4 body ♣

1A. Dokážte, že ak je daných 2026 prirodzených čísel, tak medzi nimi existujú dve, ktorých rozdiel je deliteľný číslom 2025.

Riešenie: Ľubovoľné prirodzené číslo má po delení číslom 2025 zvyšok, ktorý môže byť len z množiny $\{0, 1, 2, \ldots, 2024\}$. Po delení číslom 2025 teda existuje 2025 rôznych zvyškov.

V úlohe máme daných 2026 prirodzených čísel. Rozdelme si tieto čísla do skupín podla toho, aký zvyšok dávajú po delení číslom 2025. Keďže čísel je 2026 a možných zvyškov je len 2025, v aspoň jednej skupine musia byť aspoň 2 čísla. Ak spravíme rozdiel dvoch čísel s rovnakým zvyškom po delení číslom 2025, tak tento rozdiel bude deliteľný číslom 2025.

Vysvetlenie: Označme si tieto dve čísla čísla a a b a ich zvyšok po delení číslom 2025 nech je z.

$$a = m \cdot 2025 + z \quad \text{a} \quad b = n \cdot 2025 + z \quad \Rightarrow \quad a - b = (m - n) \cdot 2025\,.$$

---

4 body ♣

1B. Dokážte, že ak je daných $n+2$ po sebe idúcich prirodzených čísel, tak medzi existujú dve, ktorých súčet je deliteľný číslom $2n+1$.

Riešenie: Postup je veľmi podobný ako v úlohe 1A. Prirodzené číslo po delení číslom $2n + 1$ môže mať ako zvyšok len niektoré číslo z množiny $\{0, 1, 2, \ldots, 2n\}$, t. j. $2n + 1$ možností zvyšku.

Máme $n + 2$ **po sebe idúcich** prirodzených čísel, čo znamená, že ich zvyšky po delení číslom $2n + 1$ musia byť **navzájom rôzne**. Ak prvé z týchto čísel má po delení číslom $2n + 1$ zvyšok k, tak posledné z týchto čísel bude mať zvyšok $k+n+1$. Na to, aby sme vedeli medzi danými číslami nájsť dve, ktorých súčet je deliteľný číslom $2n + 1$ nám stačí, aby súčet zvyškov najväčších dvoch z daných čísel, bol väčší než $2n$. Zvyšky najväčších dvoch čísel sú $k+n$ a $k+n+1$ a pre ich súčet platí

$$(k + n) + (k + n + 1) = 2n + 1 + k > 2n \ \text{ pre } \ \forall k \in \{0, 1, 2, \ldots, 2n\}\,.$$

7 bodov

2A. Ku každému prirodzenému číslu n existuje jeho celočíselný násobok v tvare

$$77\dots700\dots0$$

Riešenie: Znenie úlohy 1A. môžeme zovšeobecniť nasledovným spôsobom

> *Dokážte, že ak je daných $n+1$ prirodzených čísel, tak medzi nimi existujú dve, ktorých rozdiel je deliteľný číslom n.*

Dôkaz takéhoto tvrdenie je úplne rovnaký ako dôkaz úlohy 1A. Stačí v dôkaze úlohy 1A. nahradiť číslo 2025 číslom n a číslo 2026 číslom $n+1$. No a z takto zovšeobecneného tvrdenia vyplýva dôkaz úlohy 2A.
Zoberme si $n+1$ prirodzených čísel v tvare

$$7\,,\ 77\,,\ 777\,,\ 7777\,,\ \dots\ \underbrace{777\dots7}_{n+1 \text{ sedmičiek}}$$

Medzi týmito $n+1$ číslami musia existovať dve také, že ich rozdiel je deliteľný číslom n. A rozdiel takýchto dvoch čísel bude mať tvar $77\dots700\dots0$.

---

7 bodov

2B. Je daných 2027 prirodzených čísel. Dokážte, že spomedzi nich možno vybrať niekoľko čísel tak, že ich súčet je deliteľný číslom 2026.

Riešenie: Pri tejto úlohe si možno niektorí povedia, že to je zovšeobecnenie úlohy 1B. a že stačí aj výrazne menej než uvedených 2027 prirodzených čísel. Ale nie je to tak. Je tu totiž jeden významný rozdiel. V úlohe 1B. sme mali $n+1$ **po sebe idúcich** prirodzených čísel a tú máme 2027 **ľubovoľných** (môžu byť aj rovnaké) prirodzených čísel.

Označme si daných 2027 prirodzených čísel takto: $a_1, a_2, a_3, \dots, a_{2027}$. Potom si skonštruujeme ďalších 2027 prirodzených čísel nasledovným spôsobom

$$s_1 = a_1 \quad , \quad s_2 = a_1 + a_2 \quad , \quad s_3 = a_1 + a_2 + a_3 \quad , \quad \dots \quad s_{2027} = a_1 + a_2 + a_3 + \dots + a_{2027}$$

Podľa zovšeobecneného tvrdenia úlohy 1A. (uvedené v riešení úlohy 2A.) musia medzi týmito číslami existovať dve také, že ich rozdiel bude deliteľný číslom 2026. No a rozdiel dvoch z čísel $s_1, s_2, \dots, s_{2027}$ nie je nič iné ako súčet niekoľkých z daných čísel $a_1, a_2, a_3, \dots, a_{2027}$.

9 bodov

3A. V kocke o hrane 15 cm leží 2198 bodov. Dokážte, že vzdialenosť niektorých dvoch z týchto bodov je menšia než 2.

Riešenie: Platí $13^3 = 2197$. Kocku o hrane 15 cm si preto rozdelíme, na 2197 malých kociek o hrane $\frac{15}{13}$, t. j. hrany pôvodnej kocky rozdelíme na 13 rovnakých častí. Ak máme kocku o hrane a, tak jej telesová uhlopriečka, čo je najväčšia možná vzdialenosť medzi dvoma bodmi v kocke, bude mať dĺžku $a\sqrt{3}$. Telesová uhlopriečka o hrane $\frac{15}{13}$ potom bude $\frac{15}{13}\sqrt{3} \approx 1,99 < 2$.

V pôvodnej kocke o hrane 15 cm leží 2198 bodov, my sme túto kocku rozdelili na 2197 malých kociek a podľa Dirichletovho princípu v aspoň jednej malej kocke musia ležať aspoň 2 z daných bodov. To ale znamená, že vzdialenosť týchto bodov musí byť menšia než 2.

---

9 bodov

3B. Dokážte, že každé prirodzené číslo, ktorého dekadický zápis končí niektorou z cifier $\{1, 3, 7, 9\}$ má celočíselný násobok, ktorého dekadický zápis pozostáva zo samých trojok.

Riešenie: Rovnakým spôsobom akým sme dokazovali tvrdenie z úlohy 2A. vieme dokázať aj tvrdenie

Ku každému prirodzenému číslu n existuje jeho celočíselný násobok v tvare $33\ldots300\ldots0$.

Majme teraz prirodzené číslo n, ktorého dekadický zápis končí niektorou z cifier $\{1, 3, 7, 9\}$. K tomuto číslu musí existovať jeho celočíselný násobok v tvare $33\ldots300\ldots0$. Číslo $33\ldots300\ldots0$ musí byť teda deliteľné číslom n a dá sa zapísať ako

$$33\ldots300\ldots0 = 33\ldots3 \cdot 10^k \quad \text{kde} \quad k \in \mathbb{N}\,.$$

Číslo n však končí niektorou z cifier $1, 3, 7$, alebo 9 a takéto číslo nemôže byť deliteľom čísla 10^k, pretože prvočíselný rozklad 10^k pozostáva len z mocnín prvočísel 2 a 5 a čísla končiace sa ciframi $\{1, 3, 7, 9\}$ nemajú v prvočíselnom rozklade ani 2, ani 5. To potom znamená, že číslo n musí byť deliteľom čísla $33\ldots3$.

10 bodov

4A. Šachového turnaja sa zúčastňuje 6 hráčov. Dokážte, že počas celého trvania turnaja vieme spomedzi nich vždy vybrať trojicu hráčov tak, že buď v tejto trojici už každý s každým hral, alebo ešte žiaden so žiadnym nehral.

Riešenie: Označme hráčov ako h_1, h_2, h_3, h_4, h_5 a h_6 a následne hráčov h_1, h_2, h_3, h_4, h_5 rozdelíme do dvoch skupín podľa toho, či už s hráčom h_6 hrali, alebo s ním ešte nehrali. Z Dirichletovho princípu je zrejmé, že v jednej z týchto dvoch skupín musia byť aspoň 3 hráči. Na označení hráčov nezáleží, takže môžeme predpokladať, že jedna skupina bude $\{h_1, h_2, h_3\}$ a títo hráči už s h_6 hrali a druhá skupina bude $\{h_4, h_5\}$ a títo hráči s h_6 ešte nehrali.

Sústreďme sa teraz na prvú skupinu, t. j. na hráčov $\{h_1, h_2, h_3\}$, ktorí už s h_6 hrali. Môžu nastať len dve možnosti. Buď niektorí dvaja z hráčov h_1, h_2, h_3 už spolu hrali, a potom títo dvaja spolu s h_6 sú trojicou hráčov, ktorí už hrali každý s každým, alebo žiaden so žiadnym z hráčov h_1, h_2, h_3 ešte spolu nehrali, a potom títo traja hráči sú trojicou, v ktorej žiadny so žiadnym hráčov nehral. To znamená, že počas celého trvania turnaja vždy existuje taká trojica hráčov, že buď v tejto trojici už každý s každým hral, alebo ešte žiaden so žiadnym nehral.

Poznámka: Podobne by sa úloha dokázala, ak by $\{h_1, h_2, h_3\}$ bola skupina hráčov, ktorí ešte nehrali s h_6.

10 bodov

4B. Postupnosť $2, 0, 2, 7, 1, 0, 0, 8, 9, 7, 4, 8, 8, \ldots$ je konštruovaná nasledovne: *každé ďalšie číslo je poslednou cifrou súčtu predchádzajúcich štyroch čísel.* Ak si túto postupnosť zapíšeme ako $(a_n)_{n=1}^{\infty}$, tak jej prvé členy sú $a_1 = 2, a_2 = 0, a_3 = 2, a_4 = 7$ a jej ďalšie členy sú dané rekurentným vzťahom $a_n = \big(a_{n-1} + a_{n-2} + a_{n-3} + a_{n-4}\big) \bmod 10\,.$

Vysvetlenie: mod znamená *„modulo“* a je to zvyšok po delení. Takže $a \bmod b = c$ znamená, že číslo a má po delení číslom b zvyšok c.

a) Zistite, či sa štvorica $2, 0, 2, 7$ v danej postupnosti zopakuje. Svoje tvrdenie musíte dokázať. **[8 bodov]**

b) Zistite, či sa v danej postupnosti vyskytne štvorica $4, 8, 7, 3$. Svoje tvrdenie musíte dokázať. **[2 body]**

Riešenie:

a) Označme si konštruovanú postupnosť $(a_n)_{n=1}^{\infty}$. Pre všetky jej členy platí $a_i \in \{0, 1, 2, \ldots, 9\}$. Z Dirichletovho princípu vyplýva, že nejaká štvorica po sebe idúcich členov tejto postupnosti, označme si ju $(a_n, a_{n+1}, a_{n+2}, a_{n+3})$, sa musí zopakovať. Možných štvoríc pozostávajúcich z cifier 0 až 9 je len 10 000 a naša postupnosť má nekonečne veľa členov. Musí preto existovať nejaké $m \in \mathbb{N}$, $m > n$ také, že $a_m = a_n$, $a_{m+1} = a_{n+1}$, $a_{m+2} = a_{n+2}$ a $a_{m+3} = a_{n+3}$.

Teraz sporom dokážeme, že opakovať sa musí už hneď prvá štvorica konštruovanej postupnosti. Predpokladajme, že tomu tak nie je. Takže predpokladáme, že prvé také n, pre ktoré sa bude opakovať štvorica $(a_n, a_{n+1}, a_{n+2}, a_{n+3})$ je nejaké $n > 1$. Potom ale podľa rekurentného vzťahu pre výpočet členov postupnosti platí

$$a_{n+3} = \big(a_{n-1} + a_n + a_{n+1} + a_{n+2}\big) \bmod 10 \quad \Rightarrow \quad a_{n-1} = \big(a_{n+3} - a_n - a_{n+1} - a_{n+2}\big) \bmod 10\,.$$

Podobne vieme ukázať, že potom platí aj $a_{m-1} = \big(a_{m+3} - a_m - a_{m+1} - a_{m+2}\big) \bmod 10$, čo znamená, že aj štvorica $(a_{n-1}, a_n, a_{n+1}, a_{n+2})$ sa zopakuje. To je však spor s tým, že $n \in \mathbb{N}$ je najmenšie také číslo, pre ktoré sa štvorica $(a_n, a_{n+1}, a_{n+2}, a_{n+3})$ zopakuje.
Takže ak by sme konštruovanú postupnosť začali akoukoľvek štvoricou čísel 0 až 9, tak táto sa musí v postupnosti zopakovať.

b) Odpoveď na otázku, či sa v danej postupnosti vyskytne štvorica $4, 8, 7, 3$ je triviálna. Pozrime sa ako by vyzerala konštruovaná postupnosť, ak by sme ju začali týmito štyrmi číslami

$$4, 8, 7, 3, 2, 0, 2, 7, \ldots$$

Keďže sme už v časti a) dokázali, že štvorica čísel $2, 0, 2, 7$ sa v konštruovanej postupnosti zopakuje (dokonca nekonečne veľa krát), tak sa tam musí vyskytovať a opakovať aj štvorica $4, 8, 7, 3$, ktorá jej predchádza.

16 bodov

♡

5A. Pátranie po poklade (Light): Pirátska posádka sa práve vylodila na opustenom ostrove, aby našla bájny poklad. Ostrov sa dá zjednodušene predstaviť ako mriežka (graf), kde každá bunka mriežky predstavuje určitý typ terénu. Existujú 2 typy terénu:

- Piesok: Cesta, ktorá je priechodná a dá sa cez ňu voľne pohybovať.
- Hory: Úplne nepriechodné.

Piráti môžu spraviť len jeden krok naraz, a to buď horizontálne, alebo vertikálne (diagonálny pohyb nie je možný). Jeden krok trvá 1 minútu.

Úloha: Tvojou úlohou je napísať v Pythone program, ktorý nájde najrýchlejšiu cestu z východiskového bodu (pirátskej lode) k pokladu.

Vstupné údaje: Vstupné údaje budú zadané v súbore `mapa.txt`. Prvý riadok bude obsahovať rozmery mapy R (počet riadkov) a C (počet stĺpcov), oddelené medzerou. Nasleduje R riadkov, pričom každý z nich má C znakov, ktoré predstavujú mapu ostrova.

H – Hory	P – Piesok
L – Pirátska loď (**východiskový bod**)	X – Poklad (**cieľ**)

Príklad súboru `mapa.txt`:

```
5 5
PLPHP
PHHPP
PPPHH
PHPPH
PPHXH
```

Výstupné údaje: Výstupom bude jedno číslo, napríklad 8, ktoré predstavuje minimálny čas, za ktorý je možné dosiahnuť poklad. Ak nie je možné dostať sa k pokladu, výstupom bude číslo `-1`. Výstup, t. j. najrýchlejšia cesta, sa uloží do súboru `cesta.txt`.

Riešenie: Tento problém je klasickou úlohou hľadania najkratšej cesty v grafe. Keďže každý pohyb má v zadaní rovnaký čas (prejsť cez každé políčko s pieskom trvá 1 minútu), algoritmus BFS (Breadth-First Search – prehľadávanie grafu do šírky) je na toto ideálny, pretože systematicky prehľadáva susedné bunky vrstvu po vrstve, čím zaručuje nájdenie najkratšej cesty. Na mriežku s jednotlivými políčkami sa pozeráme ako na jednotlivé vrcholy grafu, ktoré sú vzájomne prepojené. Políčka, ktoré predstavujú hory nezahrnieme do grafu, resp. zostanú ako izolované (nespojené s ostatnými vrcholmi) vrcholy grafu, nakoľko hory sú nepriechodné. Počet minút, ktorý zaberie prechod jedným políčkom v mriežke na ceste k pokladu je hodnotou hrany, ktorá prepája dva susedné vrcholy. Štartovacím vrcholom je políčko pirátskej lode (`L`). Cieľovým vrcholom je políčko s pokladom (`X`).

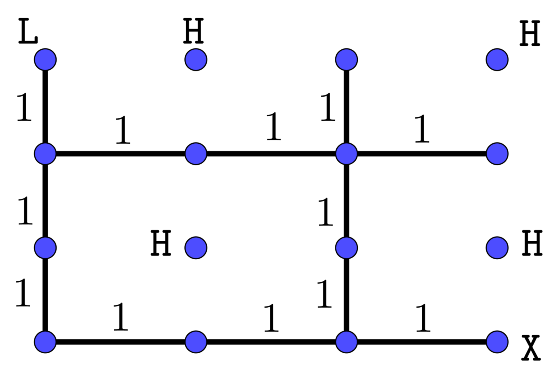
Postup: Predstavme si BFS ako vlnu, ktorá sa šíri z lode (`L`) na všetky strany súčasne.

1. Inicializácia
   - Do fronty (špeciálny zoznam typu FIFO – First In, First Out) pridáme štartovací bod (`L`) s časom 0.
   - Vytvoríme si zoznam navštívených bodov. Loď si rovno označíme ako navštívenú.
2. Cyklus prehľadávania (kým fronta obsahuje nejaké prvky):
   - Vezmeme prvý bod z fronty (najstarší).
   - Kontrola cieľa: Ak je tento bod poklad (`X`), našli sme najkratšiu cestu! Vrátime aktuálny čas.
   - Preskúmanie susedov: Pozrieme sa na všetky 4 susedné bunky (hore, dole, vľavo, vpravo).
   - Pre každého suseda, ktorý:
     - je v rámci mapy,
     - nie je hora (`H`),
     - ešte nebol navštívený:
       * označíme ho ako navštívený.
       * pridáme ho do fronty spolu s časom `aktualny_cas + 1`.
3. Koniec:
   - Ak fronta ostane prázdna a poklad sme nenašli, znamená to, že cesta k nemu neexistuje (vrátime -1).

Komentovaný príklad: Majme nasledovnú mapu ostrova (`mapa.txt`):

```
4 4
LHPH
PPPP
PHPH
PPPX
```

	0	1	2	3
0	L	H	P	H
1	P	P	P	P
2	P	H	P	H
3	P	P	P	X



Mapa ostrova obsahuje extra označenia riadkov a stĺpcov pre jednoduchšiu orientáciu a zodpovedajúci graf je nakreslený vedľa mapy.

Hľadanie cesty k pokladu začína na políčku so súradnicami (0,0). Priebeh hľadania najkratšej cesty k pokladu najlepšie pochopíme, keď popíšeme jednotlivé kroky algoritmu BFS. Priebeh algoritmu je zobrazený v nasledujúcej tabuľke.

Krok	Operácia	Fronta
0	Začíname v (0, 0).	[(0, 0)]
1	Z (0,0) môžeme ísť len dole na (1, 0)	[(1,0)]
2	• Z (1,0) môžeme ísť na (1,1) • Z (1,0) môžeme ísť na (2,0)	[(1,1), (2,0)]
3	• Z (1,1) môžeme ísť na (1,2) • Z (2,0) môžeme ísť na (3,0)	[(1,2), (3,0)]
4	• Z (1,2) môžeme ísť na (1,3) • Z (1,2) môžeme ísť na (2,2) • Z (3,0) môžeme ísť na (3,1)	[(1,3), (2,2), (3,1)]
5	• Z (1,3) už nemôžeme ísť nikam • Z (2,2) môžeme ísť na (3,2) • Z (3,1) už nemôžeme ísť nikam	[(3,2)]
6	• Z (3,2) môžeme ísť na (3,3)	`STOP`: našli sme poklad na súradnici (3,3)

Implementácia v Pythone

Zdrojový kód programu v Pythone, ktorý vypočíta najkratšiu cestu z políčka s pirátskou loďou k pokladu je na nasledujúcej strane. Ak nie je možné dostať sa k pokladu, výstupom bude -1.

```
from collections import deque

def najdi_poklad(cesta_k_suboru):
    # Načítanie mapy zo súboru
    with open(cesta_k_suboru, 'r') as f:
        r, c = map(int, f.readline().split())
        mapa = [list(line.strip()) for line in f.readlines()]

    start = None
    # Nájdenie štartovacieho bodu (L)
    for i in range(r):
        for j in range(c):
            if mapa[i][j] == 'L':
                start = (i, j)
                break

    # Fronta pre BFS: (riadok, stĺpec, aktuálny_čas)
    fronta = deque([(start[0], start[1], 0)])
    navstivene = {start}

    while fronta:
        curr_r, curr_c, cas = fronta.popleft()

        # Kontrola, či sme našli poklad
        if mapa[curr_r][curr_c] == 'X':
            return cas

        # Možné pohyby (hore, dole, vpravo, vľavo)
        for dr, dc in [(-1, 0), (1, 0), (0, -1), (0, 1)]:
            nr, nc = curr_r + dr, curr_c + dc

            # Kontrola hraníc mapy a či je pole priechodné (nie H)
            if 0 <= nr < r and 0 <= nc < c and mapa[nr][nc] != 'H':
                if (nr, nc) not in navstivene:
                    navstivene.add((nr, nc))
                    fronta.append((nr, nc, cas + 1))

    return -1 # Ak sa k pokladu nedá dostať

# Spustenie
print(najdi_poklad('mapa.txt'))
```

Na nasledujúcej strane je ešte niekoľko príkladov vstupov a výstupov uvedeného programu.

Vstup:		Výstup:
	5 5 PPPHP XHPPP HPPPP PPPHP PPLPP	7

Vstup:		Výstup:
	8 8 PPPPPPPP PHPPPPPP PPPHPPPP LPPPPPPP PPPPPHPP PPHPPPPH PPPPHPPP PHPPPPPX	11

Vstup:		Výstup:
	10 10 PPHPPPPPPP PPPPPPPPPP PPHPHLHPPP PPPPPHPPPP PPHPPPPPHP PPPPPPHPPP HPHPPPPPPP PPPPHPPPPP PPPPPPPPPP XHPPPPPPPP	14

16 bodov

♡

5B. Pátranie po poklade (Hard): Pirátska posádka sa práve vylodila na opustenom ostrove, aby našla bájny poklad. Ostrov sa dá zjednodušene predstaviť ako mriežka (graf), kde každá bunka mriežky predstavuje určitý typ terénu. Existujú 3 typy terénu:

- Piesok: Cesta, ktorá je priechodná a dá sa cez ňu voľne pohybovať.
- Džungľa: Priechodná, ale pohyb cez ňu je spomalený.
- Hory: Úplne nepriechodné.

Piráti môžu spraviť len jeden krok naraz, a to buď horizontálne, alebo vertikálne (diagonálny pohyb nie je možný). Jeden krok cez piesok trvá 1 minútu. Jeden krok cez džungľu trvá 2 minúty.

Úloha: Tvojou úlohou je napísať v Pythone program, ktorý nájde najrýchlejšiu cestu z východiskového bodu (pirátskej lode) k pokladu.

Vstupné údaje: Vstupné údaje budú zadané v súbore `mapa.txt`. Prvý riadok bude obsahovať rozmery mapy `R` (počet riadkov) a `C` (počet stĺpcov), oddelené medzerou. Nasleduje `R` riadkov, pričom každý z nich má `C` znakov, ktoré predstavujú mapu ostrova.

`D` – Džungľa

`H` – Hory

`P` – Piesok

`X` – Poklad (**cieľ**)

`L` – Pirátska loď (**východiskový bod**)

Príklad súboru `mapa.txt`:

```
5 5
PLPHP
PHDDP
PPPHH
PDPDH
PPDXH
```

Výstupné údaje: Výstupom bude jedno číslo, napríklad `8`, ktoré predstavuje minimálny čas, za ktorý je možné dosiahnuť poklad. Ak nie je možné dostať sa k pokladu, výstupom bude číslo `-1`. Výstup, t. j. najrýchlejšia cesta, sa uloží do súboru `cesta.txt`.

Riešenie: Tento typ úlohy je ideálne riešiť pomocou Dijkstrovho algoritmu, ktorý hľadá najkratšiu vzdialenosť v ohodnotenom grafe. Na mriežku s jednotlivými políčkami sa pozeráme ako na jednotlivé vrcholy grafu, ktoré sú vzájomne prepojené. Do grafu nezahrnieme políčka, ktoré predstavujú hory, nakoľko tie sú nepriechodné. Počet minút, ktorý zaberie prechod jedným políčkom v mriežke na ceste k pokladu je hodnotou hrany, ktorá prepája dva susedné vrcholy. Štartovacím vrcholom je políčko pirátskej lode (`L`). Cieľovým vrcholom je políčko s pokladom (`X`).

Postup

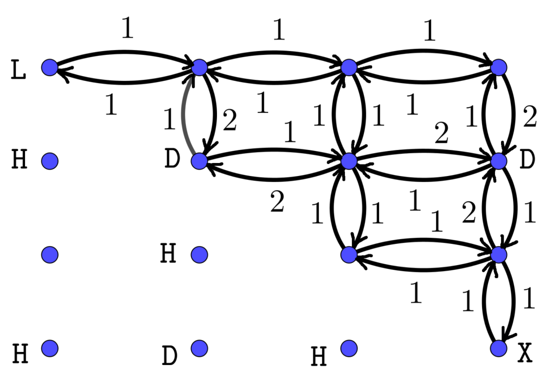
1. **Prioritná fronta**: Budeme udržiavať frontu políčok, ktoré čakajú na spracovanie, zoradenú podľa doteraz nazbieraného času (od najmenšieho).
2. **Tabuľka vzdialeností**: Budeme si viesť záznam o minimálnom čase potrebnom na dosiahnutie každého políčka `[r, c]`, na ktoré sa dá vstúpiť. Na začiatku nastavíme všetko na nekonečno, okrem lode `L`, ktorá má čas 0.
3. **Iterácia**:
   - Vezmeme políčko s najmenším časom z fronty.
   - Preskúmame jeho 4 susedov (hore, dole, vľavo, vpravo).
   - Ak je sused priechodný (nie je to políčko `H`) a nájdeme cestu k nemu, ktorá je kratšia ako doteraz známa, aktualizujeme čas a pridáme suseda do fronty.
   - Cena pohybu do susednej bunky: Piesok `P` = 1, džungľa `D` = 2, poklad `X` = 1, rovnako ako piesok.

Komentovaný príklad: Majme nasledovnú mapu ostrova (`mapa.txt`):

```
4 4
LPPP
HDPD
PHPP
HDHX
```

Mapa ostrova obsahuje extra označenia riadkov a stĺpcov pre jednoduchšiu orientáciu. Mapu ostrova a jej zodpovedajúci (orientovaný) graf vidíme na nasledujúcom obrázku.

	0	1	2	3
0	L	P	P	P
1	H	D	P	D
2	P	H	P	P
3	H	D	H	X



Hľadanie cesty k pokladu začína na políčku so súradnicami (0,0). Priebeh hľadania najkratšej cesty k pokladu najlepšie pochopíme vizualizáciou stavov prioritnej fronty po jednotlivých iteráciách (krokoch). Prioritná fronta obsahuje trojice `(c,r,s)`, kde `c` je celkový čas cestovania na toto políčko, `r` je riadok a `s` je stĺpec.

Krok	Vybrané z fronty (aktuálny vrchol)	Susedia / Operácia	Stav prioritnej fronty
0	–	Štart na (0,0)	[(0, 0, 0)]
1	(0, 0, 0)	Sused (0,1) je `P` (+1)	[(1, 0, 1)]
2	(1, 0, 1)	Sused (0,2) je `P` (+1); (1,1) je `D` (+2)	[(2, 0, 2), (3, 1, 1)]
3	(2, 0, 2)	Sused (0,3) je `P` (+1); (1,2) je `P` (+1)	[(3, 0, 3), (3, 1, 1), (3, 1, 2)]
4	(3, 0, 3)	Sused (1,3) je `D` (+2)	[(3, 1, 1), (3, 1, 2), (5, 1, 3)]
5	(3, 1, 1)	Susedia sú Hory alebo už navštívené	[(3, 1, 2), (5, 1, 3)]
6	(3, 1, 2)	Sused (2,2) je `P` (+1); (1,3) je `D` (+2)	[(4, 2, 2), (5, 1, 3)]
7	(4, 2, 2)	Sused (2,3) je `P` (+1)	[(5, 1, 3), (5, 2, 3)]
8	(5, 1, 3)	Susedia sú Hory alebo už navštívené	[(5, 2, 3)]
9	(5, 2, 3)	Sused (3,3) je Poklad (+1)	[(6, 3, 3)]
10	(6, 3, 3)	Dosiahnutý cieľ (`X`)	[ ]

- **Výsledok:** Najkratšia cesta k pokladu trvá 6 minút.
- **Výsledná cesta pirátov:** $(0,0)$ loď $\rightarrow$ $(0,1)$ piesok $\rightarrow$ $(0,2)$ piesok $\rightarrow$ $(1,2)$ piesok $\rightarrow$ $(2,2)$ piesok $\rightarrow$ $(2,3)$ piesok $\rightarrow$ $(3,3)$ poklad.

Implementácia v Pythone

Zdrojový kód programu v Pythone, ktorý vypočíta najkratšiu cestu z políčka s pirátskou loďou k pokladu, je na nasledujúcej strane. Ak nie je možné dostať sa k pokladu, výstupom bude -1.

```python
import heapq

def najdi_poklad(subor):
    with open(subor, "r") as f:
        riadky = f.readlines()

    R, C = map(int, riadky[0].split())
    mapa = [riadok.strip() for riadok in riadky[1:]]

    start = None
    koniec = None

    # Nájdenie súradníc štartu (L) a cieľa (X)
    for r in range(R):
        for c in range(C):
            if mapa[r][c] == "L":
                start = (r, c)
            elif mapa[r][c] == "X":
                koniec = (r, c)

    # Prioritný rad pre Dijkstru: (cas, r, c)
    pq = [(0, start[0], start[1])]
    navstivene = {start: 0}

    while pq:
        cas, r, c = heapq.heappop(pq)

        if (r, c) == koniec:
            return cas

        # Susedia: hore, dole, vľavo, vpravo
        for dr, dc in [(-1, 0), (1, 0), (0, -1), (0, 1)]:
            nr, nc = r + dr, c + dc

            if 0 <= nr < R and 0 <= nc < C and mapa[nr][nc] != "H":
                # Cena pohybu: Džungľa = 2, Piesok/Poklad = 1
                cost = 2 if mapa[nr][nc] == "D" else 1
                novy_cas = cas + cost

                if (nr, nc) not in navstivene or novy_cas < navstivene[(nr, nc)]:
                    navstivene[(nr, nc)] = novy_cas
                    heapq.heappush(pq, (novy_cas, nr, nc))

    return -1  # Ak cesta neexistuje

# Spustenie
vysledok = najdi_poklad("mapa.txt")
print(vysledok)
```

Tu je ešte niekoľko príkladov vstupov a výstupov uvedeného programu.

Vstup:
```
5 5
PLPHP
PHDDP
PPPHH
PDPDH
PPDXH
```
Výstup: 8

Vstup:
```
10 8
PDPPPPHP
PDPPHXHP
PDPDHHHP
PPDDPPPP
PPDPPPPP
PPDPPPPP
PPDPPPPP
PPLDPPPP
PPPPPPPP
PDDDPPPP
```
Výstup: 14

Vstup:
```
15 15
LPPPPPPPPPPPPPP
PDPHHPDPPPPPPPP
PPPPPPPPPPPPPPP
DHPPPDPPPPPDPPP
PDPPPDPPPPPPPPP
HPHPDHPPHPPHHHP
PPPPPPPPPPPPPPP
HHDPPPPDPPHDPPH
PPPPPPPPPPPPPPP
PDPDHPPPHDPPHPP
PDPDPPPPDPPPPPP
PPPPHHHHHHPPPPP
PPPPPDDDPPPPPPP
PPPDPPDPPHDPPHP
PDPPDPPPHPDPPPX
```
Výstup: 28

Spoločný úvod k úloham 6A. a 6B.

V úlohách 6A. a 6B. máme text zašifrovaný jednoduchou zámenou (substitúciou). Je to najtriviálnejšia možná substitučná šifra, v ktorej sa znak zamieňa za iný znak a rovnakým znakom otvoreného textu zodpovedajú rovnaké znaky zašifrovaného textu. Táto šifra sa používala už od staroveku a jej obrovskou slabinou je to, že zachováva štatistické vlastnosti pôvodného textu. Zachováva jeho dĺžku, zachováva frekvenciu znakov, zachováva frekvenciu bigramov (dvojice po sebe idúcich znakov) a vo všeobecnosti aj frekvenciu n-gramov. Všetky uvedené štatistické vlastnosti, okrem dĺžky textu, ktorá sa môže meniť, sú pre daný jazyk charakteristické, a preto sa dajú využiť pri lúštení textov zašifrovaných jednoduchou zámenou. Toto bolo známe pravdepodobne už v staroveku a písomne je to doložené od 8. až 9. storočia nášho letopočtu. Takže najneskôr od 9. storočia musí byť táto šifra považovaná za nebezpečnú („prelomenú“).

V úlohách 6A. a 6B. nepotrebujeme používať zložitejšie štatistické vlastnosti ako sú frekvencie bigramov a vyšších n-gramov, alebo index koincidencie, alebo entropiu. V týchto úlohách sú zabudované úmyslené kryptografické chyby tak, aby to uľahčilo lúštenie. Preto vystačíme s frekvenciou znakov. Tabuľku frekvencií znakov pre slovenčinu, češtinu, nemčinu a angličtinu máte uvedenú na nasledujúcej strane. Z tabuľky 1 je vidno, že nie každý znak sa vyskytuje v každom jazyku. Napr. slovenčina nemá „ostré S“ ß a nemčina zasa nemá znak ň a ďalšie slovenské znaky s diakritikou. Znaky s diakritikou, aj keď sa to môže zdať prekvapivé[1], výrazným spôsobom uľahčujú lúštenie, a preto sa v šifrovaných textoch nepoužívajú. To znamená, že napríklad v slovenčine sú písmená S a Š reprezentované len znakom S atď. Na strane 18 je uvedená tabuľka frekvencií znakov získaná z románu *Demokrati*, tentoraz už bez diakritiky. Naviac sú v tejto tabuľke uvedené aj frekvencie znakov na začiatkoch a koncoch slov, čo je ďalšia charakteristická štatistická vlastnosť jazyka, avšak my ju nebudeme potrebovať. Nám z tabuľky 2 stačí prvý stĺpec uvádzajúci frekvenciu znakov bez diakritiky v slovenčine.

Pri riešení kryptografických úloh 6A. a 6B. budeme potrebovať zistiť početnosti jednotlivých znakov. Toto sa v predpočítačovej ére robilo ručne. My si môžeme pomôcť nejakým vhodným nástrojom. Jedným z najuniverzálnejších nástrojov na „hranie sa“ s rôznymi šiframi a zisťovanie základných štatistických vlastností textu je program `CrypTool`. Tento je dostupný v dvoch verziách. Pre naše potreby je vhodnejšia a jednoduchšia na použitie staršia verzia programu `CrypTool 1`, z ktorej pochádzajú aj screenshoty v riešeniach. Novšia verzia programu `Cryptool 2` má síce modernejšie rozhranie a obsahuje viac nástrojov a rôznych šifier, avšak na tak jednoduché účely ako potrebujeme my, je náročnejšia na použitie.

[1] Niekto by si mohol myslieť, že čím viac znakov, tým komplikovanejšie lúštenie, ale opak je pravdou.

znak	EN	DE	SK	CZ
a	6.554 %	4.414 %	8.154 %	6.312 %
á	0.000 %	0.000 %	1.483 %	1.718 %
ä	0.000 %	0.464 %	0.081 %	0.000 %
b	1.196 %	1.596 %	1.560 %	1.436 %
c	2.087 %	2.600 %	1.659 %	2.033 %
č	0.000 %	0.000 %	1.002 %	0.647 %
d	3.663 %	4.560 %	2.889 %	3.093 %
ď	0.000 %	0.000 %	0.188 %	0.043 %
e	10.485 %	14.534 %	6.770 %	6.997 %
é	0.000 %	0.000 %	0.458 %	0.729 %
ě	0.000 %	0.000 %	0.000 %	1.182 %
f	1.918 %	1.318 %	0.189 %	0.160 %
g	1.566 %	2.603 %	0.214 %	0.142 %
h	5.505 %	4.257 %	2.080 %	2.089 %
i	5.420 %	6.754 %	4.934 %	3.584 %
í	0.000 %	0.000 %	0.774 %	1.976 %
j	0.109 %	0.164 %	1.394 %	1.979 %
k	0.524 %	0.971 %	3.124 %	3.327 %
l	3.090 %	3.018 %	4.120 %	4.501 %
ľ	0.000 %	0.000 %	0.388 %	0.000 %
ĺ	0.000 %	0.000 %	0.011 %	0.000 %
m	2.041 %	2.166 %	2.883 %	2.830 %
n	5.545 %	8.695 %	4.512 %	4.883 %
ň	0.000 %	0.000 %	0.154 %	0.037 %
o	6.106 %	1.892 %	7.708 %	6.475 %
ó	0.000 %	0.000 %	0.044 %	0.023 %
ô	0.000 %	0.000 %	0.135 %	0.000 %
ö	0.000 %	0.245 %	0.000 %	0.000 %
p	1.359 %	0.544 %	2.541 %	2.566 %
q	0.102 %	0.039 %	0.003 %	0.000 %
r	4.908 %	5.823 %	3.764 %	2.696 %
ŕ	0.000 %	0.000 %	0.018 %	0.000 %
ř	0.000 %	0.000 %	0.000 %	0.826 %
s	5.155 %	5.570 %	3.842 %	3.707 %
š	0.000 %	0.000 %	0.802 %	0.858 %
ß	0.000 %	0.378 %	0.000 %	0.000 %
t	7.409 %	4.994 %	3.658 %	4.323 %
ť	0.000 %	0.000 %	0.631 %	0.045 %
u	2.254 %	3.094 %	2.223 %	2.624 %
ú	0.000 %	0.000 %	0.601 %	0.069 %
ů	0.000 %	0.000 %	0.000 %	0.258 %
ü	0.000 %	0.527 %	0.000 %	0.000 %
v	0.816 %	0.694 %	3.403 %	3.427 %
w	1.869 %	1.310 %	0.010 %	0.019 %
x	0.132 %	0.633 %	0.020 %	0.014 %
y	1.403 %	0.127 %	1.234 %	1.701 %
ý	0.000 %	0.000 %	0.725 %	0.647 %
z	0.054 %	0.960 %	1.677 %	1.598 %
ž	0.000 %	0.000 %	0.829 %	1.084 %
␣	18.732 %	15.621 %	17.113 %	17.351 %

Tabuľka 1: Frekvencie znakov angličtiny, nemčiny, slovenčiny a češtiny

Relatívne početnosti znakov v románe *Demokrati*					
Znak	Všeobecne	Znak	Začiatky slov	Znak	Konce slov
A	11.93 %	**S**	12.26 %	**A**	19.47 %
O	9.00 %	**P**	9.56 %	**E**	12.75 %
E	8.77 %	**N**	9.24 %	**O**	10.91 %
I	7.08 %	**A**	7.59 %	**I**	9.99 %
N	5.94 %	**V**	6.66 %	**Y**	7.02 %
S	5.59 %	**Z**	6.08 %	**U**	6.94 %
T	5.27 %	**T**	6.06 %	**L**	5.92 %
L	5.22 %	**K**	5.14 %	**M**	5.26 %
R	4.45 %	**M**	4.81 %	**T**	3.69 %
K	4.27 %	**D**	4.58 %	**K**	2.75 %
V	3.99 %	**C**	4.12 %	**S**	2.05 %
D	3.84 %	**B**	3.83 %	**V**	2.03 %
U	3.38 %	**O**	3.64 %	**N**	2.02 %
M	3.38 %	**J**	3.05 %	**H**	1.97 %
C	3.28 %	**H**	2.99 %	**D**	1.90 %
P	2.87 %	**R**	2.62 %	**J**	1.73 %
Z	2.80 %	**L**	2.47 %	**Z**	1.58 %
Y	2.57 %	**U**	2.18 %	**C**	0.97 %
H	2.36 %	**I**	2.00 %	**R**	0.86 %
B	1.92 %	**E**	0.46 %	**B**	0.07 %
J	1.70 %	**F**	0.41 %	**F**	0.06 %
F	0.19 %	**G**	0.22 %	**P**	0.06 %
G	0.15 %	**Y**	0.00 %	**G**	0.03 %
X	0.01 %	**Q**	0.00 %	**X**	0.00 %
Q	0.00 %	**X**	0.00 %	**Q**	0.00 %
W	0.00 %	**W**	0.00 %	**W**	0.00 %

Tabuľka 2: Porovnanie relatívnych početností znakov v románe Janka Jesenského: *Demokrati* všeobecne, na začiatku a na konci slov

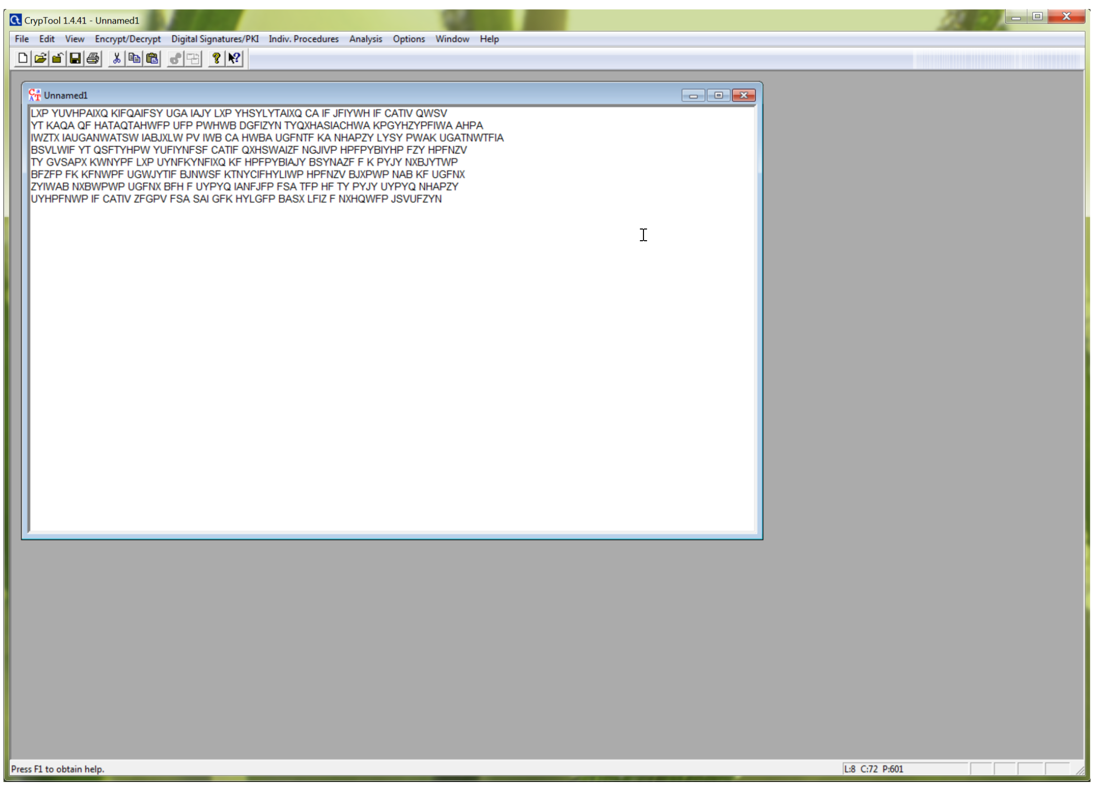
14 bodov

6A. Rozlúštite nasledujúci text a zistite odkiaľ pochádza:

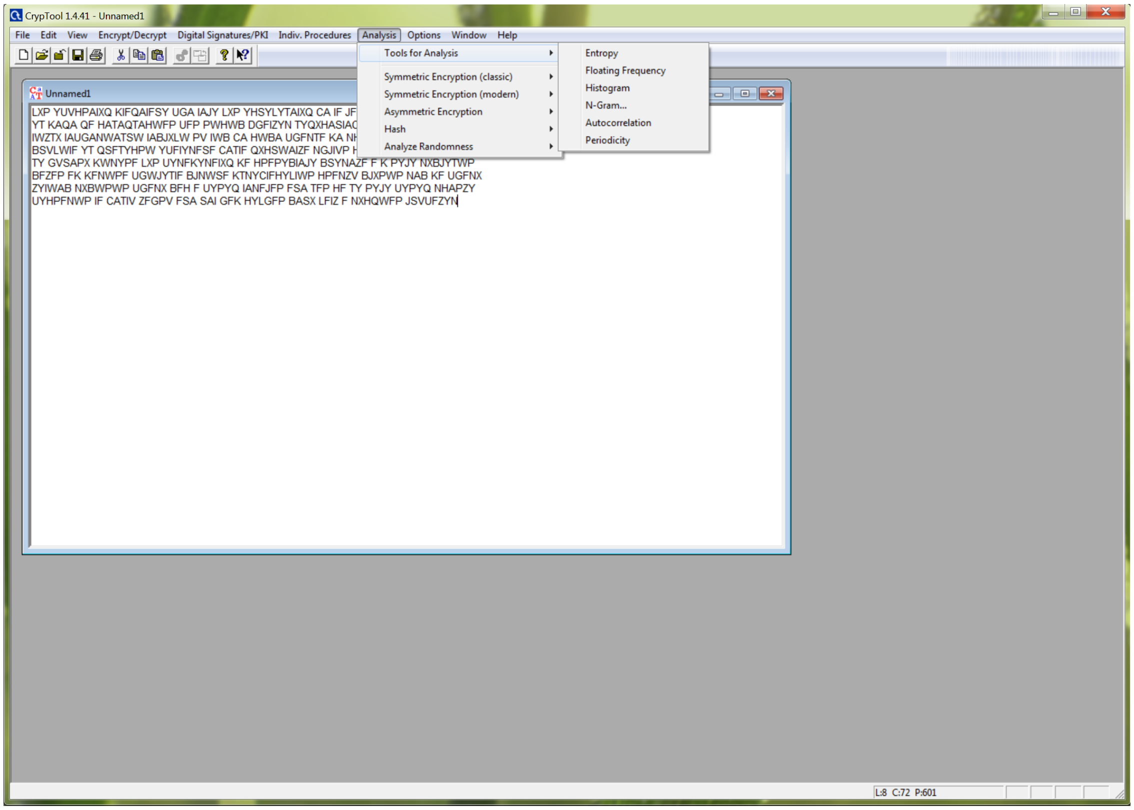
```
LXP YUVHPAIXQ KIFQAIFSY UGA IAJY LXP YHSYLYTAIXQ CA IF JFIYWH IF CATIV QWSV
YT KAQA QF HATAQTAHWFP UFP PWHWB DGFIZYN TYQXHASIACHWA KPGYHZYPFIWA AHPA
IWZTX IAUGANWATSW IABJXLW PV IWB CA HWBA UGFNTF KA NHAPZY LYSY PWAK UGATNWTFIA
BSVLWIF YT QSFTYHPW YUFIYNFSF CATIF QXHSWAIZF NGJIVP HPFPYBIYHP FZY HPFNZV
TY GVSAPX KWNYPF LXP UYNFKYNFIXQ KF HPFPYBIAJY BSYNAZF F K PYJY NXBJYTWP
BFZFP FK KFNWPF UGWJYTIF BJNWSF KTNYCIFHYLIWP HPFNZV BJXPWP NAB KF UGFNX
ZYIWAB NXBWPWP UGFNX BFH F UYPYQ IANFJFP FSA TFP HF TY PYJY UYPYQ NHAPZY
UYHPFNWP IF CATIV ZFGPV FSA SAI GFK HYLGFP BASX LFIZ F NXHQWFP JSVUFZYN
```

Jedná sa o slovensky písaný text bez diakritických a interpunkčných znamienok. Text je zašifrovaný jednoduchou zámenou so zachovanými prirodzenými medzerami.
Upozornenie: Koniec riadku je tiež medzera medzi slovami.

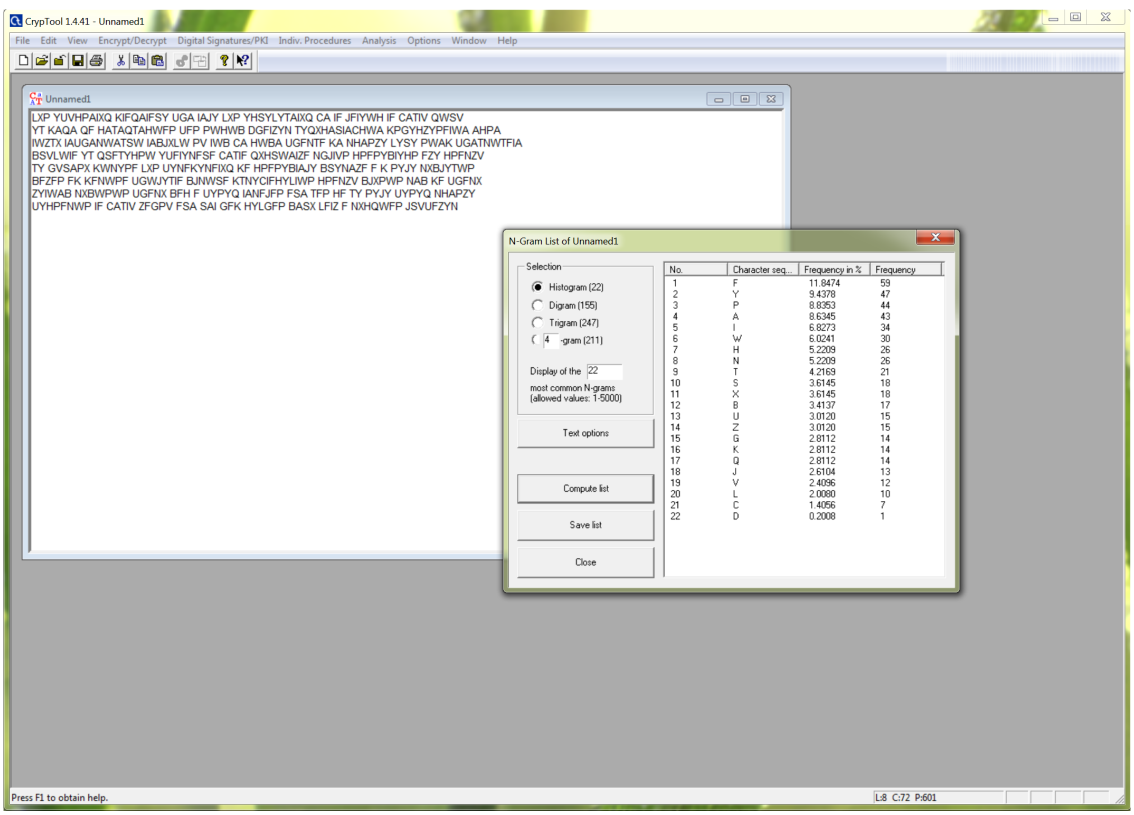
Riešenie: Najväčšou slabinou tejto šifry je to, že zachováva prirodzené medzery. To znamená, že vieme identifikovať jednotlivé slová. Pri lúštení sú pre nás najdôležitejšie krátke slová, pretože tých je v slovenčine málo. Z 1-písmenkových slov to sú len spojky ako A, výnimočne I a zopár predložiek ako sú K, O, U, V, S, Z. Pri 2-písmenkových slovách je možností tiež málo a okrem toho je pri 2-písmenkových slovách vždy jedno písmeno spoluhláska a druhé samohláska. Ak 2-písmenkové slová naviac porovnáme s 1-písmenkovými slovami, tak takmer okamžite budeme vedieť identifikovať niekoľko najčastejších znakov. Otvorme si teda program `CrypTool 1` a načítajme si v ňom zašifrovaný text z úlohy.



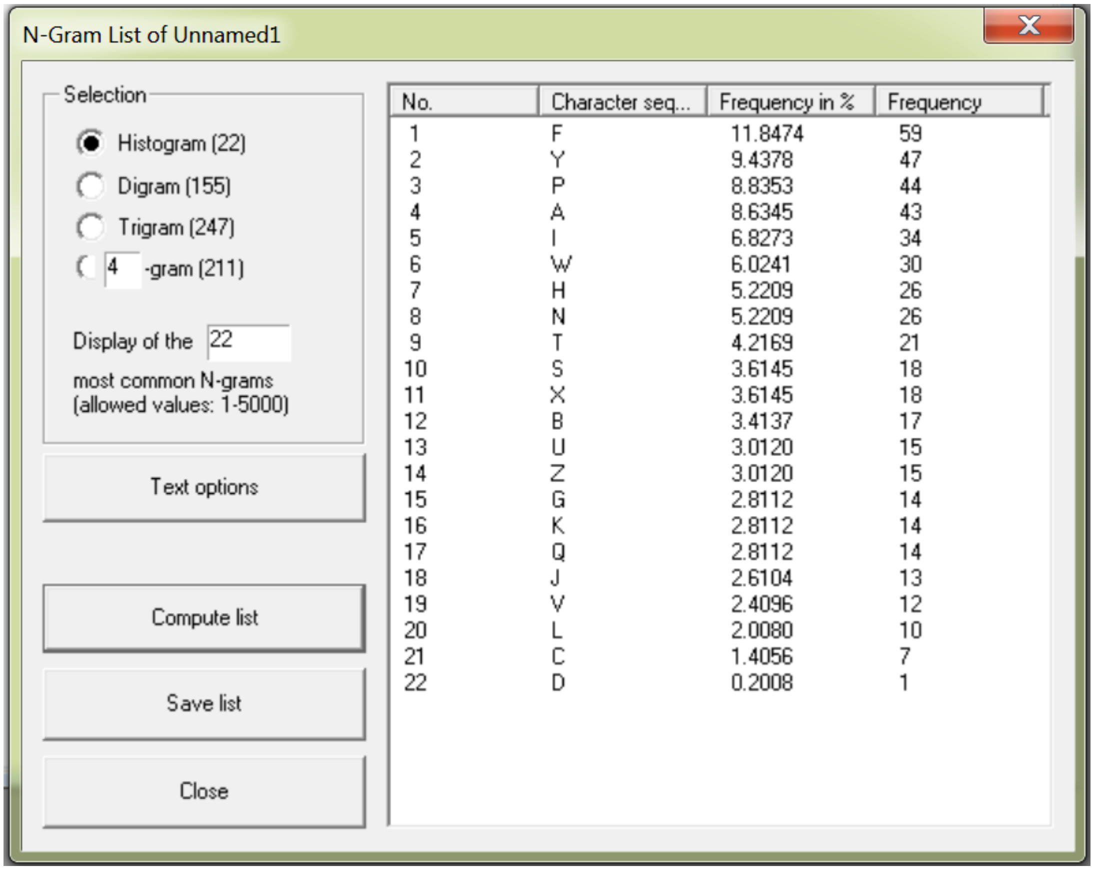
Ďalej si v menu `Analysis` vyberieme `N-Gram...`



a necháme si zobraziť frekvencie jednotlivých znakov.



Tabuľku frekvencií znakov máme lepšie zobrazenú na ďalšej strane.



No.	Character seq...	Frequency in %	Frequency
1	F	11.8474	59
2	Y	9.4378	47
3	P	8.8353	44
4	A	8.6345	43
5	I	6.8273	34
6	W	6.0241	30
7	H	5.2209	26
8	N	5.2209	26
9	T	4.2169	21
10	S	3.6145	18
11	X	3.6145	18
12	B	3.4137	17
13	U	3.0120	15
14	Z	3.0120	15
15	G	2.8112	14
16	K	2.8112	14
17	Q	2.8112	14
18	J	2.6104	13
19	V	2.4096	12
20	L	2.0080	10
21	C	1.4056	7
22	D	0.2008	1

Pozrime sa teraz na to, aké 1-písmenkové slová sú v zašifrovanom texte. Trikrát sa tam vyskytuje F a raz sa tam vyskytuje K. Teraz sa pozrime na relatívne početnosti týchto písmen. F má relatívnu početnosť približne 11,8 % a K má relatívnu početnosť 2,8 %. Pozrime sa ešte na tabuľku 2 a hneď vidíme, že takmer so 100 % istotou bude F=a.

Ďalej budeme skúmať 2-písmenkové slová. Trikrát sa tam vyskytuje IF, dvakrát sa tam vyskytujú CA, KF, YT, TY a raz sa tam vyskytujú FK, HF, KA, PV, QF. Relatívna početnosť I je 6,8 %, čo podla tabuľky znamená, že pravdepodobne IF=na, alebo s trochu menšou pravdepodobnosťou IF=sa. Vyskúšať musíme obe možnosti a potom uvidíme, ktorá z nich je správna.

Zaujímavé sú slová YT a TY. V prvom rade Y má relatívnu početnosť 9,4 % čo takmer s istotou znamená, že Y=o. To potom ale znamená, že YT=od a TY=do. Ďalšia podobná dvojica je FK a KF. Vieme, že skoro iste F=a a aké dvojpísmenkové slová, ktoré obsahujú a a ešte jedno písmeno a vyskytujú sa v oboch možných poradiach, poznáme? Sú to KF=za a FK=az.

Teraz už máme 4 písmená takmer isté F=a, Y=o, T=d a K=z a jedno písmeno buď I=n, alebo I=s. Tieto písmená si v zašifrovanom texte substituujeme a začnú sa nám tam objavovať zmysluplné sekvencie. Samozrejme sústredíme sa stále predovšetkým na dvoj a trojpísmenkové slová, ale aj v ďalších slovách sa nám budú objavovať zmysluplné časti. Toto nebudeme ďalej naťahovať. Na ďalšej strane je rozlúštený otvorený text.

```
byt opustenym znamenalo pre neho byt oslobodenym je na hanois na jednu milu
od zeme ma sedemdesiat pat tisic frankov domyselnejsie ztroskotanie este
nikdy nepreviedli nechybi tu nic je sice pravda ze vsetko bolo tiez predvidane
clubina od mladosti opanovala jedna myslienka vrhnut statocnost ako stavku
do rulety zivota byt povazovanym za statocneho cloveka a z toho vychodit
cakat az zavita prihodna chvila zdvojnasobnit stavku chytit vec za pravy
koniec vycitit pravy cas a potom nevahat ale dat sa do toho potom vsetko
postavit na jednu kartu ale len raz sobrat cely bank a vysmiat hlupakov
```

Text je písaný archaickou slovenčinou a vyhľadávaním na internete rýchlo zistíme, že sa jedná o pasáž z románu *Robotníci mora* od Viktora Huga, ktorý by sme mali poznať aj z literatúry na základnej (strednej?) škole.

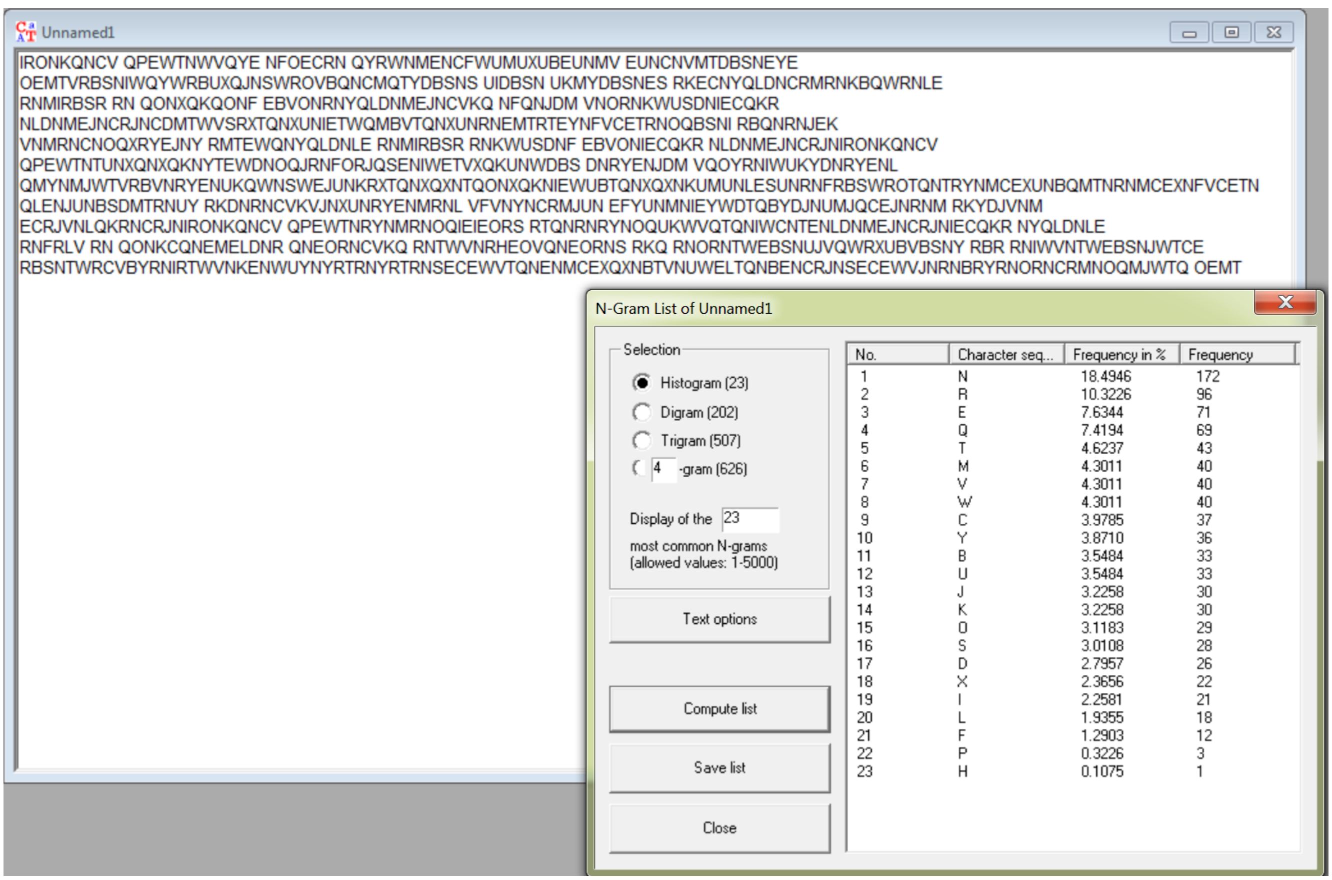
14 bodov

6B. Rozlúštite nasledujúci text a zistite odkiaľ pochádza:

```
IRONKQNCV  QPEWTNWVQYE NFOECRN QYRWNMENCFWUMUXUBEUNMV EUNCNVMTDBSNEYE
OEMTVRBSNIWQYWRBUXQJNSWROVBQNCMQTYDBSNS UIDBSN UKMYDBSNES RKECNYQLDNCRMRNKBQWRNLE
RNMIRBSR RN QONXQKQONF EBVONRNYQLDNMEJNCVKQ NFQNJDM VNORNKWUSDNIECQKR
NLDNMEJNCRJNCDMTWVSRXTQNXUNIETWQMBVTQNXUNRNEMTRTEYNFVCETRNOQBSNI RBQNRNJEK
VNMRNCNOQXRYEJNY RMTEWQNYQLDNLE RNMIRBSR RNKWUSDNF EBVONIECQKR NLDNMEJNCRJNIRONKQNCV
QPEWTNTUNXQNXQKNYTEWDNOQJRNFORJQSENIWETVXQKUNWDBS DNRYENJDM VQOYRNIWUKYDNRYENL
QMYNMJWTVRBVNRYENUKQWNSWEJUNKRXTQNXQXNTQONXQKNIEWUBTQNXQXNKUMUNLESUNRNFRBSWROTQNTRYNMCEXUNBQMTNRNMCEXNFVCETN
QLENJUNBSDMTRNUY RKDNRNCVKVJNXUNRYENMRNL VFVNYNCRMJUN EFYUNMNIEYWDTQBYDJNUMJQCEJNRNM RKYDJVNM
ECRJVNLQKRNCRJNIRONKQNCV  QPEWTNRYNMRNOQIEIEORS RTQNRNRYNOQUKWVQTQNIWCNTENLDNMEJNCRJNIECQKR NYQLDNLE
RNFRLV RN QONKCQNEMELDNR QNEORNCVKQ RNTWVNRHEOVQNEORNS RKQ RNORNTWEBSNUJVQWRXUBVBSNY RBR RNIWVNTWEBSNJWTCE
RBSNTWRCVBYRNIRTWVNKENWUYNYRTRNYRTRNSECEWVTQNENMCEXQXNBTVNUWELTQNBENCRJNSECEWVJNRNBRYRNORNCRMNOQMJWTQ OEMT
```

Jedná sa o slovensky písaný text bez diakritických a interpunkčných znamienok. Text je zašifrovaný jednoduchou zámenou so zachovanými prirodzenými medzerami avšak aj medzera je šifrovaná. Abecedy otvoreného aj šifrovaného textu pozostávajú teda zo znakov A, B, C, . . . , Z a z medzery ␣.
Upozornenie: Koniec riadku je tiež medzera medzi slovami.

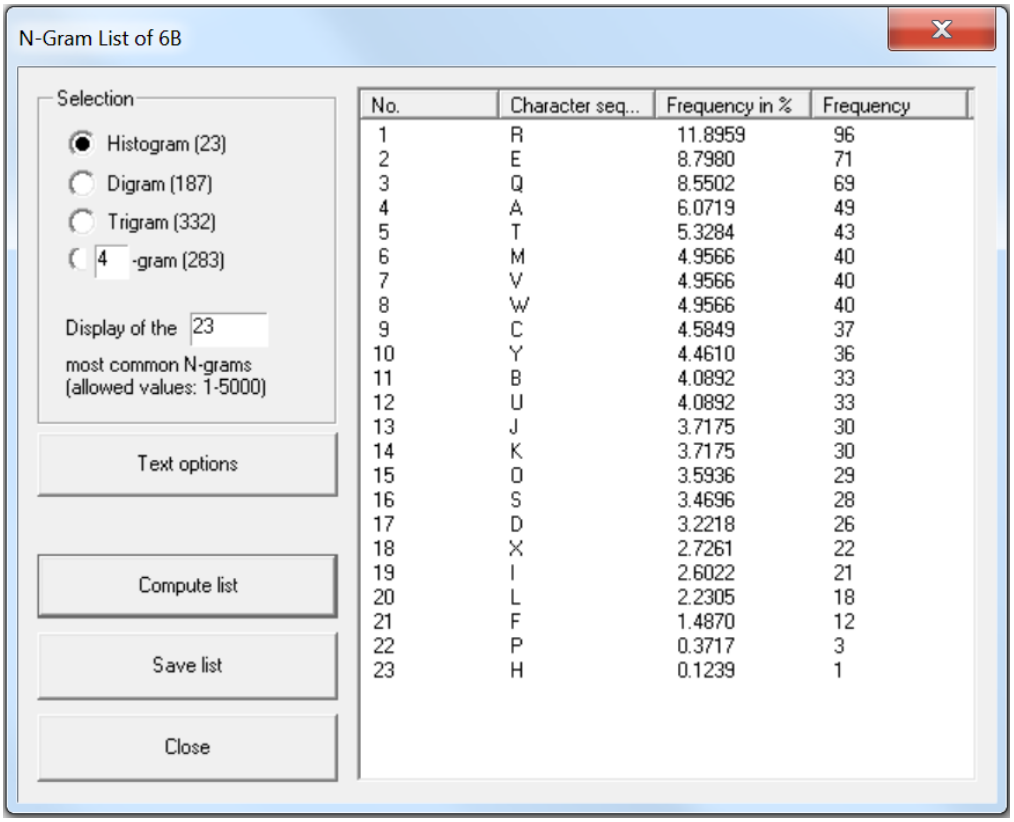
Riešenie: Rovnako ako v predošlej úlohe si pomocou programu `CrypTool 1` zistíme relatívne početnosti jednotlivých znakov.



Táto šifra je zdanlivo komplikovanejšia než šifra z úlohy 6A., pretože nie sú zachované pôvodné medzery. Úmyselná chyba zabudovaná do tejto šifry je tá, že aj samotná medzera je šifrovaná ako znak. Ak si pozrieme relatívne početnosti znakov, tak vidíme, že znak N má relatívnu početnosť približne 18,5 %, čo je ovela viac než ktorékoľvek písmeno abecedy podľa tabuľky 2. Keď sa však pozrieme na tabuľku 1, tak vidíme, že absulútne najčastnejším znakom, nielen v slovenčine, je práve medzera. To znamená, že znak N=␣ a zašifrovaný text vieme prepísať do poboby so zachovanými prirodzenými medzerami. Pôvodné medzery v zašifrovanom texte nahradíme písmenom A, ktoré sa v pôdodnom zašifrovanom texte nevyskytuje a potom písmená N nahradíme medzerami. Zašifrovaný text potom bude mať podobu.

```
IRO KQ CVAAQPEWT WVQYEA FOECR AQYRW ME CFWUMUXUBEU MVAEU C VMTDBS EYEAOEMTVRBS
IWQYWRBUXQJ SWROVBQ CMQTYDBS SAUIDBS AUKMYDBS ESARKEC YQLD CRMR KBQWR LEAR
MIRBSRAR AQO XQKQO FAEBVO R YQLD MEJ CVKQA FQ JDMAV OR KWUSD IECQKRA LD MEJ CRJ
CDMTWVSRXTQ XU IETWQMBVTQ XU R EMTRTEY FVCETR OQBS IARBQ R JEKAV MR C OQXRYEJ
YARMTEWQ YQLD LEAR MIRBSRAR KWUSD FAEBVO IECQKRA LD MEJ CRJ IRO KQ CVAAQPEWT TU
XQ XQK YTEWD OQJR FORJQSE IWETVXQKU WDBSAD RYE JDMAVQOYR IWUKYD RYE LAQMY
MJWTVRBV RYE UKQW SWEJU KRXTQ XQX TQO XQK IEWUBTQ XQX KUMU LESU R FRBSWROTQ TRY
MCEXU BQMT R MCEX FVCET AQLE JU BSDMTR UYARKD R CVKVJ XU RYE MR LAVFV Y CRMJU
AEFYU M IEYWDTQBYDJ UMJQCEJ R MARKYDJV MAECRJV LQKR CRJ IRO KQ CVAAQPEWT RY MR
OQIEIEORSARTQ R RY OQUKWVQTQ IWC TE LD MEJ CRJ IECQKRA YQLD LEAR FRLVAR AQO KCQ
EMELD RAQ EOR CVKQAR TWV RHEOVQ EOR SARKQAR OR TWEBS UJVQWRXUBVBS YARBRAR IWV
TWEBS JWTCEARBS TWRCVBYR IRTWV KE WUY YRTR YRTR SECEWVTQ E MCEXQX BTV UWELTQ BE
CRJ SECEWVJ R BRYR OR CRM OQMJWTQAOEMT
```

Dostali sme zašifrovaný text so zachovanými pôvodnými medzerami, t. j. jedná sa o úlohu rovnakého typu ako v príklade 6A. a pri jej riešení budeme postupovať rovnako ako v predošlom príklade. Ešte raz si vypočítame relatívne početnosti znakov, pretože ako si možno niektorí všimli, program `CrypTool` medzeru ignoruje. My sme medzery v zašifrovanom texte nahradili znakom A, a preto musíme ešte raz zistiť relatívne početnosti znakov. Tieto sú na nasledujúcej strane.



No.	Character seq...	Frequency in %	Frequency
1	R	11.8959	96
2	E	8.7980	71
3	Q	8.5502	69
4	A	6.0719	49
5	T	5.3284	43
6	M	4.9566	40
7	V	4.9566	40
8	W	4.9566	40
9	C	4.5849	37
10	Y	4.4610	36
11	B	4.0892	33
12	U	4.0892	33
13	J	3.7175	30
14	K	3.7175	30
15	O	3.5936	29
16	S	3.4696	28
17	D	3.2218	26
18	X	2.7261	22
19	I	2.6022	21
20	L	2.2305	18
21	F	1.4870	12
22	P	0.3717	3
23	H	0.1239	1

Ďalší postup riešenia bude potom rovnaký ako v úlohe 6A. Opäť začneme jedno, dvoj a trojpísmenkovými slovami a postupne dešifrujeme celý text. Dešifrovaný text je nasledovný.

```
pan de villefort riekol znova lekar so vzrusujucou silou v istych okolnostiach
prekracujem hranice vsetkych hlupych ludskych ohladov keby vasa dcera bola
spachala len jeden zlocin a keby som videl ze mysli na druhy povedal by som vam
vystrihajte ju potrescite ju a ostatok zivota nech place a modli sa v nejakom
klastore keby bola spachala druhy zlocin povedal by som vam pan de villefort tu
je jed ktory nema znameho protijedu rychly ako myslienka prudky ako blesk
smrtiaci ako uder hromu dajte jej ten jed poructe jej dusu bohu a zachrante tak
svoju cest a svoj zivot lebo mu chysta uklady a vidim ju ako sa blizi k vasmu
lozku s pokryteckym usmevom a sladkymi slovami beda vam pan de villefort ak sa
nepoponahlate a ak neudriete prv to by som vam povedal keby bola zabila len dve
osoby ale ona videla tri agonie ona hladela na troch umierajucich klacala pri
troch mrtvolach travicka patri do ruk kata kata hovorite o svojej cti urobte co
vam hovorim a caka na vas nesmrtelnost
```

Vyhľadávaním na internete rýchlo zistíme, že sa jedná o pasáž z románu *Gróf Monte Christo* od Alexandra Dumasa, ktorý by sme tiež mali poznať zo školy.