**Operációs rendszerek BSc**

10. Gyak.

2022. 04. 13.

**Készítette:**

Gerőcs Gergő Bsc

Mérnökinformatikus

FEU2E5

# Miskolc, 2022

A tanult rendszerhívásokkal (open(), read()/write(), close() - ők fogják a rendszerhívásokat tovább hívni - írjanak egy neptunkod\_openclose.c programot, amely megnyit egy fájlt – neptunkod.txt, tartalma: hallgató neve, szak , neptunkod.

A program következő műveleteket végezze:

* olvassa be a neptunkod.txt fájlt, melynek attribútuma: O\_RDWR
* hiba ellenőrzést,
* write() - mennyit ír ki a konzolra.
* read() - kiolvassa a neptunkod.txt tartalmát és mennyit olvasott ki (byte), és kiírja konzolra.
* lseek() – pozícionálja a fájl kurzor helyét, ez legyen a fájl eleje: SEEK\_SET, és kiírja a konzolra

A képen szöveg látható

Automatikusan generált leírás

Készítse el a következő feladatot, melyben egy szignálkezelő több szignált is tud kezelni:

1. Készítsen egy szignál kezelőt (handleSignals), amely a SIGINT (CTRL + C) vagy SIGQUIT (CTRL + \) jelek fogására vagy kezelésére képes.
2. Ha a felhasználó SIGQUIT jelet generál (akár kill paranccsal, akár billentyűzetről a CTRL+ \) a kezelő egyszerűen kiírja az üzenetet visszatérési értékét – a konzolra.
3. Ha a felhasználó először generálja a SIGINT jelet (akár kill paranccsal, akár billentyűzetről a CTRL + C), akkor a jelet úgy módosítja, hogy a következő alkalommal alapértelmezett műveletet hajtson végre (a SIG\_DFL) – kiírás a konzolra.
4. Ha a felhasználó másodszor generálja a SIGINT jelet, akkor végrehajt egy alapértelmezett műveletet, amely a program befejezése - kiírás a konzolra.

Mentés: neptunkod\_tobbszignal.c

A képen szöveg látható

Automatikusan generált leírás

Adott a következő ütemezési feladat, amit a FCFS, SJF és Round Robin (RR:4ms) ütemezési algoritmus alapján határozza meg következő teljesítmény értékeket, metrikákat (külön-külön táblázatba):

**FCFS:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **P1** |  | **P2** |  | **P3** |  | **P4** |  |
| **Érkezés** |  | 0 |  | 0 |  | 2 |  | 5 |
| **CPU idő** |  | 24 |  | 3 |  | 6 |  | 3 |
| **Indulás** |  | 0 |  | 24 |  | 27 |  | 33 |
| **Befejezés** |  | 24 |  | 27 |  | 33 |  | 36 |
| **Várakozás** |  | 0 |  | 24 |  | 25 |  | 28 |
| **Körülfordulási idő** |  | 24 |  | 27 |  | 31 |  | 31 |

**SJF:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **P1** |  | **P2** |  | **P3** |  | **P4** |  |
| **Érkezés** |  | 0 |  | 0 |  | 2 |  | 5 |
| **CPU idő** |  | 24 |  | 3 |  | 6 |  | 3 |
| **Indulás** |  | 12 |  | 0 |  | 3 |  | 9 |
| **Befejezés** |  | 36 |  | 3 |  | 9 |  | 12 |
| **Várakozás** |  | 12 |  | 0 |  | 1 |  | 4 |
| **Várakozók** |  | - |  | P1, P3 |  | P1, P4 |  | P1 |
| **Körülfordulási idő** |  | 36 |  | 3 |  | 7 |  | 7 |

Sorrend: P2, P3, P4, P1

**RR:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **RR: 10 ms** | **P1** |  |  | **P2** |  | **P3** |  | **P4** |  |
| **Érkezés** | 0 | *4* | *15* |  | 0 | 2 | *11* |  | 5 |
| **CPU idő** | 24 | *20* | *16* |  | 3 | 6 | *2* |  | 3 |
| **Indulás** | 0 | *11* | *20* |  | 4 | 7 | *18* |  | 15 |
| **Befejezés** | 4 | *15* | *36* |  | 7 | 11 | *20* |  | 18 |
| **Várakozás** | 0 | *7* | *5* |  | 4 | 5 | *7* |  | 10 |
| **Körülfordulási idő** |  | 36 |  |  | 7 | 1 | 8 |  | 13 |

Sorrend: P1, P2, P3, P1, P4, P3, P1

Külön táblázatba számolja a teljesítmény értékeket!

CPU kihasználtság: számolni kell a cs: 0,1(ms) és sch: 0,1 (ms)

|  |  |
| --- | --- |
| Algoritmus neve | FCFS |
| CPU kihasználtság | 36/36.4 = 98.9 % (4 db cs) |
| Körülfordulási idők átlaga | (24+27+31+31)/4 = 28.25 ms |
| Várakozási idők átlaga | (0+24+25+28)/4 = 19.25 ms |
| Válaszidők átlaga | (0+24+25+28)/4 = 19.25 ms |

|  |  |
| --- | --- |
| Algoritmus neve | SJF |
| CPU kihasználtság | 36/36.4 = 98.9 % (4 db cs) |
| Körülfordulási idők átlaga | (36+3+7+7)/4 = 13.25 ms |
| Várakozási idők átlaga | (12+0+1+4)/4 = 4.25 ms |
| Válaszidők átlaga | (12+0+1+4)/4 = 4.25 ms |

|  |  |
| --- | --- |
| Algoritmus neve | RR |
| CPU kihasználtság | 36/37 = 97.3% (7 db cs, 3 db sch) |
| Körülfordulási idők átlaga | (36+7+18+13)/4 = 18.5 ms |
| Várakozási idők átlaga | (0+7+5+4+5+7+10)/4 = 9.5 ms |
| Válaszidők átlaga | (0+4+5+10)/4 = 4.75 ms |

**Gyakorló feladatok:**

1. **feladat.**

Írjon C nyelvű programot, amelyik kill() seg.-vel SIGALRM-et küld egy argumentumként megadott PID-u processznek, egy másik futó program a SIGALRM-hez rendeljen egy fv.-t amely kiírja pl. neptunkodot, továbbá pause() fv.-el blokkolódjon, majd kibillenés után jelezze, hogy kibillent és terminálódjon.

A képen szöveg látható

Automatikusan generált leírásMentés. neptunkod\_gyak9\_1.c

1. **feladat.**

**A képen szöveg látható

Automatikusan generált leírás**Írjon C nyelvű programot, amelyik a SIGTERM-hez hozzárendel egy fv-t., amelyik kiírja az int paraméter értéket, majd végtelen ciklusban fusson, 3 sec-ig állandóan blokkolódva elindítás után egy másik shell-ben kill paranccsal (SIGTERM) próbálja terminálni, majd SIGKILL-el.” Mentés. neptunkod\_gyak9\_2.c