# 12. Tétel

Ismertesse a Linux rendszerben használatos inode fogalmát, olvasásának és állításának rendszerhívásait és függvényeit!

## Inode

Leíró adatstruktúra, amely az adott állomány paramétereit tartalmazza (jogok, méret)

### Típusai:

* on-disk inode: lemezen tárolt leíró
  + jogok, méret
  + rá hivatkozó fájlnevek száma. Amikor egy állományt, ami erre hivatkozik törlünk, csökken a számláló. Amint ez 0 lesz + egyetlen processz sem tartja nyitva 🡪 a fájl véglegeen törlődik
* in-core inode: a processz megnyitásakor a memóriába másolódik az on-disk inode + számon tartja a processzek számát. Ha ez módosul, a processz bezárásakor felülírja az on-disk inodeot

**C/C++ használata:** #include<unistd.h>

## Inode infók kiolvasása

jogok olvasásához nem kell különleges jogosultság, csak az adott könyvtár keresési joga

* stat(const char \*file\_name, struct stat \*buf)
  + a file\_name paraméter által megadott állomány inode infóit adja vissza
* lstat(const char \*file\_name, struct stat \*buf)
  + stat a link lekövetése nélkül
* fstat(int fd, struct stat \*buf)
  + megnyitott állományok inode infóinak elérése

### Struct\_stat

* dev\_t stdev – álományt tartalmazó eszköz azonosítója
* ino\_t st\_ino – állomány on-disk inode száma
* mode\_t st\_mode – állomány jogai és típusa
* nlink\_t st\_nlink – referenciák száma erre az inodera
* uid\_t st\_uid – állomány ownerjének uidje
* dig\_t st\_gid – állomány ownerjének gidje
* dev\_t st\_rdev – ha az állomány speciális eszközleíró, akkor ez a mező tartalmazza a major és minor azonosítót
* off\_t st\_size – állomány mérete bájtokban
* unsigned long st\_blksize – fájlrendszer blokkmérete
* unsigned long st\_blocks – állomány által allokált blokkok száma
* time\_t st\_atime – legutolsó hozzáférés időpontja
* time\_t st\_mtime – legutolsó módosítás időpontja
* time\_t st\_ctime – legutolsó változtatás időpontja (az állományon, vagy az inodeon)

## Jogok lekérdezése/állítása

### Lekérdezés:

* hiába tartalmazza az st\_mode az álomány jogait, ebből az infó kinyerése nem olyan egyszerű
* megoldás: int access(const char \*pathname, int mode)
* mode
  + F\_OK – létezik/elérhető?
  + R\_OK – a processz olvashatja az állományt?
  + W\_OK – a processz írhatja az állományt?
  + X\_OK – a processz futtathatja az állományt (kereshet benne könyvtár esetén? )
* visszatérési érték: 0, ha siker

### Jogok állítása:

* int chmod(const char \*path, mode\_t mode): a fájl elérési útvonalával és nevével
* int fchmod(int fd, mode\_t mode): a file descriptorral (egyedi azonsítója a fájlnak)
* mode: hozzáférést szabályozó bitek kombinációja (néhány kapott saját konstanst is )
  + S\_ISUID – 04000 – setuid
  + S\_ISGID – 02000 – setgid
  + S\_ISVTX – 01000 – set sticky bit
  + S\_IRUSR – 00400 – csak a tulaj olvashatja
  + …

## Tulajdonos és csoport beállítása

* fajtái
  + int chown(const char \*path, uid\_t owner, gid\_t group)
  + int lchown(const char \*path, uid\_t owner, gid\_t group)
  + int fchown(int fd, uid\_t owner, gid\_t group)
* csak root jogosultsággal lehetséges