JEGYZŐKÖNYV

Operációs rendszerek BSc

2022. tavasz féléves feladat

Készítette: **Pócsi Ákos**

Neptunkód: **ITUCXP**

A feladat leírása: Adott egy *igény szerinti lapozást* használó számítógéprendszer, melyben futás közben egy processz számára a következő laphivatkozással lehet hivatkozni : 6, 5, 4, 3, 5, 6, 2, 8, 5, 6, 5, 4, 7, 8, 4, 5, 6, 5, 5, 8
Memóriakeret (igényelt lapok): 3, ill. 4 memóriakeret.

Készítse el a laphivatkozások betöltését külön-külön táblázatba 3, ill. 4 memóriakeret esetén.
Mennyi laphiba keletkezik az alábbi algoritmusok esetén : LRU, FIFO?
Hasonlítsa össze és magyarázza az eredményeket!

A futtatás eredménye:

LRU	6	5	4	3	5	6	2	8	5	6	5	4	7	8	4	5	6	5	5	8
Memória-keret:3	6/0	6/1	6/2	3/0	3/1	3/2	2/0	2/1	2/2	6/0	6/1	6/2	7/0	7/1	7/2	5/0	5/1	5/0	5/0	5/1
	-	5/0	5/1	5/2	5/0	5/1	5/2	8/0	8/1	8/2	8/3	4/0	4/1	4/2	4/0	4/1	4/2	4/3	4/4	8/0
	-	-	4/0	4/1	4/2	6/0	6/1	6/2	5/0	5/1	5/0	5/1	5/2	8/0	8/1	8/2	6/0	6/1	6/2	6/3

14 db laphiba

LRU	6	5	4	3	5	6	2	8	5	6	5	4	7	8	4	5	6	5	5	8
Memória-keret:4	6/0	6/1	6/2	6/3	6/4	6/0	6/1	6/2	6/3	6/0	6/1	6/2	6/3	8/0	8/1	8/2	8/3	8/4	8/5	8/0
	-	5/0	5/1	5/2	5/0	5/1	5/2	5/3	5/0	5/1	5/0	5/1	5/2	5/3	5/4	5/0	5/1	5/0	5/0	5/1
	-	-	4/0	4/1	4/2	4/3	2/0	2/1	2/2	2/3	2/4	4/0	4/1	4/2	4/0	4/1	4/2	4/3	4/4	4/5
	-	-	-	3/0	3/1	3/2	3/3	8/0	8/1	8/2	8/3	8/4	7/0	7/1	7/2	7/3	6/0	6/1	6/2	6/3

10 db laphiba

FIFO	6	5	4	3	5	6	2	8	5	6	5	4	7	8	4	5	6	5	5	8
Memória-keret:3	6	5	4	3	3	6	2	8	5	6	6	4	7	8	8	5	6	6	6	6
	-	6	5	4	4	3	6	2	8	5	5	6	4	7	7	8	5	5	5	5
	-	-	6	5	5	4	3	6	2	8	8	5	6	4	4	7	8	8	8	8

14 db laphiba

FIFO	6	5	4	3	5	6	2	8	5	6	5	4	7	8	4	5	6	5	5	8
Memória-keret:4	6	5	4	3	3	3	2	8	5	6	6	4	7	8	8	5	6	6	6	6
	-	6	5	4	4	4	3	2	8	5	5	6	4	7	7	8	5	5	5	5
	-	-	6	5	5	5	4	3	2	8	8	5	6	4	4	7	8	8	8	8
	-	-	-	6	6	6	5	4	3	2	2	8	5	6	6	4	7	7	7	7

13 db laphiba

A feladat elkészítésének lépései:

LRU (Least Recently Used):
Az algoritmus mindig a legrégebben használt elemet cseréli le. Elkezdjük sorban feltölteni a memóriakeretet. Minden új laphivatkozás laphibát eredményez, tehát at első 3 vagy 4 memóriakeret laphiba. Aztán amíg a memóriakeretbe már ismert elem érkezik nincs laphiba, de egy új elem érkezése laphibát eredményez és a láncolt lista végén elhelyezkedő (legrégebben használt) elemet lecseréli a rendszer az új elemre.

FIFO (First-In-First-Out):
Az algoritmus a legelőször használt elemet cseréli le. Elkezdjük sorban feltölteni a memóriakeretet. Minden új laphivatkozás laphibát eredményez, tehát at első 3 vagy 4 memóriakeret laphiba. 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).

Összegzés: Az **LRU** algoritmus jóval bonyolultabb, viszont kevesebb laphibát eredményez nagyobb memóriakeretek esetében. A **FIFO** algoritmus sokkal egyszerűbb, viszont hátránya, hogy olyan lapokat is lecserél, amelyeket gyakran használ a rendszer.

A feladat leírása: Adott egy *igény szerinti lapozást* használó számítógéprendszer, melyben futás közben egy processz számára a következő laphivatkozással lehet hivatkozni : 1, 2, 3, 4, 0, 2, 5, 1, 2, 3, 4, 5, 1, 2,
Memóriakeret (igényelt lapok): 3, ill. 4 memóriakeret.

Készítse el a laphivatkozások betöltését külön-külön táblázatba 3, ill. 4 memóriakeret esetén.
Mennyi laphiba keletkezik az alábbi algoritmusok esetén : LRU, OPT?
Hasonlítsa össze és magyarázza az eredményeket!

A futtatás eredménye:

LRU	1	2	3	4	0	2	5	1	2	3	4	5	1	2
Memória-keret:3	1/0	1/1	1/2	4/0	4/1	4/2	5/0	5/1	5/2	3/0	3/1	3/2	1/0	1/1
	-	2/0	2/1	2/2	0/0	0/1	0/2	1/0	1/1	1/2	4/0	4/1	4/2	2/0
	-	-	3/0	3/1	3/2	2/0	2/1	2/2	2/0	2/1	2/2	5/0	5/1	5/2

13 db laphiba

LRU	1	2	3	4	0	2	5	1	2	3	4	5	1	2
Memória-keret:4	1/0	1/1	1/2	1/3	0/0	0/1	0/2	0/3	0/4	3/0	3/1	3/2	3/3	2/0
	-	2/0	2/1	2/2	2/3	2/0	2/1	2/2	2/0	2/1	2/2	2/3	1/0	1/1
	-	-	3/0	3/1	3/2	3/3	5/0	5/1	5/2	5/3	4/0	4/1	4/2	4/3
	-	-	-	4/0	4/1	4/2	4/3	1/0	1/1	1/2	1/3	5/0	5/1	5/2

12 db laphiba

OPT	1	2	3	4	0	2	5	1	2	3	4	5	1	2
Memória-keret:3	1	1	1	1	1	1	1	1	1	1	1	1	1	2
	-	2	2	2	2	2	2	2	2	3	4	4	4	4
	-	-	3	4	0	0	5	5	5	5	5	5	5	5

9 db laphiba

OPT	1	2	3	4	0	2	5	1	2	3	4	5	1	2
Memória-keret:4	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	-	2	2	2	2	2	2	2	2	2	2	2	2	2
	-	-	3	3	3	3	3	3	3	3	4	4	4	4
	-	-	-	4	0	0	5	5	5	5	5	5	5	5

7 db laphiba

A feladat elkészítésének lépései:

LRU (Least Recently Used):
Az algoritmus mindig a legrégebben használt elemet cseréli le. Elkezdjük sorban feltölteni a memóriakeretet. Minden új laphivatkozás laphibát eredményez, tehát at első 3 vagy 4 memóriakeret laphiba. Aztán amíg a memóriakeretbe már ismert elem érkezik nincs laphiba, de egy új elem érkezése laphibát eredményez és a láncolt lista végén elhelyezkedő (legrégebben használt) elemet lecseréli a rendszer az új elemre.

OPT (Optimal):
Az algoritmus mindig azt az elemet cseréli le, amelyiket nem használjuk majd a közeljövőben. Elkezdjük sorban feltölteni a memóriakeretet. Minden új laphivatkozás laphibát eredményez, tehát at első 3 vagy 4 memóriakeret laphiba. Aztán amíg a memóriakeretbe már ismert elem érkezik nincs laphiba, de egy új elem érkezése laphibát eredményez és azt az elemet cseréli le az algoritmus, amelyiket nem- vagy legkésőbb kell újra használnunk.

Összegzés: Az **LRU** algoritmus jóval bonyolultabb és több laphibát eredményez.
Az **OPT** algoritmus sokkal egyszerűbb és sokkal optimálisabb is, azonban a használatához előre kell látni az elkövetkezendő elemeket, ami a valóságban lehetetlen.