# 13. Tétel

Ismertesse az I/O multiplexelés lehetőségeit a Linux alkalmazásokban

## I/O multiplexelés

Kliens/szerver program párhuzamos állomány olvasása és írása. (pl. web böngésző hálózati kapcsolaton keresztül több oldal komponensét töltse le, hogy gyorsítsa a hozzáférést)

### Legegyszerűbb megoldás:

* read() műveletek egy ciklusban – böngésző minden kapcsolaton beolvassa az adatokat, majd továbblép a következőre
* ha minden csatornán egyszerre érkezik adat 🡪☺
* ha nem 🡪 ☹ 🡪 nem blokkolt I/O kezelés
  + utasítani a read()-ot, hogy ne tartsa fenn a sort 🡪ha nem tudod olvasni, 0-t ad vissza (O\_NONBLOCK opció)
  + a program folyamatosan olvasgatja mindkét leírót 🡪feleslegesen terheli a rendszert

## Select

Párhuzamos beolvasása az állományoknak.

Lehetővé teszi, hogy a processz blokkolódjon, és több állományra várakozzon

int select(int n, fd\_set \*readfs, fd\_set \*writefds, fd\_set \*exceptfds, struct timeval \*timeout);

int pselect(int n, fd\_set \*readfds, fd\_set \*writefds, fd\_set \*exceptfds, const struct timespec \*timeout, const sigset\_t \*sigmask);

* fd\_set struktúra kezelése (makrókkal) (file descriptor struct)
  + FD\_ZERO(fd\_set \*set) – törli az állományleíró listát (init)
  + FD\_SET(int fd, fd\_set \*set) – fd leíró hozzáadása a listához
  + FD\_CLR(int fd, fd\_set \*set) – fd leíró kitörlése a listából
  + FD\_IDSET(int fd, fd\_set \*set) – True, ha az fd benne van a listában
* readfs: (readable file descriptors) azok a file descriptorok, amik akkor oldják fel a select() várakozását, ha olvasható állapotba kerülnek
* writefds: irásra kész állapotban oldják fel a select()-et
* exceptfds: azoknak a file descriptora, amelyeknek valamilyen különleges állapotára várunk
* n : a legnagyobb file descriptor a listából + 1
* timeout: select() max. várakozási ideje, ami után mindenképpen visszatér
  + select() esetén ennek egy módosított értékével tér vissza, amivel jelzi, hogy mennyi idő telt el a várakozással (nem minden rendszer esetén azonos)
  + pselect() – semmit nem változtat a timeout paraméterén
* sigmask: aktuális signal maskot ezzel módosítja a rendszerhívás idejére (NULL érték kikapcsolja ezt a funkciót)
  + signal mask – a signalok halmaza, amelyek kézbesítése jelenleg blokkolt a hívó számára

## Poll

Slect()-hez hasonló működésű

Linux rendszerekben a select() a poll()-al van implementálva

int poll(struct pollfd \*fds, nfds\_t nfds, int timeout)

* pollfd struct:
  + int fd – file descriptor
  + short events – milyen eseményekre várunk
  + short revents – a kapott eseményeket rögzíti a poll() ide (a feldolgozás során ezt kell vizsgálni)
* nfds – a file descriptorok száma
* timeout – max. várakozási időtartam

A poll() addig várakozik, amíg legalább egy megadott esemény be nem következik, vagy le nem jár a timeout.