**OPERÁCIÓS RENDSZEREK**

Az operációs rendszer fogalma, elvégzendő feladatai. Folyamatkezelés, ütemezés. Az ütemező módszerek és eljárások interaktív rendszerekben. Az ütemezéssel szemben támasztott követelmények, célok. Memóriakezelés, relokáció, memóriavédelem, swapping. Paging (lapozás).

Feladat: Választott OS-ben keresés logfile-ban. (Powershell, Bash, stb.)

**Fogalma**:  
Az operációs rendszer (OS) az az alapprogram, amely közvetlenül kezeli a számítógép erőforrásait (hardware) és biztosítja a felhasználói alkalmazások futtatásának alapját.  
Olyan program, ami közvetítőként szolgál a számítógép felhasználója és a számítógép hardware-e között, ezzel lehetővé téve a felhasználók számára, hogy hatékonyan és kényelmesen hajtsanak végre műveleteket.

**OS céljai:**

1. Felhasználói programok végrehajtása és a felhasználói problémák megoldásának megkönnyítése.
2. A számítógépes rendszer kényelmes használatának biztosítása.
3. A számítógép hardware-ének hatékony használata.

**Bővebben**:

1. Összehangolt működés biztosítása a hardware-el, input/output szinkronizálás (I/O bemeneti/kimeneti műveletek)
2. Megszakítási rendszer kezelése
3. Központi memória kezelése (Virtuális memóriakezelés)
4. Erőforrások felhasználásának ellenőrzése
5. A processzort, a periférikus erőforrásokat jól kihasználó és a különböző igényeket a lehető legjobban kielégítő működés
6. Felhasználóval folytatott kommunikáció, információcsere megvalósítása
7. Állománykezelés

**Folyamatkezelés, ütemezés:**

Az operációsrendszer két kulcsfontosságú funkciója. Biztosítják, hogy a számítógép erőforrásait hatékonyan osszák meg a folyamatok és a szálak között. Szorosan összefüggenek, de különböző feladatokat látnak el.

**Folyamatkezelés:**A folyamatkezelés magában foglalja folyamatok létrehozását, futásának kezelését, szüneteltetését és befejezését. Alapvetően programok végrehajtásának menedzselését jelenti az operációs rendszeren belül. Felelős a számítógépen futó folyamatok állapotainak és a folyamatok közötti átmenetek kezeléséért. Célja, hogy biztosítsa a hatékony végrehajtást, a rendszererőforrások megfelelő felhasználását, valamint a folyamatok közötti kommunikációt és szinkronizációt.

**Folyamat**:  
A folyamat egy végrehajtás alatt álló program, ami bizonyos erőforrásokat igényel a feladata elvégzéséhez.  
Egy program önmagában nem folyamat. Egy program egy passzít entitás, például egy végrehajtható fájl. Ezzel szemben egy folyamat egy aktív entitás, amely rendelkezik egy következő végrehajtandó utasítást kijelölő program számlálóval és egy hozzárendelt erőforráskészlettel. Az erőforrás lehet például CPU idő, memória, fájlok és I/O eszközök. Ezek az erőforrások vagy a folyamat létrehozásakor vannak kiosztva, vagy a futás során kerülnek hozzárendelésre. Rendelkezik kód, adat és állapot információval (pl.: fut, várakozik, szünetel). Egy program több folyamat is lehet (több felhasználó is használja ugyan azt a programot)

**Folyamat részei:**

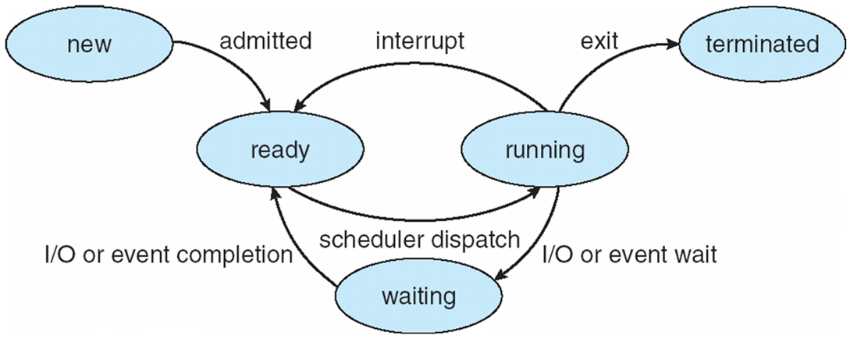
1. Programkód (**Text section**)
2. Az aktuális tevékenység, beleértve a program számlálót (**program counter**), a processzor regisztereket.
3. A vermet (**Stack**), amely ideiglenes adatokat tartalmaz, mint például függvényparaméterek, visszatérési címek és helyi változók.
4. Az adatszekciót (**Data section**), amely globális változókat tartalmaz.
5. A kupac (**Heap**), amely a futási idő alatt dinamikusan allokált memóriát tartalmaz.

**Folyamat állapot (process state):**Amikor egy folyamat végrehajtásra kerül, állapotot vált. Ez az állapot lehet:

* New: a folyamat létrejön
* Running: Utasítások végrehajtása folyamatban van
* Waitnig: A folyamat egy esemény bekövetkezésére várakozik
* Ready: A folyamat egy processzorhoz való hozzárendelésre vár
* Terminated: A folyamat végrehajtása befejeződött

**Process Control Block (PCB, Folyamat Vezérlő Blokk):**  
Az operációs rendszerben minden folyamat egy PCB által van reprezentálva. Minden PCB egy specifikus folyamatról tartalmaz információkat:

* **A folyamat állapota**: lehet running, waiting, stb.
* **Programszámláló (Program counter)**: A következő végrehajtandó utasítás helye.
* **CPU regiszterek**: A regiszterek száma és típusa eltérő lehet, függően a számítógép architektúrájától. Ide tartoznak az indexregiszterek, veremmutatók és általános célú regiszterek (valamint bármilyen condition-code információ). A programszámlálóval együtt ezt az állapotinformációt muszáj elmenteni amikor megszakítás történik a folyamat későbbi folytatása érdekében.
* **CPU ütemezési információ**: Tartalmazza a folyamat prioritását, az ütemezési sorok mutatóit (pointers to scheduling queues) és egyéb ütemezési paramétert.
* **Memory-management információ**: A folyamathoz lefoglalat memória
* **Accounting információ**: Tartalmazza a felhasznált CPU időt, a folyamat kezdete óta eltelt valós időt és időlimiteket.
* **I/O status információ**: A folyamathoz rendelt I/O eszközök listája, megnyitott fájlok listája

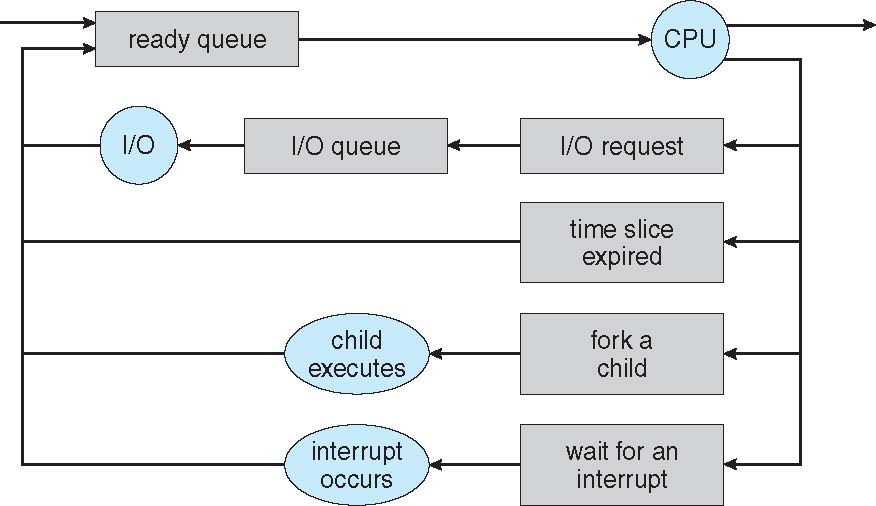
****A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

**Szálak (Threads):**  
Az eddig tárgyalt modell azt feltételezte, hogy egy folyamat egy olyan program, amely egyetlen végrehajtási szálon hajtódik végre. Ez az egyetlen vezérlési szál azt teszi lehetővé, hogy a folyamat egyszerre csak egy feladatot hajtson végre. A legtöbb modern operációs rendszer lehetővé teszi egy folyamatnak több vezérlési szál birtoklását, így egyszerre több feladat elvégzését is. Ez a funkció többmagos rendszerek esetében különösen előnyös, mivel több szál is futtatható párhuzamosan. Az ilyen rendszerek esetében a PCB-t bővíteni kell, hogy minden szál számára tartalmazzon információt.

**Folyamat ütemezés (Process Scheduling):**A multiprogramozás célja, hogy a CPU maximális kihasználtsága érdekében, mindig legyen futó folyamat. Az időmegosztás célja, hogy a folyamatok közötti váltás a lehető leggyorsabban történjen, és a felhasználó interaktívan tudjon kommunikálni minden programmal, amíg az fut. Ennek elérése érdekében a folyamat ütemező (process scheduler) kiválaszt egy rendelkezésre álló folyamatot a CPU-n való végrehajtáshoz. Egyprocesszoros rendszerek esetében sohasem lesz egynél több futó folyamat. Ha több folyamat van, a többinek várakoznia kell, amíg a CPU szabad nem lesz, és újra ütemezhetővé válik.  
A Process Scheduling a következő ütemező sorokat (scheduling queues) kezeli:

* Munkasor (Job queue): A rendszer összes folyamatát kezeli
* Készenléti sor (Ready queue): Az összes főmemóriában található folyamat halmaza, lehet készenléti, várakozó stb.
* Eszközsorok (Device queues): Az I/O eszközökre váró folyamatokat tartalmazza
* A folyamatok váltakoznak a különböző sorok között

**Ütemezés**:  
Az ütemezés egy speciális folyamatkezelési tevékenység, amely meghatározza melyik folyamatok kapják meg a CPU időt és mikor. Az ütemező dönti el, hogy mikor és melyik folyamatot helyezi át a készenléti állapotból a futó állapotba.  
A többprogramos operációs rendszerek alapja.  
A folyamatok végrehajtása a CPU végrehajtás és az I/O várakozás ciklusából áll, amelyben a folyamatok várakoznak a két állapot között (CPU–I/O Burst Cycle).  
Az ütemezési döntések akkor hozhatók meg, ha egy folyamat állapot a megváltozik.

**Ütemezők (Schedulers):**

Short-term scheduler (CPU scheduler):

* Kiválasztja a készenléti sorból, hogy melyik folyamat legyen végrehajtva következőképpen és hozzárendeli a CPU-hoz.
* A CPU ütemező döntések bekövetkezhetek amikor egy folyamat:

1. Futó állapotból várakozóra vált (pl.: I/O kérés)
2. Futó állapotból készenlétire vált (pl.: megszakítás történik)
3. Várakozó állapotról készenlétire vált
4. Befejeződik

Az 1. és 4. ütemezés nem megelőző (**nonpreemptive**).  
A többi ütemezés megelőző (**preemptive**).

* Néha ez az egyetlen ütemező a rendszerben.
* Gyakran kerül meghívásra(ezredmásodpercenként), gyorsnak kell lennie.

Long-term scheduler (Job scheduler):

* Kiválasztja, hogy melyik folyamat kerüljön a készenléti sorba.
* Ritkán kerül meghívásra (másodpercek, percek), lassú lehet.
* Szabályozza a multiprogramozás mértékét.

A folyamatok lehetnek:

* I/O-hoz kötött folyamatok: Több időt tölt I/O műveletekkel, mint számításokkal (short CPU burts).
* CPU-hoz kötött folyamatok: Több időt tölt számításokkal (few very long CPU burst).

**Ütemezéssel szemben támasztott kritériumok:**

* CPU kihasználtság
* Áteresztőképesség – egységnyi idő alatt végrehajtott folyamatok száma
* Várakozási idő – Egy folyamat készenléti sorban töltött várakozási ideje
* Válaszidő

**Ütemező algoritmusok:**

**First-Come, First-Served (FCFS):**

* A folyamatokat egy várakozási sorba helyezi és azokat érkezési sorrendben szolgálja ki.
* Nonpreemptive vagyis, ha egy folyamat végrehajtása elkezdődött azt nem szakíthatja meg más folyamat.
* Előnye a könnyű megvalósítás.
* Ha a folyamatok végrehajtási ideje eltér, a hatékonysága csökken.
* Egy hosszú, CPU intenzív folyamat blokkolhatja a többi folyamatot, amik így feltorlódnak (convoy effect).

**Shortest-Job-First (SJF):**

* Minden folyamathoz hozzárendeljük a következő CPU-burst hosszát.
* Ezeket arra használjuk, hogy a lehető legrövidebb CPU-burst-tel rendelkező folyamatot ütemezzük.
* Optimális, mivel a minimum átlagos időt biztosítja egy adott folyamatkészlet számára.
* Nehézségét, a következő CPU kérés hosszának meghatározása jelenti.

**Priority Scheduling:**

* Minden folyamathoz tartozik egy prioritási szám (integer).
* A CPU a legmagasabb prioritású folyamathoz lesz hozzárendelve (smallest integer = highest priority).
* A SJF is Priority Scheduling.
* Problémája a starvation: Az alacsony prioritású folyamatoknál előfordulhat, hogy sosem hajtódnak végre.
* Erre a megoldás az aging: Ahogy telik az idő, úgy növeljük a folyamat prioritását.

**Round Robin (RR):**

* Minden folyamat kap egy kis egységnyi CPU időt (time quantum q), ez általában 10 és 100 milliszekundum közé tehető. Miután ez az idő eltelt a folyamat a készenléti sor végére kerül.
* Ha n darab folyamat található a készenléti sorban és az időkvantum q, akkor minden folyamat a CPU idejének 1/n részét kapja meg, egyszerre legfeljebb q időegységben.
* Egyik folyamat sem fog többet várni mint (n-1)\*q.
* Az időmérő minden kvantum után megszakítást hajt végre, hogy ütemezze a következő folyamatot.

**Memóriakezelés**:

Memóriakezelés alatt az operációs rendszer azon képességét értjük, hogy a rendelkezésre álló fizikai és virtuális memória erőforrásokat hatékonyan felosztja és kezelje a folyamatok között.

**Relokáció**:

* Folyamatok vagy programok a memória egyik részéből egy másik részébe történő áthelyezési folyamata.
* Lehetővé teszi a programok dinamikus elhelyezését a memóriában.
* A programoknak nem kell fix memóriacímeken futniuk, így a memória kihasználtsága hatékonyabb és lehetséges a folyamatok párhuzamos végrehajtása.
* Biztosítja, hogy a különböző folyamatok memóriaterületei elkülönüljenek

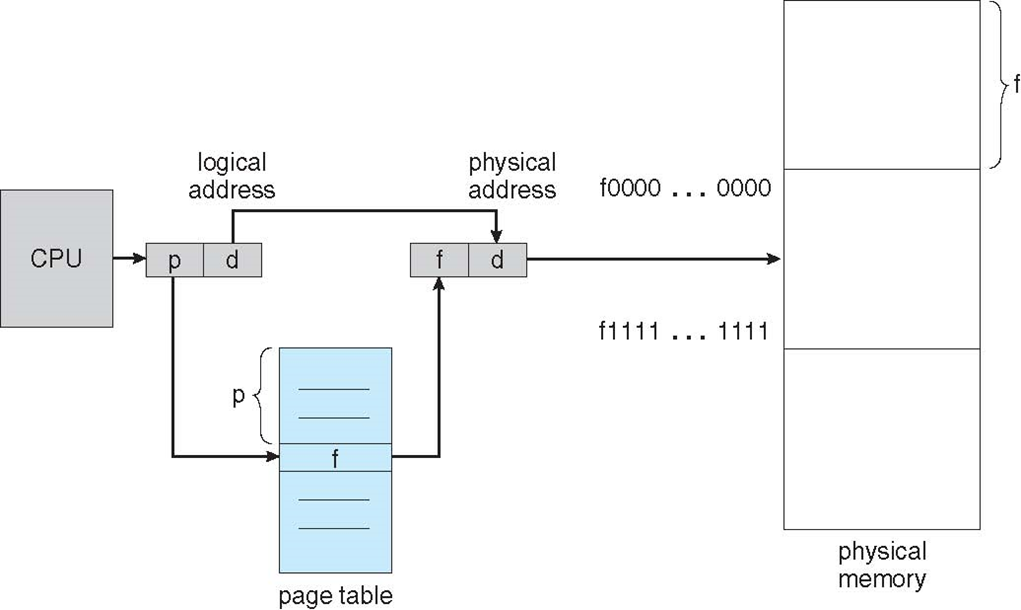
**Memóriavédelem**:

* Célja, hogy biztosítsa a különféle folyamatok által használt memóriaterületek izolációját. Ezzel védelmet nyújt a szándékos és nem szándékos hibák ellen is.
* Általában szegmens vagy lapkezelési technikákra épül, ahol minden folyamat saját címtartományt kap.
* A rendszer megakadályozza a címtér túlcsordulását vagy az illegális memóriahozzáférést.
* Minden memóriaszegmens vagy lap tartalmazhat védelmi biteket, amelyek a hozzáférést szabályozzák és megakadályozzák a jogosulatlan műveleteket.
* A felhasználói (user) és a rendszermód (kernel) között váltogatnak, ezzel is korlátozva a hozzáférést.

**Swapping**:

* Memóriakezelési technika, amely lehetővé teszi a folyamatok idegilenes áthelyezését a memóriából, egy háttértárra, ezzel fölszabadítva a memóriaterületet más folyamatok számára.
* Hasznos abban az esetben, ha a rendszerben futó folyamatok összesített memóriaigénye meghaladja a rendelkezésre álló fizikai memória mennyiségét.
* A csere során a folyamat által használt összes adatot és kódot átviszi a háttértárra (backing store).
* Amikor egy folyamatnak szüksége van a memóriára, a rendszer visszatölti a folyamat memóriaképét a fő memóriába.
* Roll Out, Roll In: Prioritáson alapul. Egy alacsonyabb prioritású folyamatot helyeznek a háttértárra, hogy a fő memória felszabaduljon egy magasabb prioritású folyamatnak.

**Paging (lapozás):**

* Memóriakezelési technika, melynek segítségével a rendszer a memóriát fix méretű blokkokra, úgynevezett oldalakra (pages) osztja fel (ezek mérete gyakran 4KB vagy nagyobb).
* Lehetővé teszi a nem összefüggő fizikai memória hatékony használatát.
* A Paging különbséget tesz virtuális (logikai) és fizikai címek között. A folyamatok logikai címekkel dolgoznak, ezeket az operációs rendszer fordítja le fizikai címekké.
* A virtuális memória oldalakra (pages), a fizikai memória pedig keretekre van felosztva (frames).
* A rendszer minden feladathoz nyilvántart egy oldaltáblát, amely a virtuális oldalaknak megfelelő fizikai keretek címét tárolja.
* ****A virtuális címek fizikai címekké való átalakítását a Memory-Management Unit végzi az oldaltábla alapján.