Rendszerközeli programozás dokumentáció

Borsos Attila Máté - EFMG5L

Project feladat:

C nyelvű program amely egy 1bit színmélységű bmp fájlt hoz létre, amely véletlenszerűen változó mennyiség időbeli változását szemléltető grafikont ábrázol.
Továbbá két eltérő üzemmódban indított példány egyetlen program, de kettő folyamat(küldő/fogadó)

Rendszer követelmények:

- Stabilitás szempontjából naprakész GCC és Linux operációs rendszer.
- Processzor: Minimum és ajánlott CPU sebesség, magok száma.
- Memória (RAM): Minimum és ajánlott mennyiség, pl. 2 GB vagy 4 GB.

Használati útmutató:

A programot -> chart <- néven kell indítani, különben nem fog működni.

```
gcc Main.c MainHelp.c -fopenmp -o chart
```

A program egy .bmp kiterjesztésű képet fog generálni.
Továbbá a felhasználó a következő parancssori argumentumokat tudja használni:

- --help
- --version
- -send
- -receive
- -file
- -socket

--version parancsot használja a felhasználó azzal letudja kérdezni, a fejlesztőjét, a program verzióját, és mikor készült.
--help paranccsal további segítséget kérhet.
Továbbá a felhasználó választhat, mely kommunikációs módot szeretné használni, -file vagy -socket. Elérhető üzemmód a -send vagy a -recieve, azaz küldő vagy fogadó.
Kettő terminálra lesz szükség amelyből az egyik a fogadó a másik a küldő akár hálózaton keresztül.

	File	Socket
Fogadó	./chart -file -receive	./chart -socket -receive
Küldő	./chart -file -send	./chart -socket -send

Hibakódok:

(1): A fajl nevenek a következőnek kell lennie!: chart
(11): Hiba a könyvtár megnyitásakor!
(12): Hiba a fajl megnyitasakor!
(14): Nem található fogadasi folyamat!
(16): Hiba a memória lefoglalása során!
(17): Szerver leallitva
(13): SIGUSR1 Hiba: A fajlon keresztüli küldési szolgaltatas nem elerheto!
(18): SIGALRM Hiba: A szerver nem valaszol időkereten belül.
(19): Socket létrehozása nem sikerult.
(20): Nem sikerult NumValues kuldese
(21): Fogadasi hiba
(22): NumValues eltérés: elvárt .. kapott ..
(23): Átküldési hiba
(25): Adatméret eltérés: elvárt .. kapott ..
(27): Rögzítő hiba
(29): Hiba valasz kuldeskor

Alprogramok rövid leírása:

- int Measurement(int **Values)

Ez a függvény egy mérési adatsort generál, és visszaadja ennek a méretét más algoritmusok számára.Az int Measurement(int **Values) függvény fogad egy int **Values típusú pointer paramétert. Ez a pointer egy olyan pointerre mutat, amelynek a típusa int. Tehát ez a függvény megváltoztatja az értéket, amelyre a Values mutat.

- void BMPcreator(int *Values, int NumValues)

Ez a függvény egy BMP (Bitmap) fájlt hoz létre egy adatsor alapján. Fájlméretet (fsize) számolunk ki az adatsor méretéből. Kiírjuk a BMP fejlécét és a bitképet a fájlba.

- int FindPID()

Ez a függvény egy másik futó alprogram PID-jét keresi a "/proc" könyvtárban található folyamatok listájából. Ha megtalálja a keresett alprogramot, visszaadja annak PID-jét, különben -1-gyel tér vissza.

- void SendViaFile(int *Values, int NumValues)

Ez a függvény egy adatsort kiír egy "Measurement.txt" nevű fájlba, majd megkeresi egy másik futó alprogram PID-jét, és SIGUSR1 jelzést küld neki. Ha a fájl megnyitása vagy a másik futó alprogram PID-jének keresése nem sikerül, a függvény hibaüzenetet ír ki és kilép a programból.

- void ReceiveViaFile(int sig)

Ez a függvény egy SIGUSR1 jelzésre reagálva kiolvassa az "Measurement.txt" nevű fájlt,

amelyben az adatsor található. Az adatokat dinamikusan tárolja, majd meghívja a BMPcreator függvényt az adatsorral.

- void SignalHandler(int sig)

Ez a függvény kezeli a különböző jeleket (SIGINT, SIGUSR1 és SIGALRM). SIGINT esetén leállítja a szerver futását, SIGUSR1 esetén hibaüzenetet jelenít meg

- void SendViaSocket(int *Values, int NumValues)

Ez a függvény adatokat küld egy szervernek UDP protokoll segítségével. Először beállít egy socketet és beállítja a kívánt paramétereket. Ezután küldi az adatok méretét, majd figyeli a szerver válaszát. Ha a válasz megfelelő, küldi az adatokat.

- void ReceiveViaSocket()

Ez a függvény UDP protokoll segítségével fogadja az adatokat egy szerver oldaláról. Először beállítja a socketet és a szükséges paramétereket, majd köti a szerver címét a sockethez. Ezt követően folyamatosan fogadja az adatokat, válaszol a kliensnek az adatok méretével, majd létrehozza a bitképet az adatok alapján.