Popis úlohy

Implementujte v jazyce C modifikovaný synchronizační problém Roller Coaster (můžete se inspirovat knihou The Little Book of Semaphores). Existují dva typy procesů, vozík a pasažéři. Pasažéři nastupují do vozíku, který má omezenou kapacitu. Jakmile je vozík plný, vyráží na trať. Další pasažéři musejí počkat, až se vozík vrátí a všichni z něj vystoupí.

Spuštění

$ ./proj2 P C PT RT kde

P je počet procesů reprezentujících pasažéra; P > 0. C je kapacita vozíku; C > 0, P > C, P musí být vždy násobkem C. PT je maximální hodnota doby (v milisekundách), po které je generován nový proces pro pasažéra; PT >= 0 && PT < 5001. RT je maximální hodnota doby (v milisekundách) průjezdu trati; RT >= 0 && RT < 5001. Všechny parametry jsou celá čísla.

Implementační detaily

Pracujte s procesy, ne s vlákny. Každému pasažérovi odpovídá jeden proces passenger. Existuje pouze jeden vozík, kterému odpovídá jeden proces car. Hlavní proces vytváří pomocný proces pro generování procesů pasažérů pomocný proces generuje procesy pro pasažéry; každý nový proces je generován po uplynutí náhodné doby z intervalu <0, PT>; celkem vygeneruje P procesů. postupně tedy vznikne hlavní proces, jeden pomocný proces, jeden proces vozíku a P procesů pasažérů. Každý proces passenger i car bude interně identifikován celým číslem I, začínajícím od 1. Číselná řada je pro každou kategorii procesů zvlášť. Každý proces passenger i car vykonává své akce a současně zapisuje informace o akcích do souboru s názvem proj2.out. Přístup k výstupnímu zařízení (zápis informací) musí být výlučný; pokud zapisuje jeden proces a další chce také zapisovat, musí počkat na uvolnění zdroje. Součástí výstupních informací o akci je pořadové číslo A prováděné akce (viz popis výstupů). Akce se číslují od jedničky. Použijte sdílenou paměť pro implementaci čítače akcí a sdílených proměnných nutných pro synchronizaci. Použijte semafory pro synchronizaci procesů. Nepoužívejte aktivní čekání (včetně cyklického časového uspání procesu) pro účely synchronizace. Procesy, které již dokončily všechny akce, čekají na všechny ostatní procesy; všechny procesy passenger i car se ukončí současně. Hlavní proces čeká na ukončení všech vytvořených procesů. Poté se ukončí s kódem (exit code) 0. Budete-li potřebovat generovat unikátní klíč, je vhodné

použít funkci ftok. Další funkce a systémová volání: fork, wait, shmat, semctl, semget, shmget, sem_open, usleep, ... Chybové stavy

Pokud některý ze vstupů nebude odpovídat očekávanému formátu nebo bude mimo povolený rozsah, program vytiskne chybové hlášení na standardní chybový výstup, uvolní všechny dosud alokované zdroje a ukončí se s kódem (exit code) 1. Pokud selže systémové volání, program vytiskne chybové hlášení na standardní chybový výstup, uvolní všechny alokované zdroje a ukončí se s kódem (exit code) 2. Popis procesů a jejich výstupů

Poznámka k výstupům:

A je pořadové číslo prováděné akce, NAME je zkratka kategorie příslušného procesu, tj. P nebo C, I je interní identifikátor procesu v rámci příslušné kategorie, Při vyhodnocování výstupu budou ignorovány mezery a tabelátory. Proces car

Po spuštění tiskne A: NAME I: started. Proces pracuje v cyklu, počet iterací je P/C V každé iteraci: proces vyvolá operaci load (zahájení nastupování); tiskne A: NAME I: load proces čeká, dokud nenastoupí všichni pasažéři jakmile je vozík plný další pasažéři nemohou nastoupit proces vyvolá operaci run (jízda vozíku); tiskne A: NAME I: run proces se uspí na náhodnou dobu z intervalu <0, RT> po probuzení vyvolá proces operaci unload (zahájení vystupování); tiskne A: NAME I: unload proces nemůže zahájit operaci load, dokud nevystoupí všichni pasažéři Těsně před ukončením proces tiskne A: NAME I: finished. Proces passenger

Po spuštění tiskne A: NAME I: started. Proces čeká na splnění podmínek nastupování (proces car vyvolal operaci load a je volné místo ve vozíku). Pokud jsou splněny podmínky nastupování, proces vyvolá operaci board (nastoupení do vozíku) tiskne A: NAME I: board pokud není proces poslední nastupující pasažér, tiskne A: NAME I: board order N, kde N je pořadí, v kterém proces nastoupil do vozíku pokud je proces poslední nastupující pasažér, tiskne A: NAME I: board last Poté, co proces car vyvolá operaci unload proces zahájí operaci unboard (vystoupení z vozíku); A: NAME I: unboard pokud není proces poslední vystupující pasažér, tiskne A: NAME I: unboard order N, kde N je pořadí, v kterém proces vystoupil z vozíku pokud je proces poslední vystupující pasažér, tiskne A: NAME I: unboard last Těsně před ukončením proces tiskne A: NAME I: finished. Společné podmínky

Všechny procesy passenger a car se ukončí současně, tj. čekají, až všichni dokončí operaci unboard nebo hlavní cyklus. Teprve poté tisknou informaci ... finished. Pokud má RT nebo PT hodnotu 0, znamená to, že na příslušném místě nedojde k uspání procesu. Ukázka výstupů

Ukázka č. 1

Spuštění: $ ./proj2 4 2 0 0

Výstup (proj2.out):

```
1     : C 1     : started
2     : C 1     : load
3     : P 1     : started
4     : P 1     : board
5     : P 1     : board order 1
6     : P 2     : started
7     : P 2     : board
8     : P 2     : board last
9     : C 1     : run
10    : P 3     : started
11    : C 1     : unload
12    : P 1     : unboard
13    : P 1     : unboard order 1
14    : P 4     : started
15    : P 2     : unboard
16    : P 2     : unboard last
17    : C 1     : load
18    : P 4     : board
19    : P 3     : board
20    : P 4     : board order 1
21    : P 3     : board last
22    : C 1     : run
23    : C 1     : unload
24    : P 4     : unboard
25    : P 4     : unboard order 1
26    : P 3     : unboard
27    : P 3     : unboard last
28    : P 4     : finished
29    : C 1     : finished
30    : P 1     : finished
31    : P 3     : finished
32    : P 2     : finished
```