JEGYZŐKÖNYV

Operációs rendszerek BSc

Operációs rendszerek – 4. konzultáció - gyakorlat

Készítette: **Kocsis Katalin**
Neptunkód: **WGOWUG**

A feladat leírása:

„1. Adott egy rendszerbe az alábbi erőforrások: R (R1: 10; R2: 5; R3: 7)
A rendszerbe 5 processz van: P0, P1, P2, P3, P4
Kérdés: Kielégíthető-e P4 (3,3,0) ill. P0 (0,2,0) kérése úgy, hogy biztonságos legyen, holtpontmentesség szempontjából a rendszer - a következő *kiinduló állapot* alapján.
Igazolja a processzek végrehajtásának sorrendjét – számolással."

Az összes osztály -erőforrások száma: (10, 5, 7)						
Kiinduló állapot						
	1. lépés			2. lépés		
	MAX. IGÉNY			FOGLAL		
	R1	R2	R3	R1	R2	R3
P0	7	5	3	0	1	0
P1	3	2	2	2	0	0
P2	9	0	2	3	0	2
P3	2	2	2	2	1	1
P4	4	3	3	0	0	2

A feladat elkészítésének lépései:

1. Felírjuk a processzek által maximálisan igényelt erőforrások számait tartalmazó mátrixot. – MAX.IGÉNY
2. Felírjuk a processzek által lefoglalva tartott erőforrások számait tartalmazó mátrixot. – FOGLAL. Itt a kéréseket is bele kell foglalnunk (ha van), úgy hogy a kiinduló FOGLAL mátrix adott processzhez tartozó adathoz a processz kérését is hozzáadjuk.
3. A MAX.IGÉNY-ből kivonjuk a FOGLAL mátrixot, így megkapjuk a még kielégítetlen igényeket leíró mátrixot – ez lesz IGÉNY mátrix.
4. Erőforrás fajtánként összeadjuk a foglalva tartott erőforrások számait, majd ezeket kivonva az egyes erőforrások összdarab erőforrások számából kapjuk meg a pillanatnyilag rendelkezésre álló erőforrások készletét – KÉSZLET.
5. Megnézzük, hogy a KÉSZLET – ből kielégíthető-e valamelyik processz igénye – IGÉNY mátrixból.
6. **a,** Ha nincs ilyen processz, a rendszer nem biztonságos állapotban van. (jelen esetben a P4 kérése)
   b, Ha van ilyen processz, kielégítjük az igényét. (jelen esetben a P0 kérése)
7. A kiválasztott processz a lefutása után felszabadítja az összes általa használt erőforrást (ez a FOGLAL mátrix), azaz KÉSZLET új értékét kapjuk, ha az előző értékéhez hozzáadjuk a processz általeredetileg lefoglalva tartott erőforrások számát – FOGLAL megfelelő sorát.
8. Visszamegyünk az 5. lépésre, és folytatjuk.

   A feladat számítását, és megoldását a '**1.fel_Bankár algoritmus'** nevű exel fájl tartalmazza.

Az eredmény:

- Holtpontmentesség szempontjából a rendszer biztonságos. **P0 kérése** kielégíthető.
- Holtpontmentesség szempontjából a rendszer nem biztonságos, nem lehet **teljesíteni P4 kérését.** Nincs elég szabad hely egyetlen egy processz esetében sem a kielégítéshez.
- Nincs elég erőforrás ahhoz, hogy **P0 ÉS P4 kérését** egyszerre ki lehessen elégíteni

A feladat leírása:

2. Adott egy rendszer (foglalási stratégiák), melyben a következő

- Szabad területek: 30k, 35k, 15k, 25k, 75k, 45k és

- Foglalási igények: 39k, 40k, 33k, 20k, 21k állnak rendelkezésre.

Határozza meg *változó partíció esetén* a következő algoritmusok felhasználásával: first fit, next fit, best fit, worst fit a foglalási igényeknek megfelelő helyfoglalást!

A feladat elkészítésének lépései:

1. Táblázatba foglaljuk az adatokat, és onnan dolgozunk. A táblázat sorai lesznek a foglalási igények, és az oszlopok pedig a szabad területek. A táblázaton belül a szabad területek közül fogunk választani a sorban következő foglalási igénynek megfelelően.

2. 4 Stratégiát dolgozunk ki a feladatban. (First fit, Next Fit, Best fit, Worst fit)
   - Első megfelelő (First fit): A rendelkezésre álló szabad területek közül a legelső elegendő méretűt foglaljuk le.
     i. 39 a 75-öt foglalja le, mert a sorba ez a **legelső elegendő** méretű. (a táblázatba beírjuk a 39-et, és mellé, hogy mennyi szabad terület van még. 75-39, azaz 36)
     ii. Ezen logika alapján folytatjuk a soron következő foglalási igényeket. (amit kiválasztottunk már azt nem lehet még egyszer lefoglalni)

   - Következő megfelelő (Next Fit): Az Első megfelelő stratégiával szemben itt a keresést nem a tár elejéről kezdjük, hanem az után a terület után, amit legutoljára foglaltunk.
     i. 39 a 75-öt fogja lefoglalni, majd a 75-től indulva a 40 pedig a 45-öt (mert az van a 75 után, és bele is fér a 40 a 45-be). A táblázatba beírjuk a lefoglaltat, és mellé pedig hogy mennyi szabad hely van még.
     ii. Ezen logika mentén folytatjuk a soron következő foglalási igényeket. (amit kiválasztottunk már azt nem lehet még egyszer lefoglalni)

   - Legjobban megfelelő (Best fit): A legkisebbet foglaljuk le azon szabad területek közül, amelyek legalább akkorák, mint a lefoglalandó terület.
     i. Itt már a 39 a 45-öt fogja lefoglalni, mert ebbe belefér, igaz a 75-be is belefér, de nekünk most az kell amelyiknél a legkisebb szabad terület lesz. Azaz beírjuk a táblázatba a 45 helyű oszlopba a 39-et, és mellé írjuk a szabad helyet, ami most 45-39=6.
     ii. Ezen logika alapján folytatjuk a soron következő foglalási igényeket. (amit kiválasztottunk már azt nem lehet még egyszer lefoglalni)

   - Legrosszabban illeszkedő (Worst fit): Az elérhető legnagyobb szabad területet allokáljuk. A spekuláció az, hogy a maradék terület még talán elegendő lesz egy újabb foglalás számára
     i. A 39-et a 75-be fogjuk tenni, mert belefér, és legnagyobb lesz a szabad terület.
     ii. Itt lehet még egyszer kiválasztani amit lefoglaltunk, de csak úgy, ha maradt elegendő hely. (pl.: 75-ös oszlopba beosztottuk a 39-et, maradt 36, és a 21-est még beletehetjük a 36-ba, így marad 15 szabad terület)

Az eredmény:

Foglalási igény		Memória terület-szabad terület						
		30	35	15	25	75	45	45
	39	30	35	15	25	75	45	45
	40	30	35	15	25	75	45	45
	33	30	35	15	25	75	45	45
	20	30	35	15	25	75	45	45
	21	30	35	15	25	75	45	45

First Fit – Szabad területek közül a legelső elegendő méertű

Foglalási igény		Memória terület-szabad terület						
		30	35	15	25	75	45	45
	39	30	35	15	25	**39, 36**	45	45
	40	30	35	15	25	**39, 36**	**40, 5**	45
	33	30	**33, 2**	15	25	**39, 36**	**40, 5**	45
	20	**20, 10**	**33, 2**	15	25	**39, 36**	**40, 5**	45
	21	**20, 10**	**33, 2**	15	**21, 4**	**39, 36**	**40, 5**	45

Next Fit – A keresés az az után a terület után, amit legutoljára foglaltunk

Foglalási igény		Memória terület-szabad terület						
		30	35	15	25	75	45	45
	39	30	35	15	25	**39, 36**	45	45
	40	30	35	15	25	**39, 36**	**40, 5**	45
	33	30	35	15	25	**39, 36**	**40, 5**	**33, 12**
	20	**20, 10**	35	15	25	**39, 36**	**40, 5**	**33, 12**
	21	**20, 10**	**21, 14**	15	25	**39, 36**	**40, 5**	**33, 12**

Best Fit – A legkisebbet foglaljuk le

Foglalási igény		Memória terület-szabad terület						
		30	35	15	25	75	45	45
	39	30	35	15	25	75	**39, 6**	45
	40	30	35	15	25	75	**39, 6**	**40, 5**
	33	30	**33, 2**	15	25	75	**39, 6**	**40, 5**
	20	30	**33, 2**	15	**20, 5**	75	**39, 6**	**40, 5**
	21	**21, 9**	**33, 2**	15	**20, 5**	75	**39, 6**	**40, 5**

Worst Fit – Az elérhető legnagyobb szabad területet allokáljuk.

Foglalási igény		Memória terület-szabad terület						
		30	35	15	25	75	45	45
	39	30	35	15	25	**39, 36**	45	45
	40	30	35	15	25	**39, 36**	**40, 5**	45
	33	30	35	15	25	**39, 36**	**40, 5**	**33, 12**
	20	30	**20, 15**	15	25	**39, 36**	**40, 5**	**33, 12**
	21	30	35	15	25	**39, 21, 15**	**40, 5**	**33, 12**

A feladat számítását, és megoldását a **'2.fel_Foglalási strat'** nevű exel fájl tartalmazza.

A feladat leírása:

3. Adott egy igény szerinti lapozást használó rendszerben a következő laphivatkozás 3 és 4 fizikai memóriakeret a processzek számára.
Laphivatkozások sorrendje: 7 6 5 4 6 7 3 2 6 7 6 5 1 2 5 6 7 6 5 2
Memóriakeret (igényelt lapok): 3 és 4 memóriakeret.
Mennyi laphiba keletkezik (mindkét memóriakeret esetén külön-külön) az alábbi algoritmusok esetén: FIFO, LRU és SC? Hasonlítsa össze és magyarázza az eredményeket.

A feladat elkészítésének lépései:

1. A feladatot 3 kilapozási stratégiával oldjuk meg. (FIFO, LRU, SC), 3 és 4 memória keretre.
   - **FIFO:** A legrégebben belapozott, belapozási sor. A behozott lapokat a behozás sorrendje szerint sorba rendezi az algoritmus, majd ha szükséges, azt cseréli le amelyet a legrégebben töltődött be – tehát amelyik az utolsó a sorban.
     i. 4 memóriakeretesnél belapozzuk a 7,6,5,4 –et a 4 üres lapra. (laphivatkozás sorrendjében lapozunk, a sorrend adott, cserélni nem lehet!!) Bekerülnek a FIFO-ba, mert ki is lapozzuk őket.
     ii. A 6,7 –et már belapoztuk, ezért kihagyjuk.
     iii. A 3-at még nem lapoztuk be, de hogy be tudjuk lapozni, a legrégebbi belapozottal cseréljük, ami itt most a 7. Majd amit belapoztunk azokat továbbra is írjuk a FIFO-ba.
     iv. Végig haladunk a laphivatkozás sorrendjén ezen logika mentén.
     v. Végül a laphibák száma úgy jön ki, hogy amit üres helyre betettünk + amiket lapcseréltünk.
     vi. A 3 memóriakeretesnél is ugyan evvel a logikával végig haladunk, csak itt a lapok száma 3-ra csökken. Itt is kell írni a FIFO-ba, és ugyan így jön ki a laphibák száma is.

- **LRU:** A legrégebben nem használt. Ha egy lapot használnak (hivatkoznak rá) sor végére kerül. A lapcserélés áldozatát az alapján választja ki, hogy melyik volt legutoljára használva.
    i. 4 memóriakeretesnél belapozzuk a 7,6,5,4 –et a 4 üres lapra. (laphivatkozás sorrendjében lapozunk, a sorrend adott, cserélni nem lehet!!)
    ii. A 6,7 –et már belapoztuk, ezért kihagyjuk. (de attól függetlenül használtuk)
    iii. A 3-at még nem lapoztuk be, de hogy be tudjuk lapozni, a legutoljára belapozottal cseréljük le, ami a az 5-ös lesz.
    iv. Evvel a logikával haladunk végig a laphivatkozás sorrendjén.
    v. Végül a laphibák száma úgy jön ki, hogy amit üres helyre betettünk + amiket lapcseréltünk.
    vi. A 3 memóriakeretesnél is ugyan evvel a logikával végig haladunk, csak itt a lapok száma 3-ra csökken.
- **SC:** Minden laphoz tartozik egy hivatkozás bit. Ha egy adott lapra hivatkozunk, ezt 1-be állítjuk. Behozáskor is, hiszen azért hozzuk be a lapot, mert hivatkozunk rá.
    i. 4 memóriakeretesnél belapozzuk a 7,6,5,4 –et a 4 üres lapra, majd mellé írjuk az 1-et, mivel mindenki kap 1 esélyt. (laphivatkozás – amit most számmal, és mellé egy csillaggal jelölünk- sorrendjében lapozunk, a sorrend adott, cserélni nem lehet!!)
    ii. A 6,7 –et már belapoztuk, ezért kihagyjuk.
    iii. A 3-at még nem lapoztuk be, de hogy be tudjuk lapozni, a legrégebbi belapozottal cseréljük, ami itt most a 7. Viszont a 7-esnek van egy esélye, azt megadjuk, így lesz 7,0. Majd a következő oszlopokban folyamatosan veszlik el a belapozottaknak az esélye. Így tudjuk behozni a 3-at a 7 helyére. A 3 az 1 eséllyel indul. (Mindegyik újonnan belapozott megkapja az 1 esélyt.)
    iv. A 2-es sincs benne, be kell lapozni. A 2-t be tudjuk tenni a 6-os (0 esélyes) helyére. (a 6os volt a legrégebbi). Ugyanígy a 6-ost az 5-ösre, 7-est a 4-esre.
    v. A 6-ost nem kell belapozni, mivel benne van, és 1 esélyes, úgyhogy azt kihagyjuk. (nem csillagozzuk, nem írunk oda semmit)
    vi. Ha olyat lapozunk be ami már benne van, akkor meg kell nézni, hogy ami benne van annak mennyi esélye van. Ha 1 esélyes, akkor kihagyjuk, ha 0 esélyes akkor növeljük az esélyét, azaz 1-esre áll az esélye.(itt sem csillagozunk)
    vii. Ezen a logikán végig haladva jutunk el a laphivatkozás sorrend végére.
    viii. A FIFO-ba minden olyan szám bekerül amit belapoztunk, és aminek az esélyét csökkentettük.
    ix. Végül a laphibák száma úgy jön ki, hogy amit üres helyre betettünk + amiket lapcseréltünk.(csillagos számok)
    x. A 3 memóriakeretesnél is ugyan evvel a logikával végig haladunk, csak itt a lapok száma 3-ra csökken. Itt is kell írni a FIFO-ba, és ugyan így jön ki a laphibák száma is.

A feladat számítását, és megoldását a ’**3.fel_Lapozo’** nevű exel fájl tartalmazza.

Az eredmény:

FIFO – 4 Memória keretes:

	Lapcsere		Üres helyre tettük be
Laphibák száma:	10	+	4
Laphibák száma:	14		

FIFO – 3 Memória keretes:

	Lapcsere		Üres helyre tettük be
Laphibák száma:	13	+	3
Laphibák száma:	16		

LRU – 4 Memória keretes:

	Lapcsere		Üres helyre tettük be
Laphibák száma:	6	+	4
Laphibák száma:	10		

LRU – 3 Memória keretes:

	Lapcsere		Üres helyre tettük be
Laphibák száma:	12	+	3
Laphibák száma:	15		

SC – 4 Memória keretes:

	Lapcsere		Üres helyre tettük be
Laphibák száma:	**10**	+	4
Laphibák száma:	14		

SC – 3 Memória keretes:

	Lapcsere		Üres helyre tettük be
Laphibák száma:	**13**	+	3
Laphibák száma:	16		