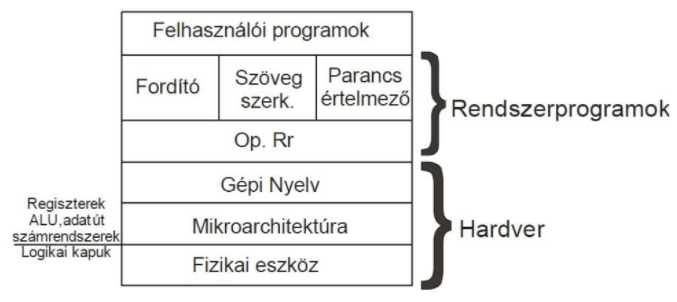
**Operációs rendszerek - licensz vizsga 2021**

**1. Operációs rendszerek absztrakciós szintjei**



* Alul van a **hardver**, amely sok esetben maga is két vagy több szintből (vagy rétegből) áll.
  + A legalsó réteg a **fizikai eszköz** szint, tartalmazza az integrált áramkör lapkákból épülő fizikai eszközöket, huzalozást, áramellátást, katódcsöveket és hasonló fizikai eszközöket.
  + A következő a **mikroarchitektúra szint**, ahol a fizikai eszközöket működési egységekké csoportosítják. Ez a szint tipikusan tartalmaz belső CPU regisztereket és aritmetikai-logikai egységet magába foglaló adatútvonalat. Az adatútvonal célja utasítások egy halmazának a végrehajtása.
  + A **gépi nyelv** (assembly) általában 50 és 300 közötti utasítást tartalmaz, többségük a gépen belüli adatmozgatásokra, aritmetikai és összehasonlító műveletekre szolgál. Ezen a szinten a bemeneti-kimeneti eszközöket vezérlik oly módon, hogy speciális eszköz regisztereket értékekkel töltenek fel.
* A második nagy szint a **rendszerprogramok**, amely szintén több rétegre oszlik fel
  + Az **operációs rendszer** egyik fő feladata az összes bonyolultság elrejtése és a programozó számára egy kényelmesebb utasításkészlet biztosítása.
  + Az operációs rendszer felett van a **rendszerszoftver** maradék része. Itt találjuk a parancsértelmezőt (**shell**), az ablakkezelő rendszert, fordítókat, szövegszerkesztőket és a hasonló, alkalmazásoktól független programokat.
* A fordítók, szövegszerkesztők felhasználói módban futnak. Ha a felhasználónak nem tetszik egy adott fordító, megírhatja sajátját.

**2. A rendszermag (kernel) és a héj szerepe valamint fontosabb feladataik.**

A **kernel** az operációs rendszer védett módban működő programja, amely felügyeli a felhasználói módú programok működését, és biztosítja hozzáférésüket a rendszer erőforrásaihoz.

**Feladatai:**

* Eszközkezelők:
  + ki és bemeneti eszközök kezelése
  + Egységes kezelői felület különböző perifériáknak
* Megszakításkezelés
* Rendszerhívás, válasz: felhasználói alkalmazások erőforrásigényének kiszolgálása
* Erőforrás kezelés: közös e usfeloldás
* Processzorütemezés: várakozó munkák közötti processzoridő beosztás (valamilyen stratégia alapján)
* Memóriakezelés: memóriaterület felosztása munkák között
* Állomány és lemezkezelés (pl. rendszerezés)

A **shell**, magyarul **héj** egy felhasználói felület, amely által hozzáférhetünk az operációs rendszer szolgáltatásaihoz.

**Feladatai:**

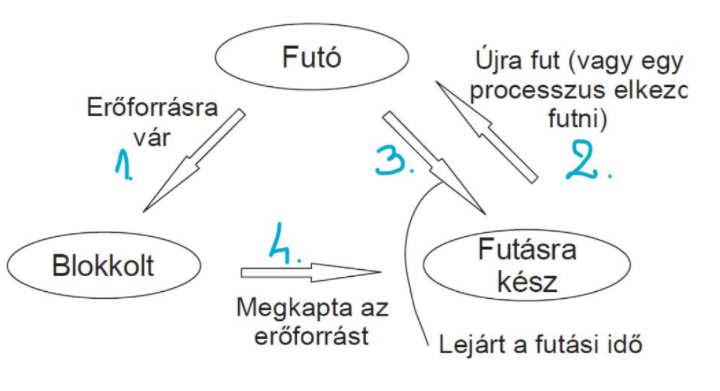
* Kommunikáció biztosítása a felhasználóval
* Alkalmazások futásának kezelése (indítás, futási feltételek biztosítása, leállítás)

**3. A processzus fogalma és a processzus állapotainak ismertetése**

A processzus egy **végrehajtás alatt lévő program**. Elméletileg minden processzusnak saját virtuális CPU-ja van, de valójában a valódi CPU kapcsolgat a processzusok között. A processzusok közötti kapcsolgatást multi programozásnak nevezzük.

A processzus lehet *futó*, *blokkolt* vagy *futáskész* állapotban**.**

* ***Futó***: az adott pillanatban éppen használja a CPU-t.
* ***Blokkolt***: bizonyos külső esemény bekövetkezéséig nem képes futni.
* ***Futáskész***: készen áll a futásra; ideiglenesen leállították, hogy egy másik processzus futhasson.



1. *Erőforrásra vár*: A processzus bemeneti adatra várva blokkol.
2. *Újra fut, vagy egy processzus elkezd futni:* Az ütemező másik processzust szemelt ki.
3. *Lejárt a futási idő:* Az ütemező ezt a processzust szemelte ki.
4. *Megkapta az erőforrást:* A bemeneti adat elérhető.

**4. Folyamatok ütemezése időosztásos és valósidejű rendszerekben**

**Ütemezés *időosztásos* rendszerekben:**

Több felhasználó/folyamat dolgozhat egyszerre, minden felhasználó/folyamat több feladatot, programot futtathat. Minden program kap egy adott időszeletet az operációs rendszer ezért felelős moduljától a processzor idejéből. Egy meghatározott rövid ideig futtatja az alkalmazást, az időszelet lejárta után egy következő program kapja meg a processzort.

**Ütemezés *valós idejű* rendszerekben:**

Ha egy vagy több külső fizikai eszköz megszakítást küld a számítógép fele, akkor reagálni kell egy adott időn belül.

* ***szigorú valósidejű rendszerek***: rendszerek, amelyek specifikációja egyértelműen rendszerhibának tekinti valamely időkövetelmény be nem tartását. Ez azt jelenti, hogy a szigorúan valós idejű rendszerek normál működése során nem engedhető meg, hogy valamilyen időkövetelmény teljesítése elmaradjon. Ha mégis bekövetkezne ilyen esemény, a rendszernek a kidolgozott stratégiának megfelelően kell reagálnia, semmiképp sem szabad határozatlan állapotba kerülnie.
* ***toleráns valósidejű rendszerek:*** rendszerek, amelyek körében az ilyen mulasztást csak a rendszer működőképességének vagy teljesítményének csökkenéseként fogjuk fel, és erre az esetre is specifikáljuk a rendszer viselkedését.

Az események, amelyre egy valósidejű rendszerben kell válaszolni: periodikus, aperiodikus

**5. A rendszerhívás fogalma, rendszerhívás végrehajtásának alapelve**

**A rendszerhívások:**

* Az operációs rendszerek a felhasználói programok elől elrejtik a számítógéphez illesztett hardver részleteket, így kényelmes programfejlesztési környezetet biztosítva a programozónak.
* Az operációs rendszer úgynevezett szolgáltatásokat nyújt a programoknak.
* Az operációs rendszer a programok “kéréseit” ezen szolgáltatások segítségével hajtja végre.

A **rendszerhívás** a programozó számára látszólag egy függvényhívás, valójában azonban összetettebb működésű, hiszen váltania kell a CPU-nak üzemmódok között. Ez a „többlet” tevékenység a programozó számára rejtett, jellemzően a rendszerkönyvtárak oldják meg a váltást szoftver megszakítás segítségével. A CPU a megszakítás hatására üzemmódot vált (felhasználóiból védett módba), és a kernel megszakításkezelőjét kezdi el futtatni. Az észleli a megszakítás okát, azonosítja és végrehajtja rendszerhívást. A rendszerhívás végén a kernel visszatér a megszakításból, és folytatódik a normál módú működés.

Pl.

* Processzuskezelő rendszerhívások
* Szignál kezelő rendszerhívások
* Fájl kezelő rendszerhívások

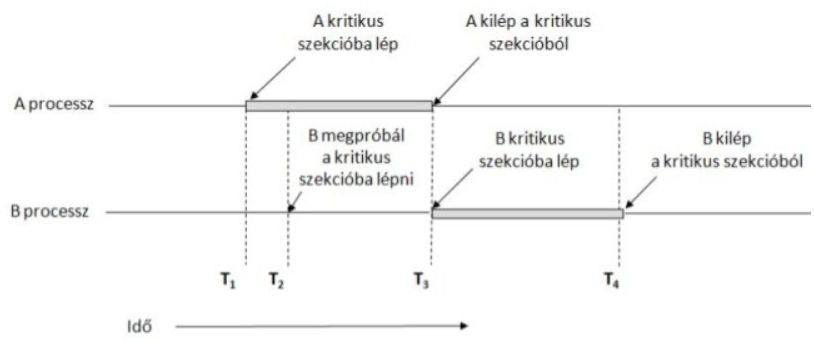
**6. Atomi műveletek. Kritikus szekció. Kölcsönös kizárás. Szemaforok.**

**Atomi műveletek:**

* Atomi az a műveletsor, amely tovább nem osztható, nem megszakítható.
* Atomi az a művelet végrehajtása nem megszakítható konkurens műveletek által.

**Kritikus szekció:**

* A programnak azt a részét, amelyben megosztott erőforrást használ, kritikus területnek, illetve kritikus szekciónak nevezzük.
* A kritikus szakaszt szokás szinkronizációs blokknak is nevezni: belépési és kilépési pontja van.
* Többféle szinkronizációs megoldás létezik a kritikus szakaszok biztonságossá tételére, amelyek célja, hogy a kritikus szakaszban csak egy szál legyen egy időben.



**Kölcsönös kizárás:**

* Mialat egy processzus a saját kritikus szekciójában a megosztott erőforrást aktualizálja, ne legyen más olyan processzus, amely saját kritikus szekciójába lép, vagy ugyanazt az erőforrást használná, (a rendszer hibás működését okozhatja).
* A kölcsönös kizárás lényege tehát, hogy más processzusokat távol tartsunk a közös erőforrásoktól, amíg azokat egy futó processzus használja.

**Szemaforok:**

* Pozitív egész számot tartalmazó közös változó és egy várakozási sor.
* A szemafor a számítógép-programozásban használt változó vagy absztrakt adattípus, amit az osztott erőforrásokhoz való hozzáférések szabályozásához használnak a többszálú környezetekben.
* Atomi művelet: Egy időben csak egy processzus operálhat egy szemaforon
* Szemaforon két atomi operáció:
  + szemafor lefoglalása Down(P):
    - folyamat a kritikus szekcióba való belépéskor meghívja a Down(P) rendszerhívást
    - ha foglalt a szemafor blokkolódik a folyamat
    - a blokkolódásból “felébredő“ processzus végrehajtja azt az operációt, amin előzőleg blokkolódott.
  + szemafor elengedése Up(P)
    - elengedi a szemafort
    - jelzést küld
* Ezeket a műveleteket egy ugyanazon időben csak egy folyamat hívhatja a szemaforon.

**7. A holtpont fogalma. A holtpont kialakulásának előfeltételei**

**Holtpont:**

* A holtpont egy rendszernek egy olyan állapota, ahonnan külső beavatkozás nélkül nem tud elmozdulni.
* Holtpont akkor fordulhat elő, amikor a folyamatok egy adott halmazában minden egyes elem leköt néhány erőforrást, és ugyanakkor várakozik is másokra.
* Több folyamat egy olyan erőforrás felszabadulására vár, amit csak egy ugyancsak várakozó folyamat tudna előidézni.
* Holtpont esetén a folyamatok körkörösen egymásra várakoznak.

**A holtpont kialakulásának előfeltételei:**

* ***Kölcsönös kizárás***: vannak olyan erőforrások a rendszerben, melyeket a folyamatok csak kizárólagosan használhatnak.
* ***Foglalva várakozás***: legyen olyan folyamat amely lefoglalva tart erőforrásokat, miközben más erőforrásokra várakozik.
* ***Nincs erőszakos erőforrás-elvétel***: a folyamatok addig birtokolják az erőforrást, míg saját jószántukból fel nem szabadítják azokat.
* ***Körkörös várakozás***: létezik a rendszerben egy olyan folyamatsorozat, melyben minden folyamat az utána következő folyamat által foglalt erőforrásra vár, a sorozat utolsó tagja pedig a sorozat első tagjára.

**8. A virtuális memória. Virtuális memória megvalósítása lapozással**

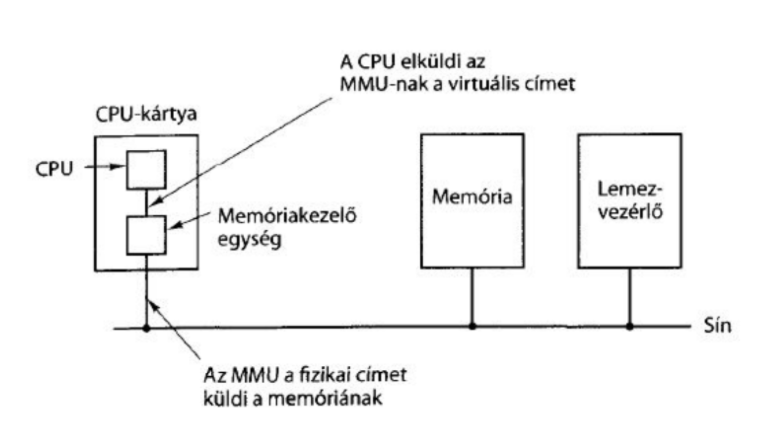
* Akkor használják, amikor nincs elég rendelkezésre álló RAM és van olyan futó alkalmazás, ami RAM-ot igényel. Megcímezzük a merevlemez bizonyos regisztereit, a merevlemezből veszünk el memóriát és ez szolgál virtuális memóriaként.
* A virtuális memória lényege, hogy a program, az adat és a verem (átmeneti tároló) együttes mérete meghaladhatja a fizikai memória mennyiségét.
* Az op. rendszer csak a program éppen használt részét tartja a memóriában, a többi a lemezen van.
* A program úgy látja, mintha rendelkezésére állna a teljes RAM készlet.

**Feladatai:**

* ***címleképezés (address translation):*** a virtuális címek fizikai címre transzformálása
* ***lapozás (paging):*** a virtuális és a fizikai memória részekre bontása, és azok megfeleltetése egymásnak
* ***cserehely (swap):*** a fizikai memória kapacitásának bővítése háttértárak segítségével

**Virtuális memória megvalósítása lapozással:**

* A virtuális memóriát használó rendszerekben leggyakrabban a lapozás technikáját használják.
* A program által generált címeket virtuális címeknek nevezzük, ezek halmazát pedig vírtuális címtartománynak nevezik.
* Ha virtuális memóriát használunk, akkor a virtuális címek nem kerülnek közvetlenül a memóriasínre, ehelyett a memóriakezelő egységbe (MMU-Memory Management Unit) kerülnek, amely a virtuális címeket képezi le a fizikai címekre.



A virtuális címteret lapnak (page) nevezett egységekre osztják, ennek megfelelő egység a fizikai memóriában a lapkeret. A lapok és a lapkeretek mindig pontosan egyforma méretűek.

* A memóriakezelő egység a bejövő virtuális címet egy lapszámra és egy ofszetre bontja.
* A lapszámot használjuk a laptábla indexeként.
* A laptáblák szerepe a virtuális lapok lapkeretekre való leképezése.

