Préparation DS SE.

1-Peut-on partager les registres entre différents processus ?

A un moment donné une seule instruction au plus est exécutée au nom d'un processus,càd un seul processus a le contrôle du processeur et de ses ressources.

Il est donc impossible de partager les registres du processeur entres différents processus.

2-Donner un exemple de file d'attente ?

File d'attente des processus prêts (attendant pour s'exécuter).

File d'attentes des périphériques.

3-Pourquoi on a besoin du PCB?

Dans les systèmes multiprorammés,un processus peut liberer le processeur et donner son contrôle a un autre processus,il faut donc sauvegarder les inforrmation du processus courant(PCB),pour pouvoir le relancer dans le même état.

4-Quel est l'objectif de l'ordonnancement ?

Exploiter au maximum le processeur,en respectant une politique d'ordonnancement .

Les objectifs d'un ordonnanceur d'un système multi-utilisateur sont,entre autres :

. S'assurer que chaque processus en attente d'exécution reçoive sa part de temps processeur.

. Minimiser le temps de réponse.

. Utiliser le processeur à 100%.

. Utilisation équilibrée des ressources.

. Prendre en compte des priorités.

. Être prédictibles.

5-Qu'offre l'UCT aux processus comme ressources ?

Les registres et le temps cpu(cycles) .

6-En quoi consiste l'exécution d'un processus ?

Alternance de calculs effectués par l'UCT et de requêtes d'entrées/Sorties.

7-Citer et décrire les différentes parties composant un processus en mémoire ?

Section texte : contient le code du programme, plus exactement les instructions en langage machine, en lecture seule, Du fait de son immuabilité, c'est une zone mémoire de taille fixe

La section de **données** (*data*) et la section **bss** stockent les variables globales et statiques du programme. Si ces données sont initialisées, elles sont enregistrées dans la section *data*, tandis que les autres sont dans la section bss. Ce sont également des zones mémoires de taille fixe.

Le **tas** (*heap*) est, quant à lui, manipulable par le programmeur. C'est la zone dans laquelle sont écrites les zones mémoires allouées dynamiquement (*malloc()* ou *calloc()*). Tout comme la pile, cette zone mémoire n'a pas de taille fixe. Elle augmente et diminue en fonction des demandes du programmeur, qui peut réserver ou supprimer des blocs via des algorithmes d'allocation ou de libération pour une utilisation future. Plus la taille du tas augmente, plus les adresses mémoires augmentent, et s'approchent des adresses mémoires de la pile. La taille des variables dans le tas n'est pas limitée (sauf limite physique de la mémoire), contrairement à la pile.

La **pile** (*stack*) possède également une taille variable, mais plus sa taille augmente, plus les adresses mémoires diminuent, en s'approchant du haut du tas. C'est ici qu'on retrouve les variables locales des fonctions ainsi que le cadre de pile (*stack frame*) de ces fonctions. La *stack frame* d'une fonction est une zone mémoire, dans la pile, dans laquelle toutes les informations, nécessaires à l'appel de cette fonction, sont stockées. S'y trouvent également les variables locales de la fonction.

*Les **registres** sont des emplacements mémoire qui sont à l'intérieur du processeur. Or dans un ordinateur, les emplacements mémoire les plus proches du processeur sont ceux à qui il est le plus rapide d'accéder, mais également les plus chers. Ainsi, plus on s'éloigne du processeur, plus les accès sont longs, mais les coûts sont faibles. Les registres sont les emplacements mémoire les plus proches (puisqu'ils sont internes au processeur), c'est alors la mémoire la plus rapide de l'ordinateur*

8-Préciser la partie dynamique et celle statique d'un processus en mémoire ?

Statique :Data ,section texte .

Dynamique :Heap,pile(stack).

9-En quel niveau la priorité des processus est déterminée ?

Au niveau de l'ordonnanceur.

10-Décrire l'opération d'arrêt/reprise d'un processus ? (Changement de contexte)

Sauvegarder l'état du processus courant en PCB,et charger l'état du processus élu.

Un appel à un ordonnanceur en mode noyau est fait a fin de pouvoir manipuler les PCB.

Le stockage de l'état d'un processus ou d'un thread , afin qu'il puisse être restauré et reprendre l' exécution ultérieurement. Cela permet à plusieurs processus de partager une seule unité centrale (CPU) et constitue une caractéristique essentielle d'un système d'exploitation multitâche .

Les commutateurs de contexte sont généralement gourmands en calculs et une grande partie de la conception des systèmes d'exploitation consiste à optimiser l'utilisation des commutateurs de contexte. Le passage d'un processus à un autre nécessite un certain temps pour faire l'administration - sauvegarde et chargement des registres et des cartes mémoire, mise à jour de diverses tables et listes,

11-13-Comment étendre la mémoire principale ?

La mémoire virtuelle (swap).