Tous les projets seront réalisés en Python,
ils seront notés avec des notes de 0-10

1. Émuler le fonctionnement de 2 stratégies d'ordonnancement en utilisant des threads. L'interface permettra de saisir le nombre de rafales en concurrence et de generer des valeurs aléatoires pour les durées et les moments d'arrivée de chaque rafale. Le programme simulera l'execution de ces rafales en appliquant deux algorithmes d'ordonnancement pour comparer les temps moyens de traitement et les temps moyens d'attente et affichera graphiquement le déroulement des processus.

   Les variantes pour la comparaison des strategies d'ordonnancement sont : SJFNP/SJFP, SJFNP/FCFS, SJFNP/RR, SJFP/FCFS, SJFP/RR, FCFS/RR

   2 étudiants / variante

2. Réaliser un système « client/serveur » avec 2 machines, permettant de se servir du client pour lancer des tris de fichiers sur le serveur. Les fichiers a trier contiennent des mots qu'il faut ordonner alphabetiquement. La liste triee doit ensuite etre affichee sur l'ecran du client.

   Réalisation : 2 étudiants (client / serveur)

3. Réaliser un système « client/serveur » avec 2 machines, permettant de se servir du client pour lancer des tris de fichiers sur le serveur. Les fichiers a trier contiennent des images qu'il faut ordonner sur la base de leur taille. Les images triees sernt affichees sur ll'ecran du client.

   Réalisation : 2 étudiants (client / serveur)

4. Réaliser un interpréteur de commandes UNIX (ou Linux) sur une machine équipée de Windows. L'interpreteur devra reconnaître au moins 6 commandes. Expliquer quelles commandes n'ont pas pu être réalisées et pourquoi.

   Réalisation : 1 étudiant

5. Une salle de spectacle a 500 places. Les demandes de reservation proviennent simultanement et de maniere aleatoire depuis 9 caisses differentes. Ecrire un programme de reservation pour le processus serveur et le module d'edition des billets pour les processus clients, Des messages doivent etre prevus lorsqu'une demande ne peut pas etre satisfaite en raison de l'occupation de la place demandee par un autre client..

   Realisation : 2 etudiants

6. Réaliser un programme pour l'affichage des processus en cours (exemple : PROCEXP.EXE). Le programme devra afficher les noms, l'utilisation de l'UC, la mémoire occupée ainsi que la priorité du processus en cours. On doit pouvoir arrêter un processus et changer sa priorité

   Réalisation : 1 etudiant+

   7. Écrire un programme qui lance 3 threads ayant tous les 3 comme objectif d'écrire ou de lire dans le même fichier. Les threads préparent des chaînes de caractères de longueur aléatoire dans un tampon et transfèrent le contenu du tampon vers le fichier commun lorsque les chaînes ont ete complètees.

   La demande d'écriture par un des threads doit être signalée aux deux autres qui doivent s'interrompre pour permettre la modification du fichier.
   Le programme affichera les chaînes de caractères ajoutées par chaque thread au fichier, précédées du nom du thread qui a composé la chaine.

   Réalisation : 2 étudiants

8. Simuler un système d'interruptions matérielles. La table des interruptions comprendra 4 « colonnes » :
    1. Une colonne avec les codes des interruptions (1 caractère alphabétique), pour representer l'origine de l'interruption
    2. Une colonne avec les noms des procédures à exécuter dans chaque cas
    3. Une colonne avec les durées d'exécution (secondes) de la procedure de traitement de l'interruption
    4. Une colonne avec les priorités (valeurs entières entre 1 et 16). de chaque procedure

    La table sera renseignee manuellement. Lorsque la table a ete renseignee entierement l'utilisateur pourra demarrer la procedure de simulation qui lancera de maniere aleatoire des demandes d'interruption qui seront simulees par des threads dont la duree d'execution correspond au code de l'interruption.

    Les threads executeront tous la meme procedure ::

    - envoyer un message avec le nom de la procédure et le moment de son déclenchement,
    - exécuter une boucle sans objet d'une durée égale (en secondes) à la valeur spécifiée dans la table
    - à la sortie de la boucle, envoyer un message avec le nom de la procédure et le moment de la fin.

Si pendant l'execution de la procedure pour une interruption survient une autre interruption, il peut se passer 2 cas :

- la nouvelle procedure est prioritaire : om iterromp la procedure en cours et on execute la nouvelle ; apres la fin de cette derniere on reprend la procedure interrompue
- la nouvelle procedure est de priorite inferieure a celle en cours : on la place dans un queue d'attente ou les procedures en attente seront ordonnees selon leur priorite

Une présentation graphique de l'enchaînement des procédures est demandée.

Réalisation : 3 étudiants (interface/gestion des interruptions)

9. Réaliser un programme de communication entre 10 processus par boites aux lettres. Le nombre de boites aux lettres est limité a 5.

   Chaque processus peut envoyer et/ou recevoir des messages de la part de n'importe quel autre processus mais si toutes les boites aux lettres sont occupées, la demande de l'envoyeur doit être placée dans une queue d'attente. Le rôle de gestionnaire des messages (identification de l'auteur et du destinataire, transfert du message et liberation de la boite aux lettres) est dévolue a un thread 'superviseur' qui explore régulièrement les boites aux lettres, indépendamment du fonctionnement des threads 'ouvriers' (qui produisent les messages

   Réalisation : 2 étudiants

10. Réaliser un système client-serveur destiné a proposer un choix de produits a acheter. Les produits, des équipements électroniques grand public (informatique, téléphones, audio...), sont proposés par plusieurs magasins avec des marques et des prix différents. Dans l'application client on pourra indiquer des critères de choix (type d'appareil, prix maximum, marque). L'application serveur devra sélectionner dans les listes des produits disponibles ceux qui répondent aux critères et envoyer leur liste a l'application cliente, qui se chargera de son affichage. Les recherches de produits dans l'application serveur feront l'objet de fils (threads).

    Realisation : 3 étudiants

11. Réaliser un programme pour illustrer la solution du problème des philosophes chinois en utilisant les sémaphores. Interface graphique exigee

    Réalisation : 2 étudiants

12. Réaliser un programme pour illustrer la solution du problème des philosophes chinois en utilisant un moniteur Python. Interface graphique exigee
Réalisation : 2 étudiants

13. Réaliser un programme pour la simulation de la gestion des demandes de mémoire des processus prêts a être exécutés. Les processus prêts sont introduits dans une queue de demandes de mémoire, avec leur ID, le moment de leur arrivée et la taille de la zone de mémoire sollicitée. S'il reste de la place disponible, un processus qui peut être pris en charge (sa demande en mémoire est inférieure a la place disponible) est déplacé depuis la queue des processus en attente dans celle des processus en cours d'exécution. Une horloge est prévue pour arrêter de manière aléatoire les processus en cours d'exécution, ce qui entraîne la libération de la mémoire qu'il occupait.

Le nombre total de processus a traiter ainsi que la taille de la mémoire totale dont dispose le système seront choisis par l'utilisateur tandis que la taille de la mémoire demandée par chaque processus sera générée automatiquement par un generateur de nombres aléatoires. Cette taille ne devra en aucun cas dépasser la taille de la mémoire du système.

Une présentation graphique simulant le fonctionnement est exigée.

Réalisation : 3 étudiants