2022. tavasz “Operációs rendszerek B” vizsga kérdések

A BSc képzés tesztjei linken nagyon hasonló kérdések megtalálhatóak, csak ott kicsit más az anyag.
Köszönet annak a valakinek aki csinált egy Google Form tesztet is.
Frissítve: 2022-06-07.

Igaz-Hamis (174 db):

A biztonsági mentés visszaállításához előbb telepítenünk kell az operációs rendszert
Hamis

A blokkgyorsítótár a fizikai memóriát használja a diszkműveletek gyorsítására
Igaz

A cserehely (SWAP) a taszkok teljes memóriaképét tárolja
Hamis

A cserehely a fizikai memória egy elszeparált része, ahol a nem futó taszkok adatait tároljuk
Hamis

A cserehelyre futó taszk adatai nem kerülhetnek, hiszen a CPU nem tudná elérni
Hamis

A folyamatokon belül csak egy verem lehet
Hamis

A fork() Unix rendszerhívás betölt és elindít egy új programot
Hamis

A FreeBSD egy Linux disztribúció
Hamis

A futási szint (runlevel) meghatározza a UNIX rendszerek aktív szolgáltatásait
Igaz

A futási szint (runlevel) meghatározza a Unix rendszerekben futó taszkok prioritását
Hamis

A fájlrendszerek telepítése mindig egy fa struktúrával reprezentálható (A gyökér könyvtárból kiindulva a fájlok és a könyvtárak egy fát alkotnak)
Hamis

A hálozati kommunikáció (socket) egy asszimetrikus kommunikációs forma
Igaz

A hálozati kommunikáció csak fizikai kapcsolat meglétével valósítható meg

Hamis

A hálózati kommunikáció csak fizikai hálózati kapcsolat megléte esetén alkalmazható. (???)
Hamis

A karakteres felületen müködő parancsértelmezők jellemzően programozható eszközök
Igaz

A keret- és a laptáblák száma megegyezik
Hamis

A kernel adatstruktúrák egy része a fájlrendszeri interfészen keresztül hozzáférhető Linux alatt
Igaz

A kernel az első program amit a hátértárról betöltve a processzor futtatni kezd
Hamis

A kernel moduláris felépítése csökkenti a kernel futásidejű memóriafoglalását
Igaz

A kiéheztetés a statikus prioritást alkalmazó ütemezőkben NEM kerülhető el
Igaz

A konvoj-hatás például a legrövidebb hátralevő löketidejű (SRTF) algoritmussal megszüntethető
Igaz

A kooperatív ütemezők nem veszik el a futó taszkoktól a processzort
Igaz

A körforgó (RR) ütemezés esetén a konvoj-hatás nem jelentkezik, mivel az ütemező a taszkokat csak adott ideig futtatja, utána átütemezéssel megszakítja a futásukat
Igaz

A körforgó (RR) ütemezés képes elkerülni a kiéheztetést
Igaz

A körforgó (RR) ütemező használata optimális átlagos várakozási időt eredményez
Hamis

A körforgó (RR) ütemező kooperatív és elkerüli a kiéheztetést
Hamis

A körforgó (RR) ütemező optimális várakozási időt eredményez
Hamis

A körülfordulási idő egyenlő a várakozási és a futási idők összegével
Igaz

A lapcsere során egy lap behozásakor jellemzően nem kell keretet felszabadítani, mivel szabad keretek létezéséről egy speciális folyamat gondoskodik

Igaz

A laphiba azt jelenti, hogy az adott lap nem létezik sehol, ezért nem lehet rá hivatkozni

Hamis

A laphiba kezelése jellemzően az alkalmazás hibás viselkedés

Hamis

A laphiba kezelése sok esetben taszkok közötti átütemezéssel is jár

Igaz

A laphiba kezelése általában taszkok közötti kontextusváltással is jár

Igaz

A laphibák rendszerszintű gyakorisága a CPU kihasználtsággal (egy bizonyos mértékig) lineárisan nő

Igaz

A laphibák száma nagyon sok párhuzamosan működő taszk esetén lineárisan függ a multiprogramozás fokától

Hamis

A laplopó (page daemon) valamilyen lapcsere-algoritmust futtat

Igaz

A laplopó (page daemon) valamilyen lapozási (lapbehozási) stratégiát alkalmaz

Hamis

A lapok tárba fagyasztásának alapvető célja a firssen behozott lapokhoz tartozó keretek felszabadításának megakadályozása

Igaz

A leggyorsabb 1 TiB méretű adattároló rendszer a merevlemezes meghajtó (HDD)

Hamis

A legrégebben várakozó (FCFS) ütemezési algoritmus FIFO adatstruktúrát használ

Igaz

A legrégebben várakozó (FCFS) ütemezőnél akkor is jelentkezhet a konvoj-hatás, ha csak I/O intenzív taszkok vannak a rendszerben

Hamis

A legrégebben várakozó ütemező (FCFS) ütemező preemptív

Hamis

A legrövidebb hátralevő löketidejű (SRTF) ütemező preemptív

Igaz

A legrövidebb hátralévő löketidejű (SRTF) ütemező olyan prioritásos ütemezőnek is tekinthetőm ahol a prioritás egyenlő a taszk hátralévő löketidejével, és a legkisebb értékkel rendelkező taszk fog futni

Igaz

A legrövidebb löketidejűt előre (SJF) ütemező preemptív

Hamis

A Linux felhasználói módban többszintű ütemezőt használ

Igaz

A Linux kernel átütemezési pontjainak alkalmazása javítja a válaszidőt, mivel így a kernel módban futó taszkok bármikor megszakíthatók

Hamis

A mai operációs rendszerekbe jellemzően többféle ütemező müködik egyszerre

Igaz

A mai operációs rendszerekben egyáltalán nem alkalmaznak előretekintő lapozást, mivel sosincs arról információ, hogy a jövőben milyen lapokra lesz szükség

Hamis

A mai Windows operációs rendszerekben a FAT32 a legelterjedtebb fájlrendszer

Hamis

A mai Windows változatok az NT kernelre épülnek

Igaz

A mai általános célú operációs felhasználású rendszer kernelének forráskódja több százezer programsor

Hamis

A memória-intenzív taszkok nagy memóriafoglalás esetén CPU-intenzívvé válnak

Hamis

A memóriaalapú permanens tárolók (SSD-k) élettartalma jelemzően 5 év körüli

Hamis

A memóriakezelés során fellépő védelmi hiba hardver megszakítást okoz

Igaz

A Microsoft RDP egy elterjedt kijelzőszerver protokoll

Igaz

A modern mikrokernelek (pl. L4) nagyon lassú üzenetalapú kommunikációval működnek (???)

Igaz

A multiprogramozott operációs rendszer abban különbözik más rendszerektől, hogy többféle programozási nyelvet támogat

Hamis

A OS kernelek minden része (eljárása) védett módban működik
Hamis

A POSIX ACL egy szerepalapú hozzáférés-szabályozási rendszer
Igaz

A PRAM (pipelined RAM) modell nem engedi meg a közös memória konkurens írását két (vagy több)taszk által, ezért ilyen esetekben is garantálja a programok helyes működését
Hamis

A PRAM modell írás-olvasás ütközésnél mindig először az írás műveletet hajtja végre, hogy az olvasás már az új értékkel térhessen vissza
Hamis

A pí szám kiszámítása sok számjegyre egy I/O intenzív feladat
Hamis

A RAID 10 megbízhatósága jobb de teljesítménye gyengébb mint a RAID 1-nél
Hamis

A RAID 5 egy kijelölt paritástárolót (partíciót, diszket) használ a redundáns tárolás céljára
Hamis

A RAID0 (stripe) két diszk esetén két példányban tárol minden adatot
Hamis

A RAID0 általában gyorsabb a RAID1-nél. de a RAID1 megbízhatóbb
Igaz

A rendszerhívások jellemzően megszakításokkal járnak együtt
Igaz

A rendszerkönyvtárak az operációs rendszer védett módban működő részei
Hamis

A rendszerkönyvtárak olyan eljárásokat tartalmaznak, amelyek sokféle feladatban előfordulnak, így nem kell minden programban külön-külön megvalósítanunk azokat, hanem támaszkodhatunk egy közös implementációra
Igaz

A rendszerprogramok védett módban futnak
Hamis

A statikus többszintű ütemezőkbe nem jelentkezhet a kiéheztetés jelensége hiszen a globális ütemező preemptív
Hamis

A statikus többszintű ütemezőkben nem jelentkezhet konvoj-hatás, hiszen a globális ütemező preemptív
Hamis

A szál a taszk egy olyan megvalósítása, amely önálló memóriaterülettel rendelkezik
Hamis

A szál egy szekvenciális működésű taszk, amely egy folyamaton belül más szálakkal közös halmot (heap-et) használ
Igaz

A szálaknak saját verme van
Igaz

A számítógépeken futó taszkok többsége I/O intenzív (???)
Hamis

A taszkok adminisztratív adatait védelmi okokból mindig a kernel címterében tároljuk
Hamis

A taszkok elindulásukkor lefoglalják a számukra szükséges memória tartományt
Hamis

A taszkok löketidejét a gyakorlatban működő ütemezők előre ismerik
Hamis

A taszkok már elinudláskor lefoglalják a számukra szükséges teljes memóriatartományt
Hamis

A távoli eljáráshívás (RPC) megvalósítása csak egy folyamaton belüli szálak között lehetséges, mivel azok férnek hozzá egymás memóriaterületeihez és eljárásaihoz
Hamis

A távoli eljáráshívás (RPC) számítógépek között is működik, nem csak egy gépen belül
Igaz

A többszintű visszacsatolt sorok (MFQ) ütemező az I/O intenzív taszkokat magasabb prioritásiszinten tartja, mint a CPU-intenzíveket
Igaz

A többszintű visszacsatolt sorok (MFQ) ütemező felfelé lépteti a taszkokat a szintek között ha várakozó állapotból visszatérnek épp futásra készbe
Igaz

A többszintű visszacsatolt sorok (MFQ) ütemező lefelé lépteti a taszkokat a szintek között, ha azok az adott szinten kihasználják a rendelkezésükre álló CPU-időt (pl. a RR időszeletet)
Igaz

A többszintű visszacsatolt sorok (MFQ) ütemező preemptív
Igaz

A többszintű visszacsatolt sorok(MFQ) ütemező bemeneti szintválasztó algoritmusa a taszkok prioritása alapján határozza meg a kezdeti szintjüket

Hamis

A Unix cron egy középtávú ütemező

Hamis

A UNIX exec() rendszerhívás betölt és elindít egy új folyamatot

Hamis

A Unix flock() egy ajánlott zárolási forma

Igaz

A Unix jelzések a kommunikáció leggyorsabb formái közé tartoznak nagyon alacsony késleltetésük miatt

Hamis

A Unix operációs rendszer első változata az AT&T Bell Lab kommerciális termékeként jelent meg, amelyet számos cég és egyetem vásárolt meg

Hamis

A Unix programokban mindenféle jelzésre szabadon beállíthatunk jelzéskezelő eljárásokat

Hamis

A valósidejű működés alapvető célja az, hogy a felhasználók valós időben végezhessék a rendszeren a feladataikat

Hamis

A vergődés az a jelenség amikor a keretek számának növelése a laphibák számát is növeli

Igaz

A virtuális és fizikai címek alapvetően hardveres úton, de időnként a kernel segítségével történik

Igaz

A virtuális és fizikai memóriacímek futásidejű transzformációja alapvetően szoftveres úton történik

Hamis

A válaszidő azt fejezi ki hogy a felhasználónak mennyit kéne várnia egy program első válaszára (kimenetére) annak elindításától számítva

Hamis

A válaszidő mindig kisebb, mint a körülfordulási idő

Igaz

A várakozási idő a taszk futásra kész állapotban eltöltött ideje

Hamis

A Windows egyes beágyazott (kevés erőforrással rendelkező) rendszerben is működik

Igaz

A Windows nem rendelkezik programozható, karakteres parancsértelmezővel
Hamis

A Windows operációs rendszerekben a FAT32 a legelterjedtebb fájlrendszer
Hamis

A Windows Task Scheduler egy hosszútávú ütemező
Igaz

A Winlogon előbb fut, mint az SMSS (munkamenet-kezelő) a Windows-on
Hamis

Ajánlott zárolás (advisory locking) esetén az OS nem biztosít eszközöket a zárolás megvalósítására
Hamis

Alapvetően védelmi mechanizmusok miatt szükséges a cserehely (swap) alkalmazása mivel az ott található adatokat a CPU direkt módon nem képes elérni
Hamis

Az ablakkezelő (Windows Manager) a karakteres parancsértelmező (Shell) grafikus változata
Hamis

Az alkalmazások által használt összes memória mérete sosem haladhatja meg a fizikai memória méretét
Hamis

Az Apache webszerver szálalapú változata nagyobb teljesítményre (kérés / mp) képes, mint a folyamatalapú
Igaz

Az első szintű (RAM, MBR) betöltő már ismeri a fájlrendszerek felépítését, hiszen onnan tölti be a második szintű betöltőt
Hamis

Az elérhető virtuális címtartomány nagyobb a felhasználható fizikai címtartománynál
Igaz

Az időosztásos operációs rendszer alkalmazása csökkenti a rendszerek válaszidejét a klasszikus multiprogramozott rendszerekéhez képest
Igaz

Az időosztásos operációs rendszerek egyben multiprogramozott rendszerek is
Igaz

Az időosztásos operációs rendszerek jellemzően Round-Robin ütemezőt futattnak
Igaz

Az IEEE POSIX egy szabvány, ami előírja a kernel belső felépítését
Hamis

Az iSCSI (internet SCSI) fájlokon elvégzett műveletek hálózati átviteli protokollja
Hamis

Az operációs rendszer kernele elszeparálja (védi) egymástól a taszkokat, ezért azok védett módba futhatnak tovább
Hamis

Az operációs rendszer kernele felügyeli a felhasználói módban futó taszkok működését
Igaz

Az operációs rendszerben nem lehet több müködő taszk mint ahány végrehajtó egység van
Hamis

Az OS kernelek minden része (eljárása) védett módban történik
Hamis

Az osztott memória egy PRAM (pipelined RAM) modell szerint müködő eszköz
Igaz

Az SJF ütemező képes kezelni a konvoj-hatást, de nem a kiéheztetést
Igaz

Az SRTF ütemező olyan prioritásos ütemezőnek is tekinthető, ahol a prioritás egyenlő a taszk hátralévő löketidejével, és a legkisebb értékkel rendelkező taszk fog futni
Igaz

Az X11 egy széles körben használt kijelzőszerver protokoll, amely Windows operációs rendszeren is elérhető
Igaz

Az óra és az újabb esély lapcsere ugyanazon múltbéli adatokra támaszkodva működik (???)
Igaz

Az újabb esély (second chance) lapcsere a felszabadításra kiválasztott keretet csak akkor nem szabadítja fel ha legutobbi ellenőrzés óta módosította valamelyik taszk
Hamis

Az újabb esély lapcsere algoritmus a lapok módosítási idejét ellenőrzi
Hamis

Az ütemezés során leghatékonyabban láncolt listák segítségével tarthatjuk nyilván taszkok (pl. prioritásszerint) rendezett halmazát
Hamis

Az ütemező a várakozó állapotú taszkok közül választja ki a következő futó taszkot
Hamis

Az ütemező átbocsájtó képessége a taszkok által időegység alatt átvitt adatok mennyisége
Hamis

Az ütemező átbocsájtó képessége az egységnyi időszelet alatt átütemezett taszkok száma
Hamis

Az ütemezők nem szálakat, hanem folyamatokat ütemeznek
Hamis

Az üzenetsor egy indirekt kommunikációs megoldás
Igaz

Az üzenetváltásos kommunikáció általában lassabb mint a PRAM modell szerinti
Igaz

Az „óra” lapcsere algoritmus valójában az „újabb esély” algoritmus hatékonyabb megvalósítása
Igaz

Egy feladatot mindig egy taszk old meg
Hamis

Egy folyamaton belüli szálak a PRAM (pilenied RAM) modell szerint tudnak egymással kommunikálni
Igaz

Egy hibás memóriacímzési művelet laphibát eredményez
Hamis

Egy mai általános célú operációs rendszer kernelének forráskódja több százezer programsor
Hamis

Egy operációs rendszer forráskódja lehet néhány tízezer programsor, de akár sok millió is
Igaz

Egy operációs rendszer nem lehet egyszerre monolitikus és moduláris felépítésű
Hamis

Egy operációs rendszerben nem lehet több működő taszk, mint ahány végrehajtó egység van
Hamis

Egy párhuzamos végrehajtást (több konkurens taszk együttműködését) igénylő feladat egyetlen folyamaton belül is megvalósítható
Igaz

Egy rendszer absztrakt virtuális gépei összességükben több erőforrást tartalmaznak mint amennyi fizikailag rendelkezésre áll
Igaz

Egy statikus többszintű ütemező kimeneti szintválasztó algoritmusa lehet körfogó (round robin) ütemező, ha azt az adott alkalmazási környezet megkívánja

Igaz

Egy taszk futásra kész állapotból várakozó állapotba is átkerülhet

Hamis

Egy taszk futó állapotból futásra kész állapotba kooperatív ütemező esetén is átkerülhet

Hamis

Egy taszk létrehozása utáni első állapota a "fut"

Hamis

Egy taszk létrehozása utáni első állapota lehet a "fut"

Igaz

Egy taszk várakozási ideje mindig kisebb mint a körülfordulási ideje

Hamis

Egy taszk várakozó állapotból futó állapotba is átkerülhet

Hamis

Egy többszintű ütemező akkor is lehet preemptív, ha minden szinten kooperatív ütemezési algoritmust használ

Igaz

Egy általános célú operációs rendszerben jellemzőek 1-2 kontextusváltás történik másodpercenként

Hamis

Előfordulhat, hogy különböző taszkokhoz tartozó, eltérő virtuális memóriacímek ugyanarra a fizikai címre képződnek le

Igaz

Ha a rendszerben csak I/O intenzív feladatok vannak akkor a legrégebben várakozó (FCFS) ütemező esetén nem léphet fel konvoj-hatás

Igaz

Ha egy felhasználói program kernel módba vált (pl. rendszerhívással), a CPU utasításkészlete akkor is korlátozott marad hogy ne okozzon gondot a kernelben

Hamis

Ha egy memória-intenzív taszkokat futtató rendszerben alacsony a CPU-kihasználtság, akkor nincs elegendő memória a taszkok számára

Igaz

Ha letöröljük egy szimbolikus link által hivatkozott fájlrendszeri bejegyzést, akkor az adat elvész. Ha magát a szimbolikus linket törlöm le akkor nem

Igaz

Ha növeljük egy rendszerben a fizikai memória méretét, akkor mindig csökkenni fog a laphibák száma, hiszen egyszerre több lapot tarthatunk bent a memóriában

Hamis

Kliens (PC, telefon, tablet) gépeken több UNIX alapú rendszer fut mint Windows alapú

Igaz

Kliens platformon a Windows részesedése 90% feletti

Hamis

Kooperatív ütemező esetén a taszkok nem hajtanak végre fut -> futásra kész állapotátmenetet

Hamis

Két, egy folyamaton belüli szál azonos virtuális címen jellemzően ugyanazt látja, de van a virtuális címtartományuknak olyan része, amely biztosan különböző adatokat tartalmaz (???)

Igaz

Körforgó (RR) ütemezés során egy taszk csak akkor lép ki a futó állapotából ha lejárt az időszelete

Hamis

Különböző folyamatokban található szálak is kommunikálhatnak egymással PRAM modell szerint az OS szolgáltatásait felhasználva

Igaz

Lehetséges várakozásmentes I/O műveletek alkalmazása a programjainkban

Igaz

Megbízható (kellően redundáns) rendszer esetén nincs szükség mentésre hiszen nem következhet be adatvesztés

Hamis

Minden rendszerhívás védett módban hajtódik végre

Hamis

Szerverplatformon a Linux részesedése 10% alatti

Hamis

Szinkron üzenetváltásos kommunikáció során a küldés művelet befejezése megelőzi fogadás műveletelindítását

Hamis

Van olyan operációs rendszer, amely részben hangutasításokkal is kezelhető

Igaz

Feleletválasztós (31 db):

- A konvoj hatás kooperatív ütemező algoritmussal nem kezelhető: **egyes esetekben igen másokban nem**
- A **CPU-löket** a taszk processzoron végrehajtott utasításainak sorozata.
- Mire szolgál a System V üzenetsorok üzeneteiben a típusjelzés? **Az üzenetek szűrése**
- Mondjon egy szabványos (!) eszközt (teljes megnevezés), amely a PRAM modell szerinti kommunikációt teszi lehetővé! **POSIX Shared Memory (?)**
- Egy fájl megnyitásakor a kernel egy **???** ad vissza az alkalmazás számára, amely a további műveletek során a felhasználható a megnyitott fájl azonosítására.
- Mekkora az 5 darab 1TiB méretű diszkből összeállított RAID6 tárolórendszer teljes hasznos kapacitása? **3 TiB**
- A kiéheztetés jelensége jelentkezhet **SJF** ütemező esetén és ott **sehogyan sem** kezelhető.
- A taszk egy **végrehajtás** alatt álló program.
- A taszkok állapotváltozásai alapvetően **megszakítások** miatt következnek be.
- A több feladatot egymás után megoldó **kötegelt (batch)** rendszerek után a feladatokat párhuzamosan futtató **multiprogramozott** OS-típusok jelentek meg.
- A FAT32 maximális fájlmérete **4 GiB**.
- A **I/O-löket** a taszk működésének azon fázisa, amikor I/O művelet végrehajtására vár.
- A folyamatnak saját **memóriatartománya**, a szálnak pedig saját **verme** van.
- A(z) **átbocsájtó-képesség** meghatározza az időegység alatt elvégzett feladatok számát.
- A konvoj-hatás nem jelentkezhet **legrövidebb hátralevő löketidejű (SRTF)** és **körforgó (RR)** ütemező esetén.
- A(z) **öregítés** a futásra kész taszk prioritásának növelése az ebben az állapotában eltöltött idejével arányosan.
- A(z) **kiéheztetés** az a jelenség, amikor prioritásos ütemezés esetén egy taszkot folyamatosan megelőznek nála magasabb prioritásúak, ezért nem jut processzorhoz.
- A(z) **FCFS** ütemező saját döntése alapján nem helyez át futásra kész állapotba taszkokat. (Ne egy konkrét ütemezőt nevezzen meg, hanem egy kategóriát!)

- A Bélády-féle anomália azt jelenti, hogy a fizikai memóriakeretek számának **növelése** időnként a laphibák gyakoriságának **növekedését** vonja maga után.

- A UNIX mmap(), illetve a Windows CreateFileMapping() rendszerhívások alapvető célja, hogy a fájlok tartalma fájlműveletek helyett **memória** műveletekkel legyen elérhető.

- A Unix flock() egy **ajánlott** zárolási forma.

- Mekkora az 5 darab 1TiB méretű diszkből összeállított RAID5 tárolórendszer teljes hasznos tárolókapacitása? **4 TiB**

- A logikai kötetkezelés (LVM) egyik alapvető célja, hogy növelje a maximális **tárolórendszer-méretet**.

- A legrövidebb hátralevő löketidejű ütemező (SRTF) végrehajthat "**Fut → Futásra kész"** állapotátmenetet egy taszkon, míg például a legrégebben várakozó (FCFS) nem. "**Fut → Várakozik"** állapotátmenetet egyetlen ütemező sem hajt végre taszkokon.

- Állítsa párba az alábbi feladattípusokat és a rájuk leginkább jellemző feladatjelleget!
  Önvezető autó irányítása: **valósidejű**
  Műveletek nagy adatbázisokon: **memóriaintenzív**
  Kisvállalati fájlszerver: **I/O intenzív**
  Számítógépes játékok: **cpu intenzív**

- Rendezze sorrendbsra kész taszk prioritásának növelése az ebben az állapotában eltöltött idejével arányosan.e a feladatkezeléssel kapcsolatban megfogalmazott alábbi időjellemzőket hosszuk szerint!
  A legrövidebb: **válaszidő**
  Rövidebb: **várakozási idő**
  Leghosszabb: **körülfordulási idő**

- Állítsa párba az alábbi OS-típusokat és jellemzőiket!
  Leglényegesebb tulajdonsága, hogy több feladatot futtat egyszerre: **Multiprogramozott**
  Törekszik azonos CPU-időt biztosítani a feladatok végrehajtása számára. **Időosztásos**
  Több feladatot adhatunk a rendszernek, de egyszerre csak egyet fog guttatni. **Kötegelt**

- Milyen ütemezőket rendelne egy többszintű ütemezőben az alábbi feladattípusokhoz a lenti készletből?
  Valósidejű: **???**
  I/O intenzív: **???**
  CPU-intenzív: **???**

- Hol tároljuk az alábbi adatokat a virtuális memóriakezelésben?
  Taszkok memóriatartományainak leirói: **laptábla**
  A cserehely adatblokkjainak leírói: **diszk blokk (?)**
  A fizikai memóriatartományok leírói: **???**

- A virtuális tárkezelés során milyen hiba generálódik, amikor…
  a taszk olyan címet használ, amely számára nem engedélyezett: **védelmi hiba**
  a taszk egy igény szerint kitöltendő lapot először használ: **fill-on-demand hiba (?)**
  a taszk olyan lapra hivatkozik, amelyik cserehelyen van: **laphiba**

- Melyik memóriakezelési tevékenység tud jobban jósolni (ha lehet ilyen különbséget tenni köztük)?
  Kevesebb információt tud a lapokról, ezért kevésbé jósol jól: **Lapozási stratégia**
  Több információ áll a lapokról a rendelkezésére, ezért jobban jósol: **Lapcsere algoritmus**

Programkódok:

- Forkolás:

```
if ((res = fork()) == 0) {   // elágazás a visszatérési értékkel
                             // gyerek ága
  exec(...);                 // betöltünk egy másik programkódot
                             // ha visszatér, exec hiba történt
} else if ( res < 0 ) {      // szülő ága, hibaellenőrzés
                             // ide jön a fork() hiba kezelése
}
// res = CHILD_PID (>0), szülő kódja fut tovább
```

- Hálózati szerver:

```
sfd1 = socket()
bind(sfd1)
listen(sfd1)
while
  sfd2 = accept(sfd1)
    fork()
  szülő:
    vissza a ciklusba
  gyerek:
    recv(sfd2)
    send(sfd2)
    close(sfd2)
    exit()
```