inverza u \mathbb{Z}_{26}

1	3	5	7	9	11	15	17	19	21	23	25
1	9	21	15	3	19	7	23	11	5	17	25

vidimo da je

$$(\det X)^{-1} = 23^{-1} = 17,$$

pa je X invertibilna matrica i vrijedi

$$X^{-1} = 17 \begin{vmatrix} 13 & -4 \\ -17 & 7 \end{vmatrix} = \begin{vmatrix} 13 & 10 \\ 23 & 15 \end{vmatrix}$$

Konačno, **možemo izračunati ključ K** :

$$K = \begin{vmatrix} 13 & 10 \\ 23 & 15 \end{vmatrix} \begin{vmatrix} 17 & 21 \\ 2 & 5 \end{vmatrix} = \begin{vmatrix} 7 & 11 \\ 5 & 12 \end{vmatrix}.$$

Za kraj ćemo provjeriti odgovara li dobiveni ključ K informaciji

$$e_K(1, 4) = (1, 7).$$

Dobivamo:

$$e_K(1, 4) = (1, 4)K = (1, 4) \begin{vmatrix} 7 & 11 \\ 5 & 12 \end{vmatrix} = (1, 7),$$

dakle u redu je.