

Operációs rendszerek BSc

11. Gyak.

2022. 04. 27.

Készítette:

Berecz Antónia Bsc

Programtervező informatikus

LV8GJH

Miskolc, 2022

1. 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.

A rendszerben a memória 4 kbyte-os blokkokban kerül nyilvántartásra, ennél kisebb méretű töredék igény esetén a teljes blokk lefoglalásra kerül.

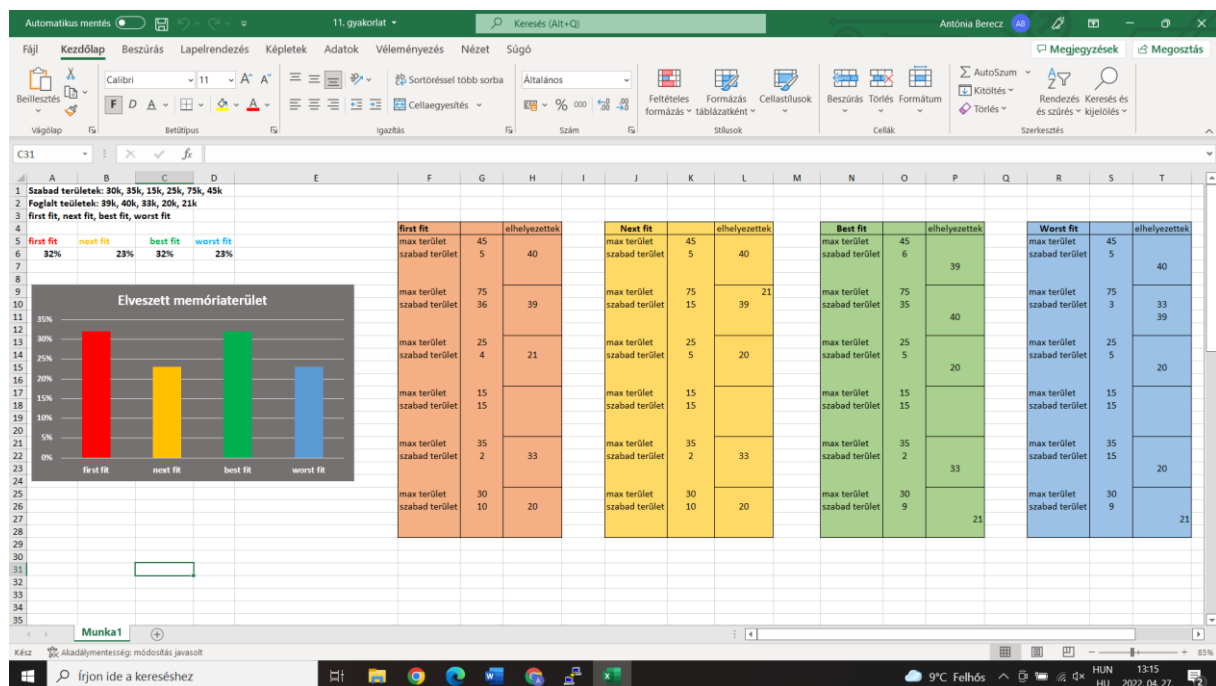
Határozza meg változó méretű partíció esetén a következő algoritmusok felhasználásával:

first fit, next fit, best fit, worst fit a foglalási igényeknek megfelelő helyfoglalást – táblázatos formában (az ea. bemutatott mintafeladat alapján)!

Hasonlítsa össze, hogy a teljes szabad memóriaterület hány százaléka vész el átlagosan az egyes algoritmusok esetén!

A kapott eredményeket ábrázolja oszlop diagrammal!

Magyarázza a kapott eredményeket és hogyan lehet az eredményeket javítani!



2. Gyakorló feladat:

- Írjon C nyelvű programokat, ahol kreál/azonosít szemafor készletet, benne N szemafor-t. A kezdő értéket 0-ra állítja – semset.c,
- kérdezze le és írja ki a pillanatnyi szemafor értéket – semval.c
- szüntesse meg a példácskák szemafor készletét – semkill.c
- sembuf.sem_op=1 értékkel inkrementálja a szemafort – semup.c

```
UW PICO 5.05 File: semset.c Modified
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <stdio.h>
#include <stdlib.h>

#define KEY 20010606
#define PERM 00666

main() {
    int id;

    if ((id = semget(KEY, 1, 0)) < 0) {
        if ((id = semget(KEY, 1, PERM | IPC_CREAT)) < 0) {
            perror(" A szemafor nem nyitható meg. ");
            exit(-1);
        }
    }
    else {
        perror(" Már létezik a szemafor. ");
        exit(0);
    }

    if (semctl(id, 0, SETVAL, 0) < 0) {
        perror(" Nem lehetett inicializálni. ");
    }
    else {
        puts(" Kész és inicializált a szemafor. ");
    }

    exit(0);
}

UW PICO 5.05 File: semval.c Modified
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <stdio.h>
#include <stdlib.h>

#define KEY 20010606
#define PERM 00666

int main() {
    int id;
    int ertek;

    if ((id = semget(KEY, 1, 0)) < 0) {
        perror(" Már nem létezik a szemafor. ");
        exit(-1);
    }

    if ((ertek = semctl(id, 0, GETVAL)) < 0) {
        perror(" Nem lehetett lekérdezni az értékét. ");
    }
    else {
        printf(" Kész és a szemafor értéke %d. ", ertek);
    }

    exit(0);
}

UW PICO 5.05 File: semkill.c Modified
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <stdio.h>
#include <stdlib.h>

#define KEY 20010606
#define PERM 00666

int main() {
    int id;

    if ((id = semget(KEY, 1, 0)) < 0) {
        perror(" Már nem létezik a szemafor. ");
        exit(-1);
    }

    if (semctl(id, 0, IPC_RMID, 0) < 0) {
        perror(" Nem sikerült törölni. ");
        exit(-1);
    }

    puts(" A szemafort megszüntettük. \n");
    exit(0);
}
```

```
UW FICO 5.05 File: semup.c Modified
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <stdio.h>
#include <stdlib.h>

#define KEY 20010606
#define PERM 00666

int main() {
    int id;
    struct sembuf up = {0, 1, 0};

    if ((id = semget(KEY, 1, 0)) < 0) {
        perror(" Még nem létezik a szemafor. ");
        exit(-1);
    }

    puts(" Itt fut a nem kritikus szakasz. ");

    puts(" Itt fut a kritikus szakasz. ");
    semop(id, &up, 1);

    puts(" Itt ismét nem kritikus szakasz fut. ");

    exit(0);
}
```

2a. Írjon egy C nyelvű programot, melyben

- egyik processz létrehozza a szemafor (egyetlen elemi szemafor; inicializálja 1-re, vagy x-re, ha még nem létezik),
- másik processz használja a szemafor, belépési szakasz (down), a kritikus szakaszban alszik 2-3 sec-et, m pid-et kiír, kilépési szakasz (up), ezt ismételve 2x-3x (és a hallgató egyszerre indítson el 2-3 ilyen processzt),
- harmadik processzben, ha létezik a szemafor, akkor megszünteti.

```
jenyituni-miskolc.hu - PuTTY
UW PICO 5.05
File: gyakorlat_2_1.c
Modified

#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <stdio.h>
#include <stdlib.h>

#define KEY 20010606
#define PERM 00666

main() {
    int id;

    if ((id = semget(KEY, 1, 0)) < 0) {
        if ((id = semget(KEY, 1, PERM | IPC_CREAT)) < 0) {
            perror("A szemafor nem nyitható meg. ");
            exit(-1);
        }
    }
    else {
        perror("Már létezik a szemafor. ");
        exit(0);
    }

    if (semctl(id, 0, SETVAL, 1) < 0) {
        perror("Nem lehetett inicializálni. ");
    }
    else {
        puts("Kész és inicializált a szemafor. ");
    }

    exit(0);
}
```

```
jenyituni-miskolc.hu - PuTTY
UW PICO 5.05
File: gyakorlat_2_2.c
Modified

#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <stdio.h>
#include <stdlib.h>

#define KEY 20010606
#define PERM 00666

int main() {
    int id;
    struct sembuf up = {0, 1, 0};
    struct sembuf down = {0, -1, 0};

    if ((id = semget(KEY, 1, 0)) < 0) {
        perror("A szemafor nem létezik!");
        exit(-1);
    }

    printf("A szemafor létezik!\n\n");

    int i = 0;

    for(i = 0; i<5; i++){
        semop(id, &down, 1);

        sleep(2);
        printf("PID = %d\n", getpid());

        semop(id, &up, 1);
    }

    exit(0);
}
```

```
jenyituni-miskolc.hu - PuTTY
UW PICO 5.05
File: gyakorlat_2_3.c
Modified

#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <stdio.h>
#include <stdlib.h>

#define KEY 20010606
#define PERM 00666

int main() {
    int id;

    if ((id = semget(KEY, 1, 0)) < 0) {
        perror("Már nem létezik a szemafor. ");
        exit(-1);
    }

    if (semctl(id, 0, IPC_RMID, 0) < 0) {
        perror("Nem sikerült törölni.");
        exit(-1);
    }

    puts("A szemafor megszüntetett. \n");
    exit(0);
}
```