Compte rendu tp System d’exploitation master 1 Rsd

Etudiant: Bouzara zakaria
Matricule: 212138069681

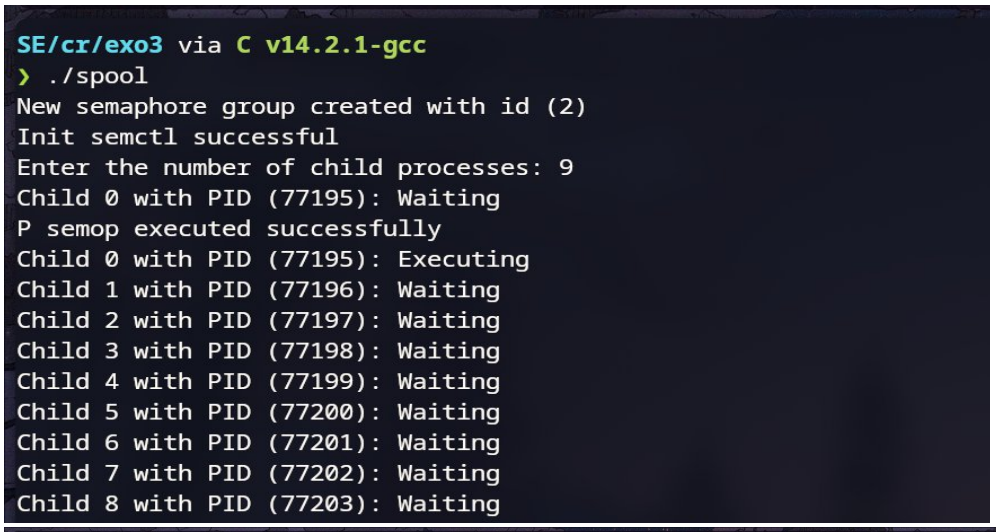
Exercice 3:

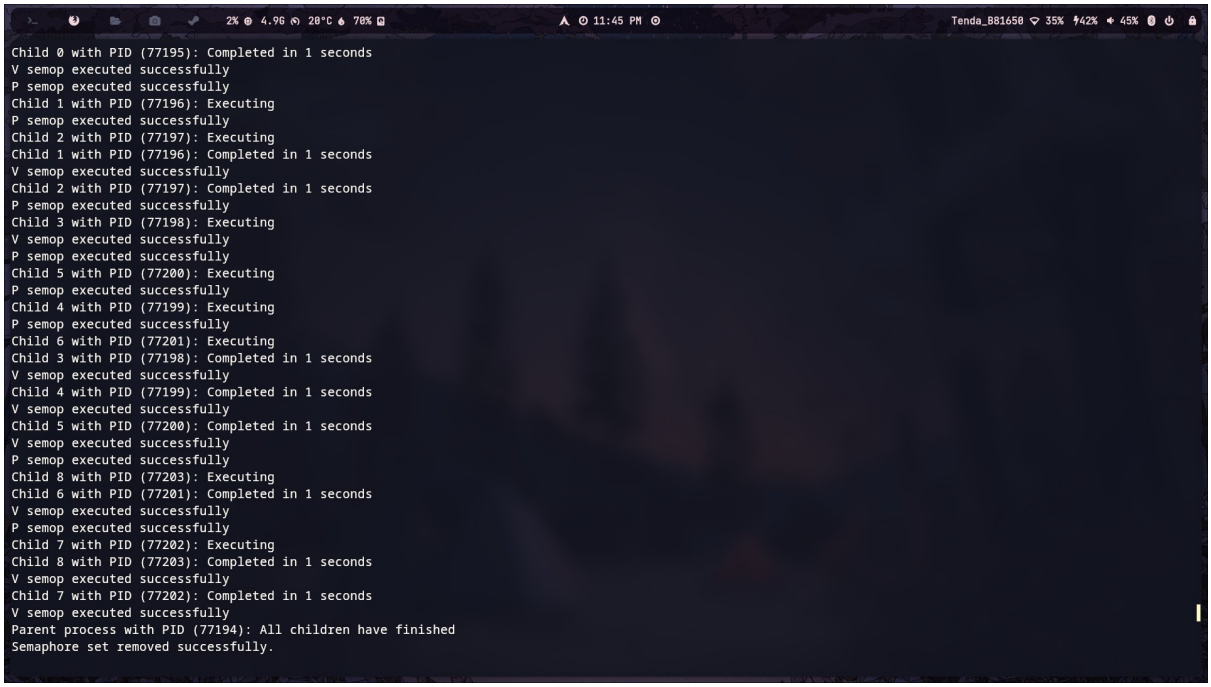
1. les codes attaches:

exo3 /

- semaphore.h
- spool.c

2. Resultat d’execution:





Si un processus se termine de manière inattendue (par exemple, en recevant le signal SIGKILL via la commande `kill -9 <pid>`) alors qu'il utilise une ressource partagée, la valeur du sémaphore peut rester décrémentée, bloquant ainsi l'accès à la ressource indéfiniment. Cela se produit parce que l'opération P (appel à `semop` avec `sem_op = -1`) décrémente le sémaphore, mais l'opération V correspondante (appel à `semop` avec `sem_op = 1`) n'est jamais exécutée en raison de l'arrêt prématuré du processus. Pour remédier à cette situation, utilisez le drapeau `SEM_UNDO` lors de l'opération P(`semop`).

Ce drapeau assure que toute décrémentation effectuée sur le sémaphore par un processus est automatiquement annulée par le système d'exploitation si ce processus se termine de manière inattendue.

Exercice 4:

1. les codes attaches:

exo4 /

- Create.c
- Consumer.c
- Producer.c
- semaphore.h

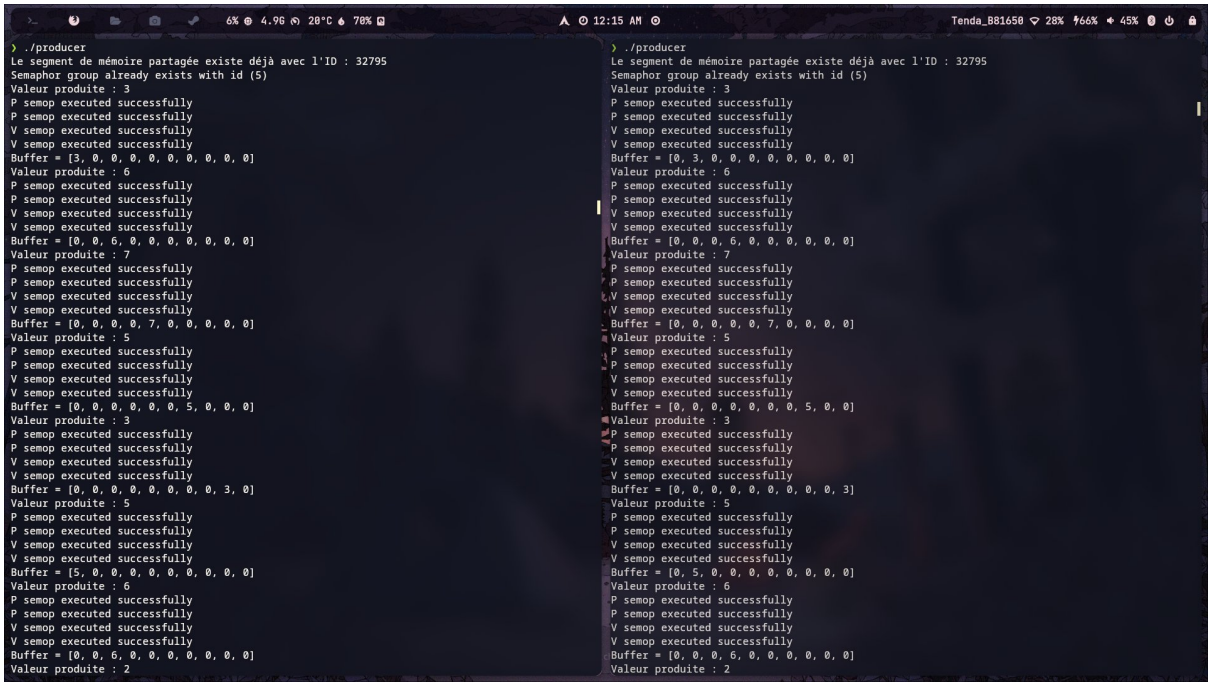
2. Resultat d'execution:

Create.c:

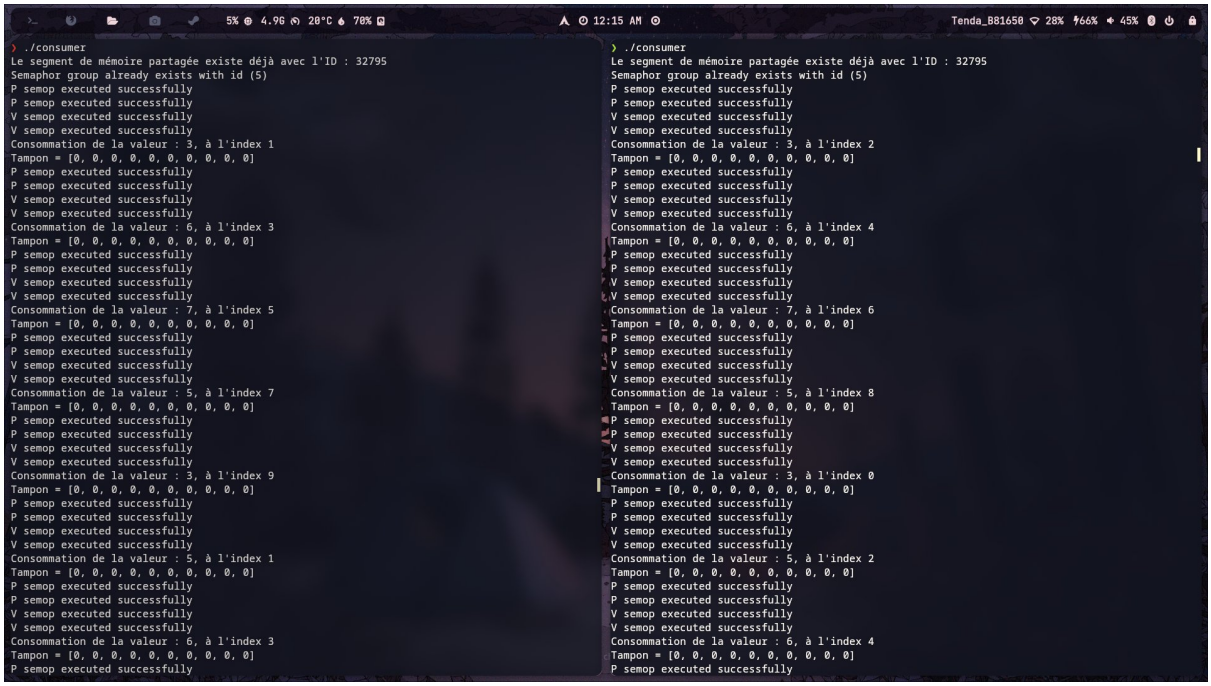
```
SE/cr/exo4 via C v14.2.1-gcc took 3m20s
❯ gcc -o create create.c

SE/cr/exo4 via C v14.2.1-gcc
❯ ./create
Nouveau segment de mémoire partagée créé avec l'ID : 32795
Indexes initialisés (Producteur : 0, Consommateur : 0)
New semaphore group created with id (5)
Init semctl successful
Init semctl successful
Init semctl successful
Init semctl successful
```

L'execution en paralelle:

2 producteurs



2 Consomateurs:



Si le producteur est arrêté :

Le consommateur pourra toujours accéder à la mémoire partagée et vérifier le tampon pour consommer des valeurs. Cependant, comme le producteur ne produit plus de nouvelles valeurs, le consommateur trouvera constamment le tampon vide (les valeurs resteront à 0). Le sémaphore empty_slots (sémaphore 0) restera inchangé car le producteur ne met pas à jour le tampon. Ce sémaphore devrait être dans un état où le consommateur attend que le sémaphore empty_slots soit signalé par le producteur. Mais comme aucune valeur n'est produite, le consommateur attendra indéfiniment.

Si le consommateur est arrêté :

Le producteur continuera à produire des valeurs et à les placer dans le tampon, en mettant à jour tab et en signalant le sémaphore full_slots (sémaphore 1) pour indiquer que des données sont disponibles pour la consommation. Cependant, comme le consommateur ne consomme pas les éléments, le sémaphore full_slots continuera à augmenter, mais le sémaphore empty_slots ne sera pas libéré. Cela peut amener le producteur à se bloquer ou à ne pas pouvoir insérer de nouvelles valeurs une fois que le tampon est plein, car le producteur attend de l'espace dans le tampon (empty_slots).

Exercice 4:

les codes attaches:

exo5 /

- Create.c
- Oxygen.c
- semaphore.h

1. si on arrête un des processus :

Si vous arrêtez un des processus (hydrogen ou oxygen), l'autre processus continuera à produire des valeurs jusqu'à ce que le tampon soit plein, puis il se bloquera en attendant que le consommateur consomme des valeurs.

À l'inverse, si vous arrêtez le producteur (oxygen), le consommateur (hydrogen) se bloquera en attendant que de nouvelles valeurs soient produites une fois qu'il aura consommé toutes les valeurs actuelles dans le tampon.