Tema 1
Client-Server App

Cerințe generale:

1. Sistem de tipul Client - Server **(mai mulți clienți și un server)**.

2. Metoda de procesare a datelor pe Server **trebuie să fie făcută concurent - folosind go routines.**

3. Există un fișier de configurare în care există parametrii inițiali ai programului.
   Ex: câte elemente are array-ul de date pe care îl poate trimite clientul, de câte ori se apelează o routină go. Decideți voi ce anume puteți include în acest fișier.

4. Trebuie să existe mesaje între client și server de tipul:
   - Client <Nume> Conectat.
   - Client <Nume> a facut request cu datele: <date>.
   - Server a primit requestul.
   - Server proceseaza datele.
   - Server trimite <raspuns> catre client.
   - Client <Nume> a primit raspunsul: <raspuns>.

Cerințe individuale:

1. Clientul trimite către server un **array de strings de aceeasi dimensiune**.
   Serverul returnează către client un array cu cuvinte unde cuvantul i din lista output este alcatuit din caracterele de pe pozitia i in fiecare din cuvintele din array-ul input.
   Exemplu: **casa, masa, trei, tanc, 4321 => cmtt4, aara3, ssen2, aaic1**
   Pentru pozitia 0 avem cmtt4 pentru ca sunt alese in ordine caracterele de pozitia 0 din fiecare string, deci c din casa, m din masa etc.

2. Clientul trimite către server un **array de strings**. Un string poate conține atât caractere, cât și cifre, amestecate.
   Serverul returnează către client numărul de numere care sunt pătrate perfecte.
   Exemplu: **abd4g5, 1sdf6fd, fd2fdsf5 => 2 pătrate perfecte: 16 din 1sdf6fd, 25 dinfd2fdsf5.**

3. Clientul trimite către server un **array de numere întregi**.
   Serverul răspunde către client cu suma numerelor array-ului format prin inversarea fiecărui element din array-ul inițial.
   Exemplu: **12, 13, 14 => 21, 31, 41 cu suma 93**

4. Clientul trimite către server un **array** având pe primele două poziții limitele întregi **a, b** ale unui interval și un **șir de valori întregi**.
   Serverul returnează media aritmetica a numerelor citite, pentru care suma cifrelor aparține intervalului [a,b].
   Exemplu: **Pentru a = 2, b = 10, n = 5 și valorile 11, 39, 32, 80, 84 server-ul va returna 41.**

5. Clientul trimite către server un **array de strings.**.
   Serverul elimina toate stringurile ce nu reprezinta numere binare, iar pentru cele ce reprezinta numere binare va face conversia in baza 10 si va returna clientului.
   Exemplu: **2dasdas, 12, dasdas, 1010, 101 => 10, 3**

6. Clientul trimite către server un **array de strings**. Un string poate conține doar caractere.
   Serverul returnează către client codificarea Caesar (https://en.wikipedia.org/wiki/Caesar_cipher) a fiecărui element, folosind **LEFT** sau **RIGHT** shift cu **k** poziții.
   Exemplu: **abcdef => xyzabc folosind LEFT shift cu 3 caractere.**
   abcdef => bcdefg folosind RIGHT shift cu 1 caracter.

7. Clientul trimite către server un text codificat după regula următoare: în fața fiecărui caracter este scris un număr ce reprezintă numărul de apariții consecutive al acestuia.
   Server-ul returnează textul decodificat. Numărul ce apare în fața unui caracter va fi mai mic sau cel mult egal cu 20.
   Exemplu: **Pentru “1G11o1L” se va afișa “GooooooooooL”.**

8. Clientul trimite catre server un **array de numere naturale**.
   Server-ul returnează numărul total de cifre al tuturor numerelor prime din șir.
   Exemplu: Pentru: **23, 17, 15, 3, 18 => 5 cifre (nr 23, 17, 3)**

9. Clientul trimite către server un **array de strings**. Un string poate conține atât caractere, cât și cifre, amestecate.
   Serverul răspunde către client cu numărul de cuvinte care au un număr par de vocale aflate pe poziții pare în cuvânt.
   Exemplu: **mama, iris, bunica, ala => 2 cuvinte: iris, ala.**

10. Clientul trimite către server un **array de strings.** Un string poate conține atât caractere, cât și cifre, amestecate. Serverul returneaza cmmdc pentru toate numere (se calculeaza listele divizorilor pentru fiecare numar si se calculeaza intersectia listelor). Exemplu: **24, 16, 8, aaa, bbb => 8.**

11. Clientul trimite către server un **array de numere întregi**.
    Serverul răspunde către client cu suma numerelor array-ului format prin permutarea la dreapta a **k** cifre ale fiecărui element din array-ul inițial.

Exemplu: **1234, 3456, 4567 => pentru k = 2: 3412 + 5634 + 6745 = 15791.**

12. Clientul trimite către server un **array de numere naturale**.
    Serverul returnează către client suma elementelor array-ului format din dublarea primei cifre a fiecărui număr.
    Exemplu: **23, 43, 26, 74 => suma de returnat = 223 + 443 + 226 + 774.**

13. Clientul trimite către server un array de întregi ce are pe primele doua poziții limitele întregi **a, b** ale unui interval, restul array-ului reprezentând componentele a n numere complexe (elementele din array luate două câte două reprezintă partea reala și cea imaginara a unui numar complex).
    Server-ul returnează în ordine crescătoare valorile modulelor ce nu aparțin intervalului [a, b]. - a se folosi **pachetul cmplx** (https://golang.org/pkg/math/cmplx/#Abs).
    Exemplu: **3,10,3,4,5,6 reprezintă: intervalul [3,10] și numerele complexe: 3+4i, 5+6i.**

14. Clientul trimite către server un **array de strings**.
    Serverul returnează către un array de stringuri cu toate stringurile ar putea sa fie parole valide (contin litere mici, mari, cifre si simboluri). Puteti folosi regex.
    Exemplu: **Ceva12!@, asa212, dasdas => Ceva12!@**

15. Clientul trimite către server un **array de caractere(pot sa fie strings de dimensiune 1) ce pot sa fie litere mici, mari, cifre sau simboluri** .
    Serverul returnează către client un numar N de parole, N fiind aleator ales intre 1 si 10. Fiecare parola este de dimensiune M, unde M este intre 5 si 15 si este alcatuita din caracterele primite de la client, avand minim o litera mica, o litera mare, un simbol si o cifra.
    Exemplu: **a, b, e, 3, !, A => bA3ba, ee3!Bab3, aabbA!**
    Serverul intoarce 3 parole (3 ales aleator din intervalul 1-10) de dimensiune 5, 8, respectiv 6 (alese aleator din intervalul 5-15).

Barem: 1.5p

1. Aplicație cu un singur client - server 0.2p
2. Mai mulți clienți: 0.5p
3. Citire din fişier: 0.2p
4. Mesaje client-server: 0.2p
5. Arhitectura generală a aplicației: 0.4p
6. Termen de prezentare: 12.12.2024