2024. december 19.

C++ parciális (II)

A feladatod egy egyszerű jelszókezelő rendszer megvalósítása, amelyben a felhasználók különböző weboldalakhoz és alkalmazásokhoz tartozó jelszavaikat tárolhatják és kezelhetik. A rendszer lehetőséget biztosít jelszavak hozzáadására, lekérdezésére, frissítésére és törlésére. A jelszavakat memóriában tároljuk, és a weboldal vagy alkalmazás neve alapján lehet hozzájuk férni.

A jelszavakat egy fájlból olvasd (**passwords.in**). Ez egy **speciális fájl**, beolvasáskor figyelj a struktúrájára:

```
Gmail;MyS3cur3P@ss
Facebook;StrongPass123
GitHub;Passw0rd!
```

1. Hozz létre egy interfészt (**IPasswordManager**), majd egy új osztályt (pl. **PasswordManager**), amely a jelszókezelő rendszer funkcionalitásait valósítja meg. Ez az osztály rendelkezzen minden olyan adattaggal és függvénnyel, amely szükséges a jelszókezeléshez. A feladat megoldásakor használj optimális adatstruktúrát.

2. A konkrét osztályban valósítsd meg a következő alapvető CRUD műveleteket:

1. **Hozzáadás vagy frissítés:** Legyen képes hozzáadni egy új jelszót egy adott weboldalhoz vagy alkalmazáshoz. Ha a weboldal már létezik, frissítse a hozzá tartozó jelszót. A jelszónak “helyesnek” kell lennie (használd a validátor függvényedet, részletek lentebb).
2. **Lekérdezés:** Legyen képes visszaadni egy adott weboldalhoz vagy alkalmazáshoz tartozó jelszót, ha az létezik. Ha a weboldal nincs a rendszerben, dobjon `std::invalid_argument` kivételt.
3. **Törlés:** Legyen képes törölni egy adott weboldalhoz vagy alkalmazáshoz tartozó jelszót. Ha a weboldal nem létezik, dobjon `std::invalid_argument` kivételt.
4. **Listázás:** Legyen képes kilistázni az összes tárolt weboldalt és a hozzájuk tartozó jelszavakat. A jelszavak megjelenítésénél kezdetben csak csillagok (pl. `*****`) látszódjanak (**fontos**: annyi csillagot írjuk ki, amennyi karakterből áll a jelszó). Adj lehetőséget a felhasználónak a jelszavak felfedésére:

   ```
   void listPasswords(bool reveal = false) const;
   ```

3. Hozz létre egy segédfüggvényt, amely ellenőrzi, hogy egy jelszó megfelel-e az alapvető biztonsági követelményeknek (validátor függvény). A jelszavak legalább 8 karakter hosszúak kell legyenek, tartalmazniuk kell legalább egy nagybetűt, kisbetűt és számot. A

függvény térítsen vissza megfelelő boolean értéket.

4. Hozz létre egy függvényt, amely generál egy véletlenszerű, erős jelszót a felhasználó számára (azaz egy “helyes” jelszót). Használd a C++ véletlenszám-generátorát:

```
const std::string chars =
      "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*()";
std::random_device rd;
std::mt19937 rng(rd());
std::uniform_int_distribution<> dist(0, chars.size() - 1);

char randomChar = chars[dist(rng)];
```

- ——————————— -

A **main.cpp** fájl tartalma legyen a következő, használd tesztelésre:

```
#include <iostream>
#include "PasswordManager.h"

int main()
{
   try
   {
       PasswordManager pm("passwords.in");
       std::cout << "Jelszavak betöltve.\n";
       // Műveletek tesztelése
       std::cout << "\n--- Jelszavak listázása (rejtve) ---\n";
       pm.listPasswords(false);
       std::cout << "\n--- Új jelszó hozzáadása ---\n";
       pm.addOrUpdatePassword("LinkedIn", "S3cur3P@ssw0rd!");
       pm.listPasswords(true);
       std::cout << "\n--- Új jelszó hozzáadása (hibás jelszó estén) ---\n";
       try
       {
           pm.addOrUpdatePassword("WrongPass", "easypassword");
       }
       catch (const std::exception &e)
       {
           std::cerr << "Hiba: " << e.what() << "\n";
       }
       std::cout << "\n--- Jelszó lekérdezése ---\n";
       std::cout << "LinkedIn jelszó: " << pm.getPassword("LinkedIn") << "\n";
       std::cout << "\n--- Jelszó törlése ---\n";
       pm.deletePassword("LinkedIn");
       pm.listPasswords(true);
       std::cout << "\n--- Véletlenszerű jelszógenerálás ---\n";
       std::cout << "Generált jelszó: " << pm.generatePassword() << "\n";
   }
   catch (const std::exception &e)
   {
       std::cerr << "Hiba: " << e.what() << "\n";
   }

   return 0;
}
```