ZH1_Szalai_péntek

Határidő Nincs megadva határidő **Pont** 10 **Kérdések** 4
Elérhető okt 28, 08:30 - okt 28, 10:08 körülbelül 2 óra **Időkorlát** 95 perc

Instrukciók

A kvíz úgy épül fel, hogy az első feladat egy 0 pontos szöveg, amely tartalmazza az összes alfeladatot. Ezután a teljes feladatra létre van hozva egy esszékérdés. Ennek a megoldásmezőjébe be kell másolni az összes (!) python szkript szövegét az alábbi formában (python szkript elnevezés csak példa):

```
# client.py
import socket
...

# server.py
import socket
...
```

Az utolsó két kérdés 0 pontos fájlfeltöltős kérdések, de ettől függetlenül kötelezők! Úgy kell az időt beosztani, hogy ez is beleférjen. **A futtatásokról kötelező olyan pillanatképeket készíteni, ahol egyértelműen látszik, hogy a program a specifikációnak megfelelően működik.** Ezeket a képeket egy előre megnyitott dokumentumba (pl. Microsoft Word, LibreOffice Writer vagy Google Docs) érdemes folyamatosan beillesztgetni és a végén PDF-ként kimenteni/kiexportálni, amit fel kell tölteni. A python szkripteket tartalmazó zip fájlt is fel kell tölteni a biztonság kedvéért (de önmagában ez nem elegendő)! Ehhez érdemes az alfeladatokat megoldó szkripteket külön-külön almappába tenni és az összeset együtt tömöríteni össze ZIP formátumba.

Nem szabad másolni más valaki megoldását, nem szabad külön csatornán a megoldásokat megbeszélni stb. Önállóan kell a feladatokat megoldani! A feladatokat és a megoldásokat nem szabad közzé semmilyen formában se (email, facebook, github, fórumok stb.)! Függetlenül attól, hogy ki adta le korábban a megoldást, egyértelmű másolás esetén az összes abban résztvevőnek elégtelen lesz a ZH jegye! Ezenkívül minden más segédeszközt lehet használni (internet, dokumentáció, diasorok, jegyzetek stb.).

Készítsen hitelesítő rendszert!

Fontos:

- Minden lépés, üzenet legyen kiírva a kimenetre is, hogy látszódjon a működés!
- Minden megoldott alfeladatról készüljön pillanatkép, amely a végén PDF-ként legyen feltöltve!
- Ha valamilyen információ nincs megadva, akkor az Önre van bízva!

Feladatok

1. alfeladat:

- Készítsen egy szerver-kliens alkalmazást, amely **TCP protokollt** használ!
- A szerver a ***Szolgáltató***, a kliens a ***Felhasználó***, aki különböző információkat kérhet le a ***Szolgáltatótól***, de nincs mindenhez joga, tehát egy *dictionary*-ben tárolva van, hogy egy információhoz - annak azonosítójával (*infoID*) megadva - egy felhasználó hozzátud-e férni vagy sem:

```
{
"1" : ["Alice", "Joe", "Jack", "Sue"],
"2" : ["Alice", "Jack"],
"3" : ["Joe", "Sue"]
}
```

- Továbbá az *infoID*-khoz a tartalom is tárolva van egy másik *dictionary*-ben:

```
{
"1" : "Nagyon izgalmas könyv tartalma",
"2" : "Ajándék",
"3" : "Népszámlálási adatok"
}
```

- A ***Felhasználó*** szkript indításakor a parancssori argumentumban kell megadni a nevét.
- A ***Felhasználó*** a standard inputon keresztül háromszor egymás után adhat meg *infoID*-kat, de úgy, hogy egy kérés utána várja meg a választ a Szolgáltatótól!
- A ***Felhasználó*** elküldi a lekérdezésnél a nevét és azt, hogy mit szeretne egy *struct*-ban : (*név* [5 karakter], *infoID* [char]).
- A ***Szolgáltató*** válasza szintén egy *struct*: (*hozzáférés engedélyezése* [bool], *tartalom* [50 karakter])
- Ha nincs joga hozzáférni a ***Felhasználónak*** a *tartalomhoz*, akkor (nyilván) a *hozzáférés engedélyezése* **False** lesz és a *tartalom* pedig "Hozzáférés megtagadva" üzenet lesz, a ***Felhasználó*** szkript ebben az esetben tegyen a tartalom elé "HIBA! " prefixet.

2. alfeladat:

- A ***Szolgáltató*** a jogokat tartalmazó *JSON* fájlt (*privileges.json*) olvas be, a forrását parancssori argumentumként kell megadni az indításkor. A fájl tartalma ugyanaz, mint a fenti *dictionary*-nek. (A másik *dictionary*-t nem kell fájllal kiváltani.)

3. alfeladat:

- A ***Szolgáltató*** legyen képes több ***Felhasználót*** is kiszolgálni a 'select' függvény segítségével.

4. alfeladat:

- Van egy ***Azonosító*** is, akihez el kell menni a ***Felhasználónak*** az indulásakor egy **tokenért UDP protokollt** használva. (Ez még azelőtt

történjen meg, mielőtt az első *infoID*-t bekérné.)
- A ***Felhasználó*** tehát elküldi a nevét, amire az ***Azonosító*** generál egy véletlen számot *10000* és *20000* között és lekéri az aktuális időt (time.time()), amelyeket egy *dictionary*-nak a *kulcs-érték* párjaként tárolja le, ahol a felhasználó *neve* lesz a *kulcs*, az *érték* pedig (*ellenőrző kód*, *kiállítás ideje*) tuple lesz.
- Ez utóbbit elküldi *struct*-tal a Felhasználónak: (*ellenőrző kód* [int], *kiállítás ideje* [float])
- Amikor a ***Felhasználó*** a ***Szolgáltatónak*** küldi az üzenetét, akkor az előző alapján módosul a *struct*: (*név* [5 karakter], *infoID* [char], *ellenőrző kód* [int], *kiállítás ideje* [float])
- A ***Szolgáltató*** először ellenőrzi a token érvényességét az ***Azonosítónál***, azaz **UDP protokollt** használva lekérdezi tőle a **tokent** az adott felhasználó nevével, aki egy *struct* üzenettel ugyanabban a formában válaszol, mint a ***Felhasználónak*** (azaz (*ellenőrző kód* [int], *kiállítás ideje* [float]), de itt a *kiállítás idejének* nem lesz szerepe).
- Az ***Azonosító*** tehát ha olyan *nevet* kap, ami már szerepel a *dictionary*-jében *kulcsként*, akkor nem generál újat, hanem csak elküldi a hozzátartozó *értéket*.
- Az ***Azonosítótól*** kapott *ellenőrző kódhoz* véletlenszerűen hozzáad *0*-t vagy *1*-t.
- Továbbá megnézi a ***Szolgáltató***, hogy ez a (néha hibás) *ellenőrző kód* megegyezik-e a ***Felhasználótól*** kapott *ellenőrző kóddal*, valamint azt, hogy a *kiállítás ideje* óta *5* mp.-nél több idő telt-e már el. Az előző esetben "Nem érvényes azonosítás", az utóbbi esetben "Lejárt az időbélyeg" *tartalommal* válaszol a ***Felhasználónak*** és - értelemszerűen - a *hozzáférés engedélyezése* **False** lesz, a ***Felhasználó*** tegyen az *tartalom* elé "HIBA! " prefixet, egyébként pedig a működés ugyanaz, mint korábban volt.

Pontozás:

- Helyesen működik a ***Felhasználó*** és a ***Szolgáltató***. (4 pont)
- Az ***Szolgáltató*** 'select' függvénnyel lett megoldva és képes több ***Felhasználót*** kiszolgálni. (1 pont)
- A fájlkezelés megfelelő. (1 pont)
- Helyesen működik az ***Azonosító***, illetve megfelelőek a szükséges módosítások. (4 pont)
(Azaz összesen 10 pontot lehet szerezni. Az eredmény 5 pont alatt elégtelen (1).)

Megjegyzések:

- Lehet egyből a 3. alfeladatot implementálni (nem szükséges külön az 1. alfeladatot). Ha az tökéletesen működik 'select' függvénnyel, akkor az 1. alfeladat pontjai is járnak érte.
- Lehet a 3. alfeladat nélkül implementálni az 1., 2. és 4. alfeladatokat, de akkor tökéletes működés esetén is csak max. 9 pontot lehet elérni.