IF-3-SYS : Contrôle continu. Avril 2019.

NOM Prénom :

Durée : 30 min
Les réponses sont à inscrire sur le sujet.
Commencez par écrire votre nom tout de suite. Vraiment.
Lisez le sujet en entier (13 questions) avant de commencer à travailler.
Documents et appareils interdits, sauf une feuille A4 recto-verso manuscrite.
Pour les calculs en binaire, vous pouvez vous aider du tableau donné en page 4.
Prenez la peine d'écrire lisiblement. Utilisez un brouillon plutôt que de faire des ratures.
Dans les questions vrai/faux, les erreurs sont comptées négativement : ne répondez pas au hasard.

1 Noyau, processus, appels système

Question 1 Pour chaque affirmation ci-dessous, indiquez si elle est correcte (entourez «V») ou si elle est fausse (entourez «F»).

V F Le noyau implémente un processeur virtuel pour chaque processus en cours d'exécution.
V F Le noyau interprète chaque instruction du programme au fur et à mesure de l'exécution.
V F Une action de l'utilisateur, par ex. un clic de souris, provoque en général un appel système.
V F Une trappe est une IRQ causée par un programme plutôt que par un évènement extérieur.

Question 2 Un processus parent crée un processus fils en invoquant la fonction `fork()`, puis appelle la fonction `execl()`. Pour chaque affirmation ci-dessous, indiquez si elle est correcte.

V F L'appel à `execl()` va échouer car le parent n'a pas le droit d'invoquer cette fonction.
V F Le processus fils va continuer à exécuter le code abandonné par son parent.
V F Le processus fils sera bloqué jusqu'à ce que son parent se termine.
V F Le processus parent sera bloqué jusqu'à ce que le fils se termine.

Question 3 Si on exécute le programme ci-dessous, combien de fois la lettre A sera-t-elle affichée ?

```
1  void main(void)
2  {
3     int n = 1;
4     while(n > 0)
5     {
6         fork();
7         n = n-1;
8         fork();
9     }
10    printf("A");
11 }
```

La lettre A sera affichée [ ] fois en tout.

Question 4 Dans la liste ci-dessous, indiquez quels appels système sont bloquants (au sens de l'ordonnancement).

V F `read`
V F `sleep`
V F `waitpid`
V F `exec`

2 Ordonnancement

Question 5 Répondez à chacune de ces questions par une seule phrase : Que fait un programme pendant une *IO-burst* ? Qu'est-ce qui marque la fin d'une *IO-burst* ?

Question 6 On s'intéresse dans cette question à une machine dotée de P processeurs. Pour chaque affirmation ci-dessous, indiquez si elle est correcte.

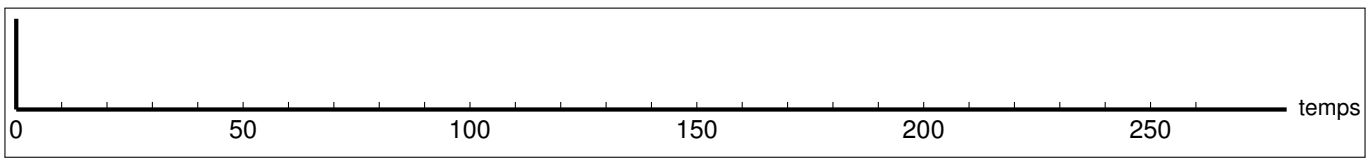
V F Le nombre maximum de processus en cours simultanément dans le système dépend de P.
V F Le nombre maximum de processus qui peuvent être simultanément BLOCKED dépend de P.
V F Le nombre maximum de processus qui peuvent être simultanément READY dépend de P.
V F Le nombre maximum de processus qui peuvent être simultanément RUNNING dépend de P.

Question 7 On suppose dans cette question un système monoprocesseur dont l'ordonnanceur gère trois niveaux de priorité. La priorité haute, moyenne, ou basse de chaque processus est fixée lorsque celui-ci est créé. Entre des niveaux distincts, l'ordonnancement est strictement préemptif. Parmi les processus d'un même niveau, l'ordonnancement est circulaire (Round Robin) avec un quantum de 20 millisecondes. La durée des commutations de contexte est supposée négligeable.

On s'intéresse aux tâches ci-dessous (les temps sont indiqués en millisecondes) :

	A	B	C	D	E	F	G
durée d'exécution	70	40	40	10	20	20	10
instant d'arrivée	0	0	9	9	9	19	19
priorité	moy	haute	basse	moy	haute	basse	moy

Sur une feuille de brouillon, dessinez un chronogramme indiquant la succession des tâches sur le processeur. Recopiez ensuite votre réponse au propre dans le cadre ci-dessous.



3 Mémoire virtuelle et caches

Question 8 On considère dans cette question une machine dont le processeur est connecté à une mémoire cache de latence 10 nanosecondes. Accéder à la mémoire principale entraîne une latence supplémentaire de 100 nanosecondes. En supposant que le cache satisfait 99% des requêtes, quel est le temps d'accès moyen à la mémoire observé par le processeur ?

Question 9 On s'intéresse dans cette question à un système de mémoire virtuelle paginée. Au cours de l'exécution d'un programme, on observe la MMU faire les traductions d'adresses ci-dessous (en notation hexadécimale) :
3E1F→7E1F, 5454→9454, 2A8D→6A8D, E127→2127, C8BA→08BA, 9762→5762, 72CF→A2CF.
Pour chaque question ci-dessous, répondez en kio, Mio, Gio (cf page 4) si les informations données permettent de répondre, ou bien répondez «inconnu» si les informations données ne permettent pas de répondre.

— La taille des pages virtuelles est ☐

— La taille des pages physiques est ☐

— La taille maximale pour un espace d'adressage virtuel est ☐

— La quantité de RAM disponible sur la machine est ☐

Question 10 Complétez ce paragraphe avec les termes manquants. Une *faute de page* est détectée par ☐ lorsque le programme essaye d'accéder à une de ses pages, mais que celle-ci n'est pas présente en mémoire. En réaction, ☐ saute alors vers le noyau. Celui-ci envoie au disque dur un ordre de lecture pour charger vers la mémoire les données manquantes. Le processus en question est marqué comme ☐ et l'ordonnanceur choisit un autre processus à exécuter. Plus tard, lorsque le disque dur lève une interruption pour signaler la fin de l'opération de lecture, le noyau met à jour ☐ du processus d'origine, et le marque à nouveau comme *prêt*.

4 Allocation dynamique

Question 11 Pour chaque affirmation ci-dessous, indiquez si elle est correcte.

☐V ☐F L'allocation statique consiste à déléguer au noyau le placement en mémoire des données.
☐V ☐F L'allocation statique consiste à déléguer au noyau le placement en mémoire des instructions.
☐V ☐F La pile d'exécution contient les instructions que le programme doit exécuter.
☐V ☐F La pile d'exécution contient les variables locales des fonctions du programe.

Question 12 L'outil `objdump` sert à examiner le contenu d'un fichier exécutable. En particulier, il permet de consulter le détail de :

☐V ☐F la section `.text`
☐V ☐F la section `.data`
☐V ☐F la section `.stack`
☐V ☐F la section `.heap`

Question 13 On s'intéresse dans cette question à un gestionnaire de mémoire dynamique appliquant la stratégie worst-fit. À l'instant initial, la *freelist* contient les blocs suivants, chaînés dans cet ordre :

A (9 kio) $\longrightarrow$ B (4 kio) $\longrightarrow$ C (20 kio) $\longrightarrow$ D (18 kio) $\longrightarrow$ E (7 kio)

On suppose que l'application demande successivement trois allocations pour des tableaux qu'on appellera X, Y, et Z, dont les tailles respectives sont 10 kio, 12 kio, et 9 kio.

Le tableau X sera alloué dans le bloc ☐

Le tableau Y sera alloué dans le bloc ☐

Le tableau Z sera alloué dans le bloc ☐

Annexe : quelques puissances de 2

$2^0 = 1$	$2^{16} = 65\,536$	$2^{32} = 4\,294\,967\,296$	$2^{48} = 281\,474\,976\,710\,656$
$2^1 = 2$	$2^{17} = 131\,072$	$2^{33} = 8\,589\,934\,592$	$2^{49} = 562\,949\,953\,421\,312$
$2^2 = 4$	$2^{18} = 262\,144$	$2^{34} = 17\,179\,869\,184$	$2^{50} = 1\,125\,899\,906\,842\,624$
$2^3 = 8$	$2^{19} = 524\,288$	$2^{35} = 34\,359\,738\,368$	$2^{51} = 2\,251\,799\,813\,685\,248$
$2^4 = 16$	$2^{20} = 1\,048\,576$	$2^{36} = 68\,719\,476\,736$	$2^{52} = 4\,503\,599\,627\,370\,496$
$2^5 = 32$	$2^{21} = 2\,097\,152$	$2^{37} = 137\,438\,953\,472$	$2^{53} = 9\,007\,199\,254\,740\,992$
$2^6 = 64$	$2^{22} = 4\,194\,304$	$2^{38} = 274\,877\,906\,944$	$2^{54} = 18\,014\,398\,509\,481\,984$
$2^7 = 128$	$2^{23} = 8\,388\,608$	$2^{39} = 549\,755\,813\,888$	$2^{55} = 36\,028\,797\,018\,963\,968$
$2^8 = 256$	$2^{24} = 16\,777\,216$	$2^{40} = 1\,099\,511\,627\,776$	$2^{56} = 72\,057\,594\,037\,927\,936$
$2^9 = 512$	$2^{25} = 33\,554\,432$	$2^{41} = 2\,199\,023\,255\,552$	$2^{57} = 144\,115\,188\,075\,855\,488$
$2^{10} = 1024$	$2^{26} = 67\,108\,864$	$2^{42} = 4\,398\,046\,511\,104$	$2^{58} = 288\,230\,376\,151\,711\,744$
$2^{11} = 2048$	$2^{27} = 134\,217\,728$	$2^{43} = 8\,796\,093\,022\,208$	$2^{59} = 576\,460\,752\,303\,423\,488$
$2^{12} = 4\,096$	$2^{28} = 268\,435\,456$	$2^{44} = 17\,592\,186\,044\,416$	$2^{60} = 1\,152\,921\,504\,606\,846\,976$
$2^{13} = 8\,192$	$2^{29} = 536\,870\,912$	$2^{45} = 35\,184\,372\,088\,832$	$2^{61} = 2\,305\,843\,009\,213\,693\,952$
$2^{14} = 16\,384$	$2^{30} = 1\,073\,741\,824$	$2^{46} = 70\,368\,744\,177\,664$	$2^{62} = 4\,611\,686\,018\,427\,387\,904$
$2^{15} = 32\,768$	$2^{31} = 2\,147\,483\,648$	$2^{47} = 140\,737\,488\,355\,328$	$2^{63} = 9\,223\,372\,036\,854\,775\,808$
			$2^{64} = 18\,446\,744\,073\,709\,551\,616$

On rappelle également que :

- 1 kio = 1024 octets,
- 1 Mio = 1024 Kio,
- 1 Gio = 1024 Mio,
- 1 Tio = 1024 Gio,
- etc. (avec dans l'ordre : Pio, Eio, Zio, Yio)

En cas de doute sur les unités de mesure, n'hésitez pas à demander des précisions.