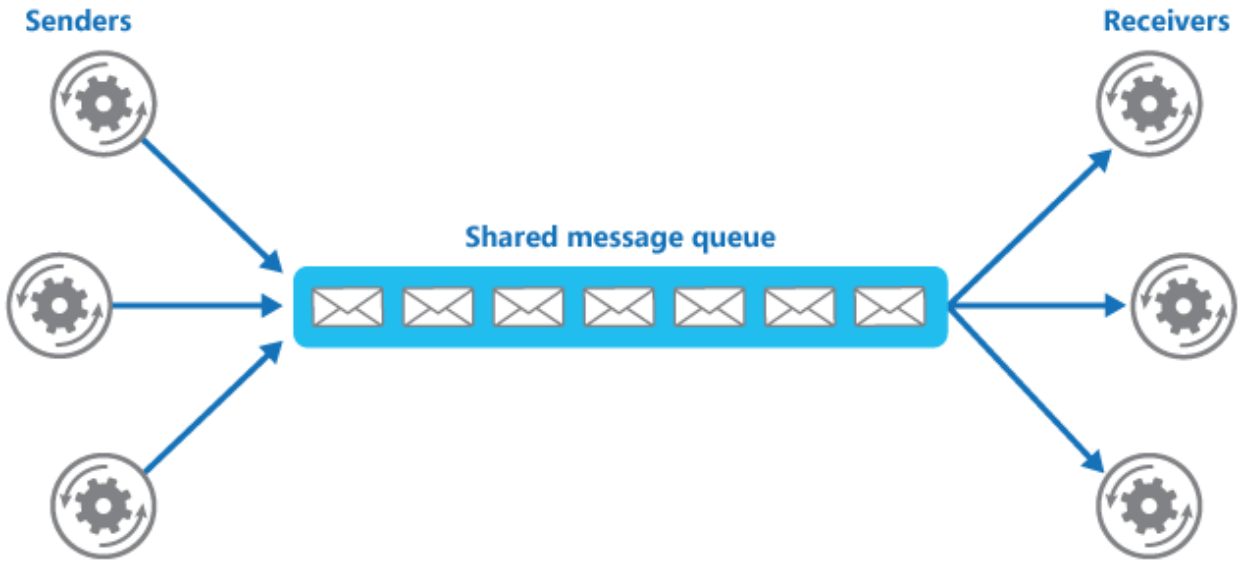
7. Gyakorlat

Processzek közötti kommunikáció: üzenetsorok



Létrehozott üzenetsorok listája: ipcs -q

Megadott azonosítójú üzenetsor törlése: ipcrm -q msqid

Üzenetsor létrehozása:

Üzenetsorokat létrehozhatunk az **msgget** segítségével, melynek paraméterként egy egyedi kulcsot, illetve a hozzáférési jogosultságokat kell átadni.
A visszatérési érték normál esetben egy üzenetsor azonosító, melyet felhasználva tudunk üzeneteket tenni az üzenetsorba, illetve kivenni abból, stb., hiba esetén pedig -1.
Pl.: *int msqid = msgget(19950914, 00600 | IPC_CREAT);*
Részletekért, lásd: **man msgget**

Üzenet küldése:

Az üzenetsorba üzeneteket az **msgsnd** segítségével küldhetünk.
A függvény paraméterei: az üzenetsor azonosítója, az üzenetre mutató pointer, az üzenet mérete, illetve egy flag.
Az üzenet formája általánosan a következő:

```
struct msgbuf {
    long mtype;
    /* tetszőleges adatok */
};
```

A flag a következő értékeket veheti fel:

- *0*: normál működés, ha nincs hely az üzenetsorban, blokkolódik, addig amíg bele nem tudja tenni az üzenetet,
- *IPC_NOWAIT*: normál működéssel ellentétben, ha nincs hely az üzenetsorban, várakozás helyett hibát jelez.

A visszatérési érték hiba esetén -1, egyébként 0.
Pl.: *int ret = msgsnd(msqid, &messageToBeSent, sizeof(struct msgbuf), IPC_NOWAIT);*
Részletekért, lásd: **man msgsnd**

Üzenet fogadása:

Az üzenetsorból az üzeneteket az **msgrcv** segítségével olvashatjuk ki.
A függvény paraméterei: az üzenetsor azonosítója, az üzenetet tároló struktúrára mutató pointer, az üzenet mérete, milyen típusú üzenetet akarunk fogadni (*mtype*), flag.
A flag a következő értékeket veheti fel:

- *0*: normál működés, ha nincs a megadott típusú üzenet a sorban, akkor várakozik,
- *IPC_NOWAIT*: ha nincs a megadott típusú üzenet a sorban, akkor hibát jelez,
- *MSG_EXCEPT*: a megadott típusún kívül bármilyen üzenetet fogad,
- *MSG_NOERROR*: ha az üzenet hosszabb a megadott méretnél, akkor levágja a végét,
- Illetve ezek kombinációit.

A visszatérési érték hiba esetén -1, egyébként a kiolvasott üzenet mérete.
Pl.: *int ret = msgrcv(msqid, &receivedMessage, sizeof(struct msgbuf), 0, MSG_NOERROR);*
Részletekért, lásd: **man msgrcv**

Üzenetsor kezelése:

Az üzenetsorok kezelése az **msgctl** rendszerhívás használható, melynek segítségével információkat kérdezhetünk le az üzenetsorról, módosíthatjuk a paramétereit, valamint törölhetjük azt.
A függvény paraméterei: az üzenetsor azonosítója, a végrehajtandó művelet, és egy *struct msqid_ds* típusú struktúrára mutató pointer, melynek felhasználása a megadott művelettől függ.
A megadható műveletek a következők:

- *IPC_STAT*: az üzenetsor adatait a megadott struktúrába másolja (ebből kiolvasható például, hogy hány üzenet van a sorban, mikor módosult utoljára, ki írt bele utoljára, stb.),
- *IPC_SET*: a megadott struktúrán keresztül módosíthatók az üzenetsor adatai (például hozzáférési jogosultságok),
- *IPC_RMID*: üzenetsor megszüntetése.

A visszatérési érték hiba esetén -1, egyébként 0.
Pl.: *int ret = msgctl(msqid, IPC_RMID, NULL);*
Részletekért, lásd: **man msgctl**

Feladat 1:

A már jól ismert tippelgetős programunkat írjuk át olyan módon, hogy a gyerek a tippjeit üzenetsoron keresztül küldje a szülőnek.

Feladat 2:

Eddigi ismereteinket felhasználva írjunk programot, amelyben a szülő processz addig küld egy üzenetsorba 0 és 1 közötti véletlenszerűen generált valós számokat, amíg a sor meg nem telik. Ezt követően 5 gyerek processz párhuzamosan elkezdi kiolvasni a számokat a sorból, kiírják a kiolvasott számok összegét a képernyőre és elküldik az üzenetsorba, az általuk feldolgozott számok darabszámával együtt. Miután a gyerekek elvégezték a dolgukat, a szülő kiolvassa az eredményüket az üzenetsorból, és meghatározza, hogy melyik gyerek dolgozta fel a legtöbb számot.

Feladat 3:

Eddigi ismereteinket felhasználva írjunk chat programot, amelyben a beszélgetésben részt vevő feleket az általuk indított processz PID-je azonosítja. A szülő processzt használjuk üzenetküldésre, minden üzenet előtt megkérdezi, hogy kinek szánjuk az üzenetet (a címzett szülőprocesszének PID-je), majd ennek megfelelően küldi a sorba a megadott szöveget. A gyerek processzt arra használjuk, hogy kiolvassa a szülőnek címzett üzeneteket a sorból és kiírja őket a képernyőre.