

Tema 2 - Templul magic

Responsabili:

- Vlad Drăghici
- Andrei-Bogdan Florea
- Eduard Scăueru

Termen de predare:

- Deadline soft: **10.01.2022**
- Deadline hard: **10.01.2022**

Pentru fiecare zi (24 de ore) de întârziere, se vor scădea 10 puncte din nota acordată, până la atingerea deadline-ului hard.

Întrebări

Dacă aveți nelămuriri, puteți să ne contactați pe canalul Temei 2.

La orice întrebare vom răspunde în maxim 24 de ore.

Nu se acceptă întrebări în ultimele 24 de ore înainte de deadline.

Introducere

Este seara de 28.11.2021. Ești la birou și aștepți cu nerăbdare Tema 2 de PCLP care trebuie să apară din moment în moment. Cu gândul la pointeri și la string-uri adormi cu capul pe masă. Fără să îți dai seama, te trezești în fața intrării unui templu antic. Încercător pașești în necunoscut în aventura visului tău.

Organizarea temei

TOATE modificările de cod se vor realiza **DOAR** în fișierele **Task1.c**, **Task2.c**, **Task3.c** și **utils.h**. Modificarea oricărui alt fișier nu va fi luată în considerare.

Urmăriți indicațiile din fișierele **Task1.c**, **Task2.c**, **Task3.c** și completați funcțiile notate cu **TODO**. În fișierele **Task1.c**, **Task2.c** și **Task3.c** se pot implementa și funcții adiționale și în fișierul **utils.h** se pot declara.

Task 1 - Cuvinte magice (40p)

Cum intri în templul magic te trezești într-o cameră uriașă sub forma unei matrici de dimensiune $N * M$. O voce îți spune că te afli în templul lui magic și că pentru a trece la camera următoare trebuie să te miști prin cameră (prin matrice) după un traseu bine stabilit. Pentru a face lucrurile mai interesante, vocea îți spune că traseul necesar trebuie decodificat de către tine apoi îți spune un cod magic.

Pe pereții camerei sunt instrucțiunile pentru a înțelege codul mistic:

1. Un cod descrie traseul pe care trebuie să îl urmezi în cameră.
2. Codul poate fi format din oricâte cuvinte magice separate prin spațiu (fiecare cuvânt descriind un singur pas din traseu).

Cuvintele sunt de 3 tipuri:

1. Cuvintele care încep cu litera 'a' sunt de forma "ax₁x₂x₃x₄" unde x_i sunt cifre ($1 \leq i \leq 4$). Fiecare cifră este corespunzătoare unei direcții de mișcare (x₁ – dreapta, x₂ – sus, x₃ – stânga, x₄ – jos în matrice). Cifra maximă dictează mișcarea din traseu corespunzătoare.

Ex: Codul "a1235" are cifra maximă (5) pe poziția lui x₄, deci următoarea mișcare din traseu este în jos.

2. Cuvintele care încep cu litera 'b' sunt de forma "bx₁x₂...x_n" unde x_i sunt cifre ($1 \leq i \leq n$). Considerăm numărul format din cifrele "x₁x₂...x_n" ca un număr K. Fie X numărul format din ultimele 2 cifre ale lui K (x_{n-1}x_n). În funcție de proprietățile de palindrom și prim ale lui K, respectiv X avem următoarele 4 situații:

1. K palindrom, X prim: stânga
2. K palindrom, X NU este prim: dreapta
3. K NU este palindrom, X prim: sus
4. K NU este palindrom, X NU este prim: jos

Ex: Codul "b121" are K = 121 și X = 21. K este palindrom și X nu este prim \Rightarrow următoarea mișcare din traseu este în dreapta.

3. Cuvinte care încep cu litera 'c' sunt de forma "cnkx₁x₂...x_n", unde n, k, x_i sunt cifre. Fie S suma primelor k cifre luate din k în k în mod circular. $S = x_0 + x_k + x_{2k} + \dots$ (dacă indicele trece peste n atunci se resetează la 0). În funcție de restul împărțirii lui S la 4 avem următoarele situații:

1. $S \% 4 = 0 \Rightarrow$ stânga
2. $S \% 4 = 1 \Rightarrow$ sus
3. $S \% 4 = 2 \Rightarrow$ dreapta
4. $S \% 4 = 3 \Rightarrow$ jos

Ex: Codul "c64123456" are $n = 6$ (6 cifre), $k = 4$ și cifrele 123456. Suma dorită este: $S = x_0 + x_4 + x_8 \% 6 + x_{12} \% 6 = 1 + 5 + 3 + 1 = 10$ Restul împărțirii este: $S \% 4 = 10 \% 4 = 2 \Rightarrow$ următoarea mișcare din traseu este în dreapta.

Task 1.1 - Cuvinte magice tipul 'a' (8p)

Sa se determine traseul corect pentru cuvinte doar de tipul 'a'.

Task 1.2 - Cuvinte magice tipul 'b' (8p)

Sa se determine traseul corect pentru cuvinte doar de tipul 'b'.

Task 1.3 - Cuvinte magice tipul 'c' (8p)

Sa se determine traseul corect pentru cuvinte doar de tipul 'c'.

Task 1.4 - Cuvinte magice de orice tip (16p)

Sa se determine traseul corect pentru orice tip de cuvinte.

Input

Pe prima linie se află N și M (dimensiunile matricei). Pe a doua linie se află codul magic.

Output

Matricea cu traseul corect. Prima mișcare este mereu intrarea în cameră în colțul stânga sus și se notează cu 1 ($matrix[0][0] = 1$). Următoarele mișcări se descifrează din codul magic. După fiecare mișcare trebuie să notezi în matrice a câta mișcare a dus în poziția actuală ($matrix[poziție_curentă_i][poziție_curentă_j] = nr. \text{ mișcare actuală}$).

Exemplu

```
Input:
4 4
a1235 b121 c64123456
Output:
1 0 0 0
2 3 4 0
0 0 0 0
0 0 0 0
```

Task 2 - Vocea stranie (50p)

La ieșirea din labirint, auzi din nou aceeași voce stranie, însă, de data aceasta, nu mai poți înțelege cuvintele pe care le rostește. Brusc, îți dai seama că mesajele pe care vocea le spune sunt criptate. Fiind un bun programator, ai decis să încerci să le descifrezi.

Task 2.1 (20p)

Prima variantă la care te gândești, cifrul Caesar, obține un text criptat pe baza unui mesaj (denumit plaintext) și a unei chei (un număr întreg) în modul următor: fiecare literă din mesaj este "rotită" cu valoarea specificată de cheie. Spre exemplu, șirul "abCZ" rotit cu cheia 3 devine "deFC". În plus față de această funcționalitate clasică, vom considera că și cifrele se pot "roti". Adică, șirul "1239" rotit cu cheia 3 devine "4562". Descifrarea este întocmai operația opusă: pornind de la un text criptat, pentru care se cunoaște cheia cu care a fost criptat, se poate afla mesajul inițial.

Se cere să se afișeze **mesajul inițial**, obținut în urma decriptării textului criptat, cunoscându-se cheia de criptare. În acest sens, se dă un input de forma:

```
2
caesar
K
S
.....
K -> cheia cu care a fost criptat șirul
S -> textul criptat
```

Task 2.2 (10p)

Cel de-al doilea cifru, Vigenère, se bazează tot pe un mesaj și o cheie, care de data aceasta este un șir de majuscule. Pentru a obține textul criptat, se repetă cheia de-a lungul mesajului, iar fiecare literă din mesaj este "rotită" pe baza poziției în alfabet a literei corespunzătoare din cheie. În mod asemănător cu cerința anterioară, și cifrele se vor "roti" în funcție de litera corespunzătoare din cheie. Exemplu de criptare:

```
Mesaj: abc1234
Cheie: ABC
Text criptat: ace1354
```

Explicatie:

```
Mesaj:      abc1234
```

```
Cheie extinsă: ABCABCA
```

```
A are poziția 0 => caracterele corespunzătoare lui A nu se vor modifica
(a->a, 1->1, 4->4)
```

```
B are poziția 1 => caracterele corespunzătoare lui B se deplasează cu 1
(b->c, 2->3)
```

```
C are poziția 2 => caracterele corespunzătoare lui C se deplasează cu 2
(c->e, 3->5)
```

Se cere să se afișeze **mesajul inițial**, obținut în urma decriptării unui text criptat, cunoscându-se cheia de criptare. Se va primi un input de forma:

```
2
vigenere
K
```

S

.....

K -> cheia cu care a fost criptat şirul

S -> textul criptat

Precizări pentru ambele cifruri:

- Atât alfabetul, cât şi mulţimea de cifre $\{0, \dots, 9\}$ se consideră circulare. Spre exemplu, în cazul cifrului Caesar, litera "b" decriptată cu cheia 2 va fi "z" ($b \rightarrow a \rightarrow z$). Similar, cifra "2" decriptată cu cheia 4 va fi "8" ($2 \rightarrow 1 \rightarrow 0 \rightarrow 9 \rightarrow 8$).
- Şirurile criptate date ca input vor conţine DOAR litere mici, majuscule şi cifre.
- În urma decriptării, literele mici din textul criptat vor rămâne litere mici, iar majusulele din textul criptat vor rămâne tot majuscule.
- Şirul criptat şi mesajul iniţial vor avea lungimea de maxim 1000 de caractere.
- Cheia folosită pentru cifrul Vigenere va avea lungimea de maxim 10 caractere.

Task 2.3 (20p)

Ai realizat că vocea foloseşte cifrul Caesar pentru criptarea mesajelor, aşa că atunci când îţi spune două numere mari, te gândeşti să calculezi suma lor, după ce le decriptezi. Deci, cerinţa constă în adunarea a două numere mari, care trebuie stocate sub forma de şir de caractere. Numerele sunt iniţial criptate folosind cifrul Caesar descris la cerinţa 2.1 şi trebuie decriptate înainte de a calcula suma lor. Se va primi un input de forma:

2

addition

K

N1

N2

.....

K -> cheia cu care au fost criptate numerele

N1, N2 -> cele două numere mari, criptate cu cheia K

Se cere să se afişeze suma celor două numere mari, după ce au fost decriptate.

Exemplu:

K = 3, N1 = "123", N2 = "456" => rezultatul este "1013"

Explicaţie:

"123" decriptat cu K = 3 este "890"

"456" decriptat cu $K = 3$ este "123"
"890" + "123" = "1013"

Numerele vor fi **obligatoriu** stocate sub forma de șiruri de caractere. Nu se vor acorda cele 20 de puncte dacă numerele sunt citite/convertite cu totul în tipuri numerice (int, long long, etc.), pentru a efectua simpla adunare a acestora.

Precizări:

- În urma decriptării, un număr dat la intrare poate începe cu zerouri; rămâne la latitudinea voastră dacă să le eliminați sau nu. Rezultatul final nu trebuie, totuși, să înceapă cu zerouri.
- Numerele primite ca input și suma lor vor avea maxim 1000 de caractere lungime.

Task 3 - Ajutorul (50p)

Fiindcă ai reușit să descifrezi și să rezolvi provocările anterioare, vocea decide să îți ceară ajutorul în finalizarea unui algoritm de prezicere a cuvintelor. Vocea îți spune ca deja cunoaște frecvențele cuvintelor într-un text, dar are nevoie de tine pentru calcularea numărului de apariții a 2-gramelor, în ordinea apariției lor. O *n-gramă* reprezintă un grup de n elemente (cuvinte, silabe, litere, etc.) consecutive dintr-un text. Pentru acest task vom trata n -gramele ca fiind grupuri de n cuvinte consecutive. Un cuvânt este separat de spații sau newline, iar punctuația precum ',', '.', '!', ';' este ignorată. De asemenea, input-ul este case sensitive și nu este nevoie de a transforma literele mari în mici.

Astfel, un exemplu ar consta în următorul sample test:

```
in:
3
The weather is nice, the sun is nice too.
The sun is yellow!
out:
10
The weather 1
weather is 1
is nice 2
nice the 1
the sun 1
sun is 2
nice too 1
too The 1
The sun 1
    is yellow 1
```

Vă puteți folosi de output-ul generat de voi, pentru a completa automat o propoziție incompletă. Pentru a face acest lucru, puteți folosi sursa `predict_words.c`, care

primește o propoziție incompletă (bazată pe input-ul din testul pe care rulați acest program, pentru a avea o acuratețe mai bună) și numărul de cuvinte pe care le va prezice (maxim 3).

Exemplu de rulare:

```
./predict <cale_fișier_out_generat> <cale_fișier_word_frequencies>  
"propoziție[spațiu]" <număr_cuvinte>
```

```
./predict output/Task3/output0.txt word_freq/words_freq_0 "Biofuel is used to  
" 2
```

Acest task este o introducere foarte simplistă în utilizarea n-gramelor în **NLP**.

Folosirea oricăror structuri de date în afara șirurilor de caractere și vectori va duce la depunctarea totală a cerinței.

După ce ai rezolvat și ultima provocare vocea stranie te felicită și îți spune că acum ești pregatit de Tema 2 la PCLP. La auzul acestor cuvinte te trezești. "A fost doar un vis", îți spui și vezi că tocmai s-a postat Tema 2.

Trimitere temă

Tema va fi trimisă folosind [vmchecker](#), cursul **Programarea Calculatoarelor (CB & CD)**.

Formatul arhivei va fi următorul:

1. {tema2.c Task1.c Task2.c Task3.c utils.h Makefile}
2. Un fișier **README** în care vă descrieți rezolvarea taskurilor.