CC - MEEF NSI Système d'exploitation

2 heures

9 juin 2022

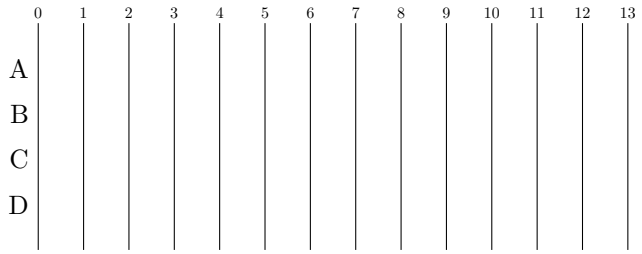
Afin d'obtenir tous les points il vous est demandé de justifier vos résultats. Le barème proposé est susceptible d'être modifié lors de la correction. Il n'est présent que pour vous donner une idée du poids relatif des différentes questions.

I Ordonnancement

- Le système gère des listes de priorités.
- On se place en mode préemptif c'est à dire que lorsqu'une tâche plus prioritaire arrive, le système lui donne immédiatement accès au processeur.
- Pour des tâches de même priorité, le système utilise le `Round Robin` avec un quantum de 2.
- *Précision :* Pour Round Robin, une tâche qui perd la main perd son tour même si son quantum d'exécution n'était pas terminé.

Tâche	Date(s) d'arrivée(s)	Priorité	Durée	Remarque
A	0	4	3	
B	0	5	2	
C	1	4	4	
D	1	5	4	

Q.I.1) - Dans la figure suivante, faites l'ordonancement sur 13 unités de temps.



II L'anniversaire de Gergovie

Pour l'anniversaire de la bataille de Gergovie, le village d'Astérix organise un banquet romain. C'est un banquet festif qui combine les plaisirs de la table et l'attaque d'un camp romain. Les convives sont invités à alternativement attaquer les romains et se restaurer ce qu'il font avec joie.

Cependant, il y a concurence pour l'accès à la nourriture (les romains étant assez nombreux pour combler les besoins de tous). Chaque convive recupère un nombre de plat fixe dépendant de son appétit et s'il l'obtient le mange. Puis frappe un romain. Il est important que le banquet ne connaisse pas de temps mort. Si un convive ne trouve pas suffisamment de plat, il saute simplement l'étape de la nouriture et ne doit en aucun cas attendre.

Un début de code sequentiel vous est donné. La fonction principale lance la fonction `cuisine` qui prépare les plats et une fonction `convive` qui simule les actions d'un convive particulier. Pour le passage des plats entre la cuisine et les convives, il a été mis en place une simple liste `plats_prets` partagée par toutes ces fonctions.

L'algorithme d'un convive est donc de :

— Vérifier qu'un nombre de plats suffisant est disponible pour lui et si c'est le cas récupérer les plats dans la liste
— Manger les plats récupérés ou rien s'il n'y avait pas suffisement à manger
— frapper un romain
— recommencer tant que la cuisine prépare des plats ou qu'il en reste suffisement.

Mais pour le moment le code est séquentiel. C'est le premier convive arrivé (Obélix bien sur) qui mange les plats ne laissant rien aux autres. Vous devez rendre ce code multithread.

Q.II.1) - Remplacer l'appel de la fonction `cuisine` par un thread qui fait la même chose.

Q.II.2) - Remplacer les appels de la fonction `convive` par des threads.

Q.II.3) - Idefix vous soufle que l'utilisation d'une simple liste pour `plats_prets` n'est pas une bonne chose, a-t-il raison ? Pourquoi ? Si oui, faites le remplacement par une structure de données appropriée.

Q.II.4) - Vous devriez constater que le programme se bloque (ou avoir détecté un problème), expliquez le problème en donnant un exemple de ce qui peut se passer et corrigez le code.

III Chemin dans l'arbre des fichiers

Je vous présente ici le résultat de 2 commandes sur mon ordinateur :

```
[frico@fedora cours-capes]$ pwd
/home/frico/Cours/SystemCapes/cours-capes
[frico@fedora cours-capes]$ ls -R
.:
Annales       cm-system       header-cm.tex  latexlib   notes.rst  README.md
tp-sys-linux  tp-sys-linux-2  tp-sys-py1     tp-sys-py2

./Annales:
20220609  20220609.tex  latexlib  Makefile

./Annales/20220609:
anniversaire_gergovie_corr.py  anniversaire_gergovie.py  Makefile

./cm-system:
cm-systeme.pdf  cm-systeme.tex  Makefile

./tp-sys-linux:
Makefile  prelude.tex  src  tp-sys-linux.tex

./tp-sys-linux/src:
code1.c  code2.c  code3.c  code4.c  code.py  testTP

./tp-sys-linux-2:
fs_repertoire_arborescence.jpg  Makefile  prelude.tex  src  tp-sys-linux-2.tex

./tp-sys-linux-2/src:
mail.eml  moliere_jalousie_barbouille.epub  moliere.sh

./tp-sys-py1:
Makefile  prelude.tex  src  tp-sys-py1.tex

./tp-sys-py1/src:
dix-procs.py        N-procs-queue.py  N-procs-value.py   open.py      tst-fork.py
tst-process-mem.py  tst-process.py    volsphere-proc.py  volsphere.py
volsphere-thread.py volsphere-trav.py

./tp-sys-py2:
Makefile  prelude.tex  src  tp-sys-py2.tex
```

```
./tp-sys-py2/src:
koch.py                          koch_thread.py  tst_thread1.py  tst_thread_account.py
tst_thread_count1.py             tst_thread_count_boucle.py      tst_thread_count.py
koch_thread_marchepas.py         tst_thread2.py                  tst_thread_count0.py
tst_thread_count_boucle_lock.py  tst_thread_count_lock.py
```

Q.III.1) - Que fait la comande `pwd` ?

Q.III.2) - La commande `ls -R` permet de voir l'arborescence des fichiers concernant ce cours.

2(a) - Quel est le chemin absolu vers la correction du sujet de programmation c'est à dire le fichier `anniversaire_gergovie_corr.py` (dans le sous répertoire Annales)

2(b) - Je souhaite copier le fichier `code.py` (dans un sous répertoire de `tp-sys-linux`) vers le répertoire `tp-sys-py1`. Quel commande tapez-vous en n'utilisant que des chemin relatifs ? Vous préciserez bien la commande et le répertoire dans lequel vous l'exécutez.