Questions de cours :

1. Citer au moins 3 rôles d'un OS ?
2. Quelle est la déférence entre Instruction et Macro-instruction ?
3. Que veut dire une séquence de tâches valide ?
4. Pour qu'un ordonnanceur soit très performant, il doit maximiser ..... et également minimiser..... (Compléter les critères).
5. Quel type d'algorithme d'ordonnancement qui souffre de la famine ?
6. La surveillance d'une centrale nucléaire nécessite un système temps réel souple : **Oui/Non.**
7. Le Memory Management Unit (MMU) traduit une adresse virtuelle en adresse logique : **Oui/Non.**
8. Les sémaphores privés sont toujours initialisés par Si = -1 : **Oui/Non.**
9. Citer au moins 3 rôles d'un gestionnaire de mémoire ?
10. Quelle est la déférence entre *Shell* et *Kernel* dans un OS ?
11. Un sémaphore **mutex** est initialisé par S = 1 dans un système qui contient deux ressources critiques (Oui/Non avec justification)**.**

Réponses aux questions :

1. **Rôles d'un OS** : 1- Gestion du processeur, 2- Gestion de la mémoire vive, 3- Gestion des entrées/sorties, 4- Gestion de l'exécution des applications, 5- Gestion des fichiers, 6- Gestion des informations.
2. **Instruction** : une opération élémentaire d'un processeur. **Macro-instruction** : instruction complexe, définissant des opérations composées à partir des instructions du répertoire de base d'un ordinateur.
3. **Une séquence de tâches valide** : si toutes les tâches respectent leur Contrainte Temporelle.
4. Pour qu'un ordonnanceur soit très performant, il doit maximiser le pourcentage d'utilisation du processeur et le débit. Également, il doit minimiser le temps de réponse, le temps de rotation et le temps d'attente.
5. L'ordonnancement basé sur les priorités (**PRIO**) souffre de **la famine**, car les processus moins prioritaires n'arrivent pas à s'exécuter pendant un temps indéterminé.
6. **Non**, La surveillance d'une centrale nucléaire nécessite un système temps réel dur.
7. **Non**, Le Memory Management Unit (MMU) traduit une adresse virtuelle (logique) en adresse physique.
8. **Non**, Les sémaphores privés sont toujours initialisés par Si = 0.
9. Au moins 3 rôles d'un gestionnaire de mémoire :
   - création d'un processus.
   - activation/désactivation d'un processus.
   - suppression d'un processus.
   - partage la mémoire disponible entre les processus (protection).
   - cartographie la mémoire.
   - alloue/dés-alloue de la mémoire dynamiquement pour les besoin d'un processus.
   - assure la cohérence de la mémoire.
   - optimise l'utilisation de la mémoire.

10. **Shell** : assure la communication avec le système d'exploitation par l'intermédiaire d'un langage de commandes, afin de permettre à l'utilisateur de piloter les périphériques.
    Kernel : représentant les fonctions fondamentales du système d'exploitation telles que la gestion de la mémoire, des processus, des fichiers, des entrées-sorties principales, et des fonctionnalités de communication.
11. **Non**, Le sémaphore de ce système est initialisé par S = 2, parce qu'il possède 2 ressources critiques.

Exercice N°1 (Gestion des tâches) :

Soit un système temps réel avec 3 tâches ayant les paramètres suivants :

Tâche	Temps de réveil (r)	Durée d'exécution (C)	Délai critique (D)	Période (P)
T1	0	2	4	6
T2	0	3	8	8
T3	0	1	3	4

Partie I : Dans cette partie, on s'intéresse uniquement à la zone grise de tableau → Ti(ri, Ci).

1- Tracer les chronogrammes de **SJF** sans préemption et **RR** (*Q* = 2) ?
2- Pour chaque cas, calculer le temps moyen d'attente et le temps moyen de rotation ?
3- Déduire le meilleur ordonnanceur ? Justifier ta réponse ?

Partie II : Maintenant, on considère tout le tableau → Ti(ri, Ci, Di, Pi).

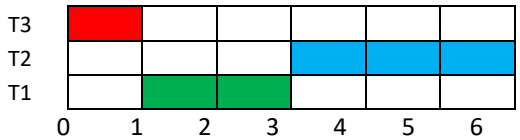
4- Calculer la période d'étude de ce système.
5- Ce système est-il ordonnançable avec **RM** ? Vérifier avec le diagramme de Gantt ?
6- Même question pour **EDF** ?

Correction exercice N° 01 :

Partie I : Ti(ri, Ci).

Tâche	Temps de réveil (r)	Durée d'exécution (C)
T1	0	2
T2	0	3
T3	0	1

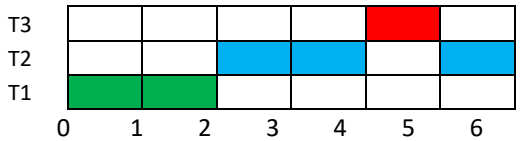
Le chronogramme de **SJF** sans préemption :



Temps moyen d'attente = (1 + 3 + 0) / 3 = 1,33

Temps moyen de rotation = (3 + 6 + 1) / 3 = 3,33

Le chronogramme de **RR** (*Q* = 2) :



Temps moyen d'attente = (0 + 3 + 4) / 3 = 2,33

Temps moyen de rotation = (2 + 6 + 5) / 3 = 4,33

3- Le meilleur ordonnanceur est le **SJF** sans préemption, car le temps moyen d'attente et le temps moyen de rotation correspondants sont les plus petits.

Partie II : Ti(ri, Ci, Di, Pi).

Tâche	Temps de réveil (r)	Durée d'exécution (C)	Délai critique (D)	Période (P)
T1	0	2	4	6
T2	0	3	8	8
T3	0	1	3	4

4- La période d'étude = [0, PPCM(P1, P2, P3)] = [0, PPCM(6, 8, 4)] = [0, 24].

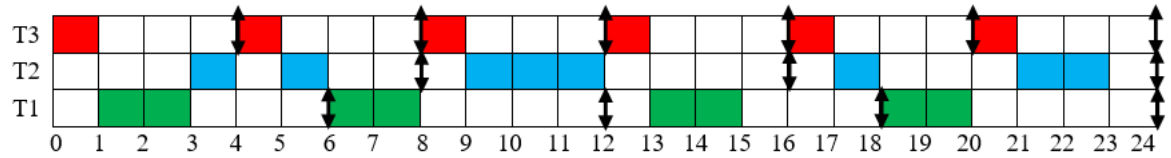
5- Test d'ordonnançabilité avec **RM** :

$$\sum_{i=1}^{n} \frac{C_i}{P_i} \leq n\left(2^{1/n} - 1\right) \rightarrow \text{Condition suffisante.}$$

$$\sum_{i=1}^{n} \frac{C_i}{P_i} = \frac{2}{6} + \frac{3}{8} + \frac{1}{4} = 0{,}958$$

$$n\left(2^{1/n} - 1\right) = 3\left(2^{1/3} - 1\right) = 0.779$$

Donc, la condition suffisante n'est pas vérifiée, on trace le diagramme de Gantt pour voir s'il y a des éventuelles fautes temporelles. On a T3 → T1 → T2.



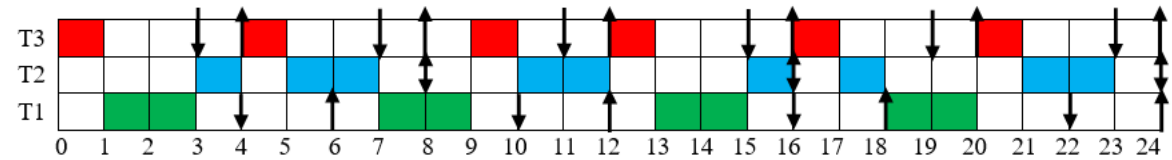
Il y a une faute temporelle dans la première période de la tâche T2, elle s'est exécutée pendant 2 UT seulement tandis que C2 = 3 UT. Donc, ce système n'est pas ordonnançable selon **RM**.

6- Même question pour **EDF** :

$$\sum_{i=1}^{n} \frac{C_i}{P_i} = 0.958 \leq 1 \text{ (La condition nécessaire est vérifiée)}$$

$$\sum_{i=1}^{n} \frac{C_i}{D_i} = 1.2 > 1 \text{ (La condition suffisante n'est pas vérifiée)}$$

On trace le diagramme de Gantt :



Exercice N°2 (Gestion des tâches) :

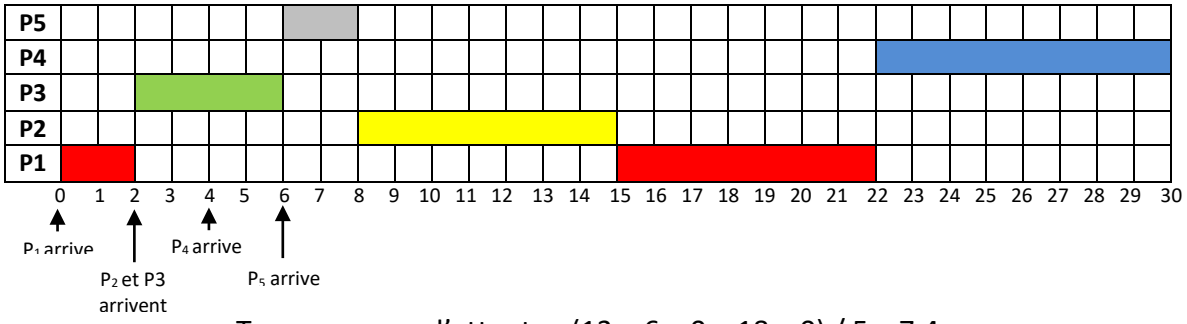
On considère les processus suivants, définis par leur durée (réelle ou estimée), leur date d'arrivée et leur priorité :

Tâche (P_i)	Date d'arrivée (r)	Durée d'exécution (C)	Priorité
P1	0	9	3
P2	2	7	3
P3	2	4	1
P4	4	8	2
P5	6	2	4

1- Dessinez un diagramme de Gantt correspondant au résultat d'un ordonnancement **SJF préemptif** et indiquez le temps d'attente moyen.
2- Dessinez un diagramme de Gantt correspondant au résultat d'un ordonnancement **Round Robin** avec un quantum de temps fixé à 2 et indiquez le temps d'attente moyen.
3- Quel est le meilleur algorithme ?

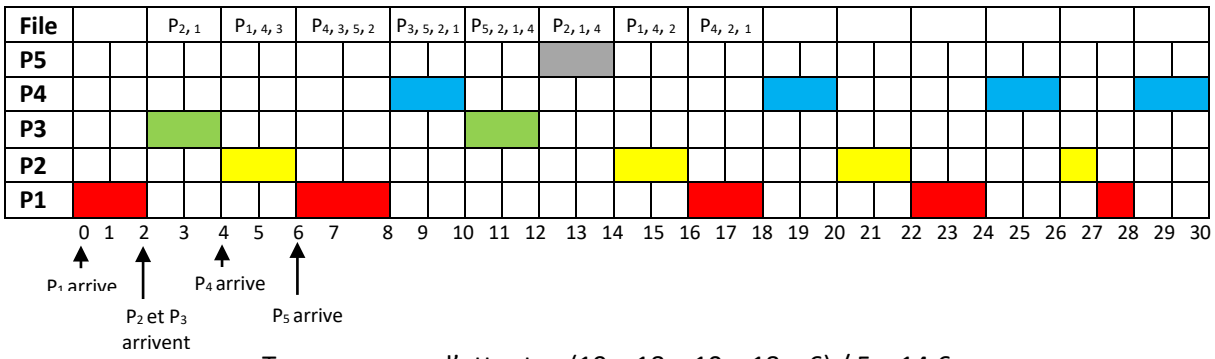
Correction Exercice N°2 :

1- SJF préemptif :



Temps moyen d'attente : (13 + 6 + 0 + 18 + 0) / 5 = 7.4

2- RR (Q=2) : dans ce cas on ignore la priorité



Temps moyen d'attente : (19 + 18 + 10 + 18 + 6) / 5 = 14.6

3- dans les deux cas, c'est le SJF préemptif le meilleur.

Exercice N°3 (Pagination) :

On suppose un espace d'adresses logiques de huit pages de 1024 octets chacune, représenté dans une mémoire physique de 32 cadres de pages. Combien de bits comporte l'adresse logique ? L'adresse physique ? Expliquez.

Correction Exercice N°3 :

1024 = 2^{10} donc 10 bits pour l'offset dans la page, 8 = 2^3 donc 3 bits pour le numéro de page et 32 = 2^5 donc 5 bits pour les cadres de pages.

- Adresse logique = 13 bits.
- Adresse physique = 15 bits.

Exercice N°4 (Pagination) :

On suppose un système de 2048 Ko de mémoire haute organisé avec des pages de 8Ko. Décrivez le système d'adressage logique. Quelle est la taille maximum de la table des pages ? Expliquez.

Correction Exercice N°4 :

2048 Ko / 8 Ko = 2^{11} / 2^3 = 2^8 pages. Pour adresser une page, il faut 1 octet.
On a donc une table des pages qui peut faire 2^8 octets = 256 o.

Exercice N°5 (Gestion de la mémoire) :

On se place dans un système de mémoire de 1700 Ko de mémoire haute (c'est-à-dire au-delà de la partie utilisée par l'OS) répartie en cinq partitions de 100Ko, 500Ko, 400Ko, 300Ko et 600Ko (dans cet ordre).
On suppose que le système d'exploitation doit allouer des processus de taille 212Ko, 417Ko, 112Ko et 426Ko (dans cet ordre). Pour chacun des algorithmes suivants, donnez l'allocation obtenue et le taux de fragmentation :

- First-Fit (prochain bloc libre)
- Best-FIt (plus petit bloc libre)
- Worst-Fit (plus grand bloc libre)

Quel algorithme utilise le plus efficacement la mémoire sur cet exemple ?
Calculer le Taux de fragmentation (espace libre/espace total) pour chaque cas ?

Correction Exercice N°5 :

On a les processus P1(212Ko), P2(417K), P3(312K) et P4(426K) :

1) First fit :

État initial	100K	500K	400K	300K	600K
Placement de P1(212K)	100K	P1+288K	400K	300K	600K
Placement de P2(417K)	100K	P1+288K	400K	300K	P2+183K
Placement de P3(312K)	100K	P1+288K	P3+88K	300K	P2+183K
Placement de P4(426K)	Impossible				

2) Best fit :

État initial	100K	500K	400K	300K	600K
Placement de P1(212K)	100K	500K	400K	P1+88K	600K
Placement de P2(417K)	100K	P2+83K	400K	P1+88K	600K
Placement de P3(312K)	100K	P2+83K	P3+88K	P1+88K	600K
Placement de P4(426K)	100K	P2+83K	P3+88K	P1+88K	P4+174K

Taux de fragmentation = (100+83+88+88+174)/(100+500+400+300+600) = 0.28

3) Worst fit :

État initial	100K	500K	400K	300K	600K
Placement de P1(212K)	100K	500K	400K	300K	P1+388K
Placement de P2(417K)	100K	P2+83K	400K	300K	P1+388K
Placement de P3(312K)	100K	P2+83K	P3+88K	300K	P1+388K
Placement de P4(426K)	Impossible				

L'algorithme qui utilise le plus efficacement la mémoire est : **Best Fit**.

Exercice N° 6 (Gestion des tâches et de mémoire) :

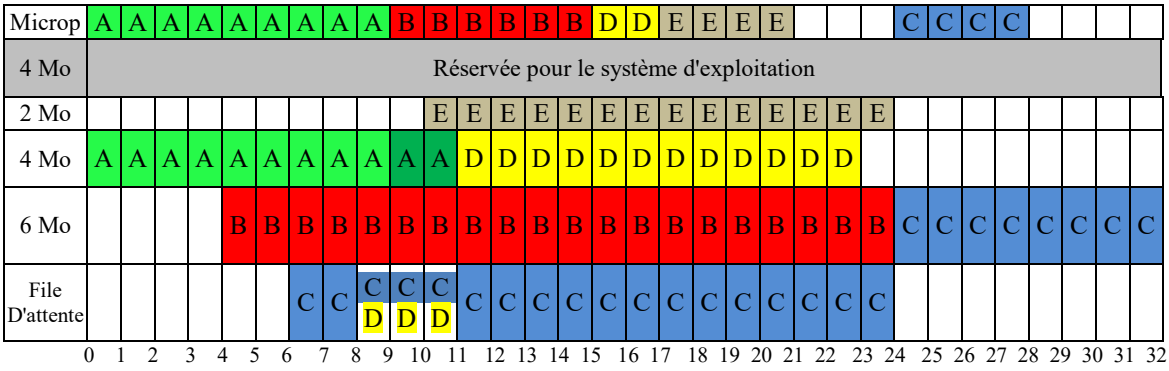
On considère un système disposant de 16 MB de mémoire physique, avec la partie résidente du système sur 4 MB. On suppose que l'exécution de chaque processus se compose d'un temps processeur suivi d'une demande d'E/S. On suppose de plus que les processus n'attendent pas pour leur E/S (par exemple, ils utilisent tous un périphérique différent). Le tableau suivant donne un exemple de séquences de tâches pour le système :

Instant *t*	Processus	Taille	Temps CPU	Durée E/S
0	A	3 MB	9 ms	2 ms
4	B	5 MB	6 ms	9 ms
6	C	5 MB	4 ms	4 ms
8	D	4 MB	2 ms	6 ms
10	E	1 MB	4 ms	3 ms

On suppose qu'un processus chargé y séjournera jusqu'à la fin de son exécution.
Donnez les états d'occupation de la mémoire aux différentes étapes de traitement de ces processus, sous les hypothèses suivantes :

- Partitions fixes de tailles 6 MB, 4 MB, 2 MB et 4 MB (pour le système) ;
- Le mode d'allocation des trous utilise l'algorithme meilleur ajustement (**Best Fit**);
- Le répartiteur de haut niveau (chargement de processus en mémoire) fonctionne selon **FCFS** ;
- Le répartiteur de bas niveau (vers microprocesseur) fonctionne selon **SJF sans préemption**.

Correction Exercice N°6 :



A t =0, A arrive et est chargé dans la partition de 4 Mo, il y séjournera jusqu'à la fin de son exécution (t=9+2).
A t = 4, B arrive et est chargé dans la partition de 6Mo. Il y séjournera jusqu'à la fin de son exécution.
A t= 6, C arrive et ne peut être chargé en mémoire. Il est mis en attente dans la file (du répartiteur de haut niveau). La file contient (C).
A t= 8, D arrive et ne peut être chargé en mémoire. Il est mis en attente dans la file (du répartiteur de haut niveau). La file contient (C D)
A t= 9, A libère le processeur. L'exécution de B est entamée. B libérera le processeur à t= 9+6 et la partition à t = 9+6+9.
A t=10, E arrive et est chargé dans la partition de 2 Mo. Il y séjournera jusqu'à la fin de son exécution.
A t= 11, l'exécution de A se termine, la partition de 4 Mo devient libre. D est chargé dans cette partition et y séjournera jusqu'à la fin de son exécution. La file contient (C).
A t= 15, B libère le processeur qui est alloué au plus court d'abord c-à-d D. D libérera le processeur à t=15+2 et la partition à t=15+2+6.
A t= 17, D libère le processeur qui est alloué au plus court d'abord c-à-d E. E libérera le processeur à t=17+4 et la partition à t=17+4+3.
A t=21, E libère le processeur mais tous les processus en mémoire sont en attente de fin d'E/S.
A t = 23, D libère la partition de 4 Mo.
A t = 24, B et E libèrent les partitions de 6 Mo et 2 Mo, C'est chargé dans la partition de 6 Mo.
A t = 28, C libère le processeur.
A t = 32, C libère la partition de 6 Mo.

Exercice N° 7 : (Concours Doctorat Université Oum Elbouagui 2019)

Les mesures prélevées sur un système donné ont montré que la durée moyenne d'exécution d'une tâche était de **T** avant que ne se produise sur les E/S. Un changement de tâche a besoin de délai de **S** que l'on peut considérer comme une perte de temps. Pour un algorithme d'ordonnancement de type tourniquet avec quantum de **Q**.

Donner une formule pour exprimer l'efficacité du processeur (facteur de son utilisation) pour chacun de ces cas suivants :

1) Q>T
2) S<Q<T
3) Q=S
4) Q proche de 0.

Correction Exercice N°7 :

T : Durée moyenne de la tâche.

S : Retard de changement (durant les commutations)

Q : quantum de Tourniquet

Efficacité de processeur (eff_p) = temps utile / temps total

Cas 1 (Q > T) :

Dans ce cas y a pas des commutations de tourniquet => eff_p = $\frac{T}{T+S}$

Cas 2 (S < Q < T) :

Pour exécuter T, on a besoin de T/Q commutations (interruptions) de Tourniquet et à chaque changement y aura un rtard S => le total de retard = S. $\frac{T}{Q}$

Donc eff_p = $\frac{T}{T+S.\frac{T}{Q}} = \frac{Q}{Q+s}$

Cas 3 (Q = S) :

eff_p = $\frac{Q}{Q+s} + \frac{Q}{Q+Q} = 1/2$ ça veut dire 50% d'efficacité de processeur.

Cas 4 (Q plus proche de 0) :

eff_p = $\frac{0}{0+s} = 0$ ça signifie que le processeur ne fonctionne pas (y a une perte de temps pour rien)

Exercice N° 8 (Pagination) :

Dans un système paginé, les pages font 256 octet en mémoire et on autorise chaque processus à utiliser au plus 4 cadres de la mémoire centrale. On considère cette table des pages avec la tâche P1 :

Page	0	1	2	3	4	5	6	7
Cadre	011	001	000	010	100	111	101	110
Présente	**oui**	non	**oui**	non	non	non	**oui**	non

1) De combien de mémoire vive dispose ce système ?

2) Calculez les adresses réelles correspondant aux adresses virtuelles suivantes :

240, 546, 1578, 2072

3) Que se passe-t-il si P1 génère l'adresse virtuelle 770 ?

4) On considère l'adresse virtuelle suivante : 0000 0000 0000 0111.

Sachant que les 4 bits de poids fort désigne le numéro de page et que 12 bits suivants représentent le déplacement dans la page, donnez l'adresse physique correspondant à cette adresse (exprimée en binaire).

Correction Exercice N°8 :

1) Comme les cadres sont numérotés sur 3 bits, il y a 2^3= 8 cadres. Taille d'un cadre = taille d'une page donc la mémoire physique comporte 8*256 = 2Ko.

2) La conversion d'une adresse virtuelle en adresse réelle est réalisée de la façon suivante :

a- Calcul du numéro de la page et du déplacement dans la page.
b- Recherche dans la table de pages de l'entrée qui correspond à la page de façon à en déduire le numéro du cadre.
c- L'adresse physique (réelle) est obtenue en ajoutant le déplacement à l'adresse physique de début du cadre.

Voici le détail des calculs pour les adresses demandées :

- 240 = 0*256 + 240 → page = 0 et déplacement = 240. D'après la table des pages, cadre= 3. Adresse physique = 3*256 + 240 = 1008.
- 546 = 2*256 + 34 → page = 2 et déplacement = 34. D'après la table des pages, cadre= 0. Adresse physique = 0*256+ 34 = 34.
- 1578 = 6*256 + 42 → page = 6 et déplacement = 42. D'après la table des pages, cadre= 5. Adresse physique = 5*256 + 42 = 1322.
- 2072 est en dehors de l'espace d'adressage virtuel du processus (2048 mots).

3) 770 = 3*256 + 2. Il s'agit d'une adresse située dans la page 3. D'après la table des pages, cette page n'est pas présente en mémoire (un défaut de page).

4) D'après la table de pages, cette page se trouve dans le cadre 011. L'adresse physique s'obtient donc simplement en substituant aux 4 bits de poids fort de l'adresse virtuelle les 3 bits du numéro de cadre : 0011 0000 0000 0111.

Exercice N°9 : (Sémaphores, Inversion de priorité, Producteur/Consommateur)

Dans cet exercice on souhaite montrer l'impact d'une inversion de priorité sur l'ordonnancement d'un jeu de tâche. Soient trois tâches définies par les paramètres suivants :

	r_i	C_i	P_i
T1	0	2	6
T2	0	2	8
T3	0	4	12

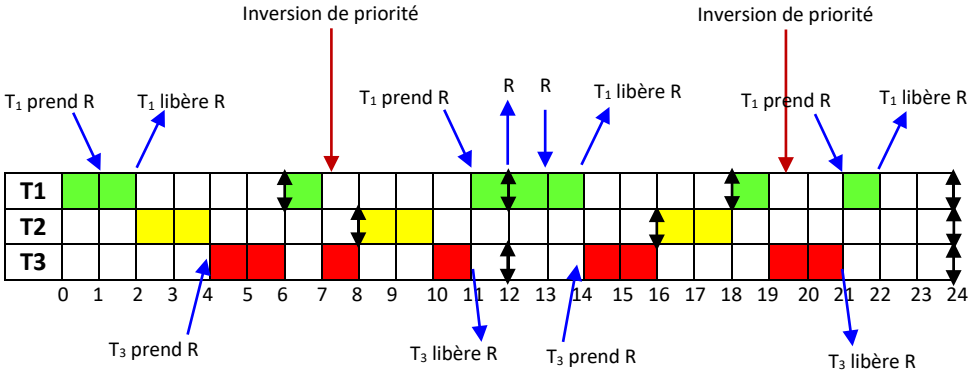
Les délais critiques sont égaux aux périodes ($D_i = P_i$), On utilise **RM** pour ordonnancer les tâches (mode préemptif).
La tâche T1 produit des données alors que T3 les consomme. Ces données sont stockées dans une mémoire commune que T1 et T3 se partagent cette ressource (R) par l'accès en exclusion mutuelle. T1 accède à la ressource durant la deuxième unité de temps de sa capacité. T3 accède à la ressource durant la totalité de sa capacité.

1. Tracer sur la période d'étude le diagramme de Gantt généré par l'ordonnanceur RM.
2. Indiquer les moments d'accès exclusif à la ressource et les moments de sa libération.
3. Est-ce qu'il y a une inversion de priorité ? Si oui, indiquer où.

Correction Exercice N°9 :

Algorithme de RM : (T1 ↦ T2 ↦ T3)

Période d'étude =[0 – PPCM(6, 8, 12)] = [0 - 24].



Exercice N° 10 (Gestion des tâches) :

Soit un système temps réel avec 3 tâches ayant les paramètres suivants :

Tâche	Temps de réveil (r)	Durée d'exécution (C)	Délai critique (D)	Période (P)
T1	0	2	4	6
T2	0	**X**	8	8
T3	0	1	3	4

(X est un nombre entier positive)

1. Déterminez sous quelle condition sur **X** ce système est ordonnançable selon RM puis EDF.
2. Nous souhaitons maintenant introduire **m** copies de la tâche T2 en parallèle :
   Déterminez la condition obtenue sur **X**, en fonction de **m**, pour que ce nouveau système soit ordonnançable selon EDF ?
3. Pour **X**=3, tracer le diagramme de Gantt avec l'ordonnanceur RR (**Q** = 2) ? Calculer le temps moyen d'attente et le temps moyen de rotation ?

Correction exercice N° 10 :

1- **Ordonnançabilité selon RM :**

Condition suffisante : $\sum_{i=1}^{n} \frac{C_i}{P_i} \leq n\left(2^{1/n} - 1\right) \Leftrightarrow \frac{2}{6} + \frac{X}{8} + \frac{1}{4} \leq 3(2^{\frac{1}{3}} - 1)$

$$\Leftrightarrow \frac{3X+14}{24} \leq 0.779$$

$$\Leftrightarrow X \leq 1.56$$

Ordonnançabilité selon EDF :

A. Condition nécessaire : $\sum_{i=1}^{n} \frac{C_i}{P_i} \leq 1 \Leftrightarrow \frac{3X+14}{24} \leq 1$

$$\Leftrightarrow X \leq 3.33$$

B. Condition suffisante : $\sum_{i=1}^{n} \frac{C_i}{D_i} \leq 1 \Leftrightarrow \frac{3X+20}{24} \leq 1$

$$\Leftrightarrow X \leq 1.33$$

2- Ordonnançabilité selon EDF avec **m** copies de T2 :

A. Condition nécessaire : $\sum_{i=1}^{n+m-1} \frac{C_i}{P_i} \leq 1 \Leftrightarrow \frac{2}{6} + \frac{mX}{8} + \frac{1}{4} \leq 1$

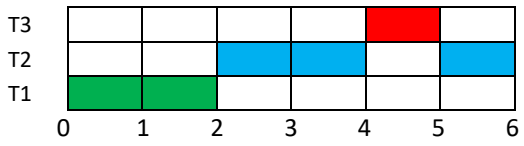
$$\Leftrightarrow X \leq \frac{3.33}{m}$$

B. Condition suffisante : $\sum_{i=1}^{n+m-1} \frac{C_i}{D_i} \leq 1 \Leftrightarrow \frac{3mX+20}{24} \leq 1$

$$\Leftrightarrow X \leq \frac{1.33}{m}$$

3- Le diagramme de Gantt pour **RR** (Q = 2 et X=3) :

Tâche	Temps de réveil (r)	Durée d'exécution (C)
T1	0	2
T2	0	X=3
T3	0	1



Temps moyen d'attente = (0 + 3 + 4) / 3 = 2,33

Temps moyen de rotation = (2 + 6 + 5) / 3 = 4,33

Exercices à corriger

Exercice N°11 :

Soit un système temps réel avec 3 tâches ayant les paramètres suivants :

Tâche	(r)	(C)	(D)	(P)
T1	0	2	5	6
T2	0	2	4	8
T3	0	4	8	12

1- Calculer la période d'étude de ce système.

2- Ce système est-il ordonnançable avec **RM** et **DM** ? Vérifier avec le diagramme de Gantt ?

3- Quel algorithme à priorités fixe vous semble le plus adapté pour cette application ?

4- Même question pour **EDF** et **LLF** ?

Exercice N°12 : (Concours Doctorat Université Oum Elbouagui 2019)

Un système d'arrosage automatique doit arroser trois types de plantes :

- les plus fragiles qui doivent être arrosées pendant 10 minutes, toutes les 40 minutes,
- une deuxième catégorie qui doit recevoir de l'eau pendant 20 minutes, toutes les heures,
- enfin, des plantes d'un troisième type qu'il faut arroser toutes les 80 minutes,

pendant 20 minutes. L'arrosage peut se faire de façon fractionnée, c'est-à-dire s'interrompre et reprendre.

Question A :

On cherche une solution pour le partage de l'eau entre ces différentes variétés de plantes :

1. Définir la liste des tâches à accomplir,
2. Puis, pour les stratégies RM et EDF :
   - Calculer l'ordonnançabilité de ces tâches,
   - Donner un schéma d'utilisation du système d'arrosage à partir du temps 0.

Question B :

On veut maintenant se servir du système d'arrosage pour nettoyer les allées qui desservent les plantations. On décide de faire cet entretien pendant 10 minutes toutes les heures.
Cet entretien est-il possible pendant les arrosages : avec RM, pourquoi ?
avec EDF, pourquoi ?

Question C :

Pour nettoyer toutes les allées, il faut 20 minutes. Si l'entretien commence 1h30 après le début de l'arrosage des plantes,
Pourra-t-on avoir complètement nettoyé les allées :

1. au bout d'une heure ?
2. après 100 minutes ?
3. conclure la meilleure durée pour nettoyer toutes les allées ?