

Introduction

- Les documents de cours sont autorisés
- Les réponses doivent être **claires** et **concises** : cela sera pris en compte lors de l'évaluation des réponses
- La durée de l'examen est 90 minutes

Q1) Donner cinq cas dans lesquels une interruption peut se produire ? (1,5pts)

- Interruption horloge
- Fin E/S
- Débordement,
- Division par zéro
- Fin de Quantum
- Etc.

Q2) Écrire un programme dans lequel un processus père crée 3 processus fils et attend leurs retours et les affiche à l'écran ? (1,5pts)

Il faut que les programmes créés soient des fils de même père. Il faut aussi que le père les attend et récupère leurs retours.

Q3) Donner la différence entre un appel système et une fonction utilisateur. Lequel entre eux s'exécute de façon plus rapide ? (1pt)

L'appel system fait partie des fonctions de la bibliothèque POSIX (c'est-à-dire des fonctions préfinies développé par les concepteurs du système d'exploitation), qui s'exécutent en mode système. Les fonction utilisateurs sont des programmes développés par les utilisateurs du système d'exploitation.

Q4) Un processeur ne traite que les interruptions et il a une capacité de lire ou écrire un mot mémoire de 4 octets en 10ns. Lors d'une interruption, il faut recopier 32 registres dans la pile du processus. Quel est le nombre maximal d'interruptions que ce processeur peut traiter en 2ms ? (2 pt)

Le temps d'une interruption est $2 \times 32 \times 10\text{ns} = 640\text{ ns}$. Le nombre d'interruptions qu'il peut traiter en 2ms est $2000000/640 = 3125$

Q5) Donnez deux raisons pour lesquelles un processus en cours d'exécution peut être interrompu (1pt).

- Déroutement
- Fin de quantum

Q6) Un disque comporte 100 pistes numérotées de 0 à 99. La dernière requête traitée concernait la piste 28 et une requête pour la piste 30 est en cours. La liste des nouvelles requêtes dans l'ordre d'arrivée est la suivante : 60,10,80,20,70,35,0,90

- Calculer le déplacement total de la tête pour les algorithmes suivants en illustrant votre réponse par un diagramme des déplacements effectués : (2 pt)
 - FIFO, PCTR, SCAN, LOOK
- Pourquoi est-il dangereux d'autoriser le traitement immédiat de nouvelles requêtes ? (1pt)

Cela crée le problème de famine

- Citez, parmi ces algorithmes, un algorithme qui souffre de cette insuffisance et expliquer pourquoi (1 pt)

- d) Le traitement immédiat de nouvelles requêtes risque d'entraîner une famine. PCTR est un algorithme qui risque d'entraîner une famine.

Pour les questions suivantes choisir la bonne réponse

Q7) Le tube ne permet pas de faire communiquer des processus de machines différentes. (1pt)

- A) Vrai
B) Faux

Q8) Lorsqu'un processus bloque un signal, le signal est délivré et le processus le gère en le jetant (1pt)

- A) Vrai
B) Faux

Q9) Voici le code (une partie) d'un programme qui permet d'initialiser un fichier via un segment de mémoire partagé :

1. fd= open(argv[1], O_RDWR);
2. stat (argv[1], &etat_fichier);
3. long taille_fichier = etat_fichier.st_size ;
4. projection = (char *) mmap(NULL, taille_fichier, PROT_READ | PROT_WRITE, MAP_SHARED, fd 0);
5. memset(projection, 'A'+1, taille_fichier);
6. close(fichier);
7. munmap((void *) projection, taille_fichier)

Si on suppose que la taille du fichier est de 10 octets, alors son contenu après l'exécution de ce programme sera : (1.5 pt)

- A) BBBBBBBBBB
B) BBBB
C) AAAAAAAAAA
D) AAAAA

Q10) Dans le système UNIX, les véritables appels système sont effectués à partir d'un programme utilisateur, d'une commande Shell, d'une procédure de la bibliothèque standard. Elles sont exécutées en : (1pt)

- A) Mode superviseur
B) Aucune de ces réponses
C) Mode utilisateur

Q11) Un lot est composé de 50 travaux, que pour simplifier, on suppose tous constitués de 3 phases indépendantes :

- lecture des cartes (20 secondes)
- calcul (15 secondes)
- impression des résultats (5 secondes).

Le temps mis pour passer d'un travail à un autre est négligeable.

Calculer le taux d'utilisation de l'unité centrale pour le calcul dans le cas où l'unité centrale gère les périphériques d'entrée-sortie (2 pt)

- A) 37,5 %
B) Aucune de ces réponses
C) 7,5%
D) 6,6%

Q12) Soit le programme suivant :

```
main(int argc, char ** argv)
{int child = fork();
int c = 5;
if(child == 0)
{c += 5;}
else
{child = fork();
c += 10;
if(child)c += 5;}}
```

Combien de copies différentes de la variable c existe-t-il ? **(1pt)**

- A) 1
- B) 2
- C) 3**
- D) 4
- E) aucune de ces réponses

Q13) On considère le morceau de programme suivant :

```
int main(void) {
int n = 0;
while(n < 3) {
int pid = fork();
if(pid == -1) exit(EXIT_FAILURE);
if(pid > 0) {
if(pid % 5 == 0) n++;
else
if(waitpid(pid, NULL, 0) == -1) exit(EXIT_FAILURE);
}
if(pid == 0) {
if(getpid() % 5 == 0) sleep(10);
exit(EXIT_SUCCESS);
}
}
for(int i = 0; i < 3; i++)
if(waitpid(-1, NULL, 0) == -1) exit(EXIT_FAILURE);
exit(EXIT_SUCCESS);
}
```

Dans le programme précédent, le processus principal crée : **(1pt)**

- A) exactement 3 fils
- B) des fils jusqu'à en avoir 3 dont le PID est un multiple de 5**
- C) des fils jusqu'à en avoir 5 dont le PID est un multiple de 3

Q14) Pour deux processus accédant à une variable partagée, l'algorithme de Peterson garantit : **(1pt)**

- A) exclusion mutuelle**
- B) progrès**
- C) attente limitée**
- D) aucune de ces réponses

Q15) Un processus Zombie est un processus : **(1pt)**

- A) qui a terminé son exécution et qui attend la prise en compte de cette fin par son père**

- B) qui a perdu son père
- C) qui a terminé son exécution en erreur
- D) aucune de ces réponses