Système d'Exploitation II

Devoir Libre

Exercice 1 :

On considère 4 processus, A, B, C, D. On suppose que l'exécution des processus nécessite :

- Pour A : 7 unités de temps CPU, 3 unités de temps d'E/S et 5 unités de temps CPU.
- Pour B : 6 unités de temps CPU, 4 unités de temps d'E/S, 4 unités de temps CPU.
- Pour C : 5 unités de temps CPU.
- Pour D : 1 unité de temps CPU, 4 unités de temps d'E/S et 2 unités de temps CPU.

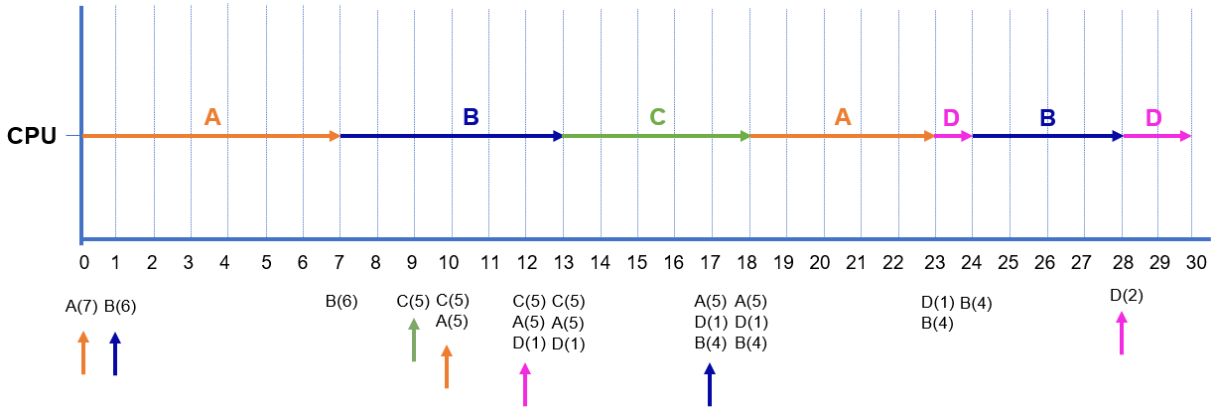
On suppose que :

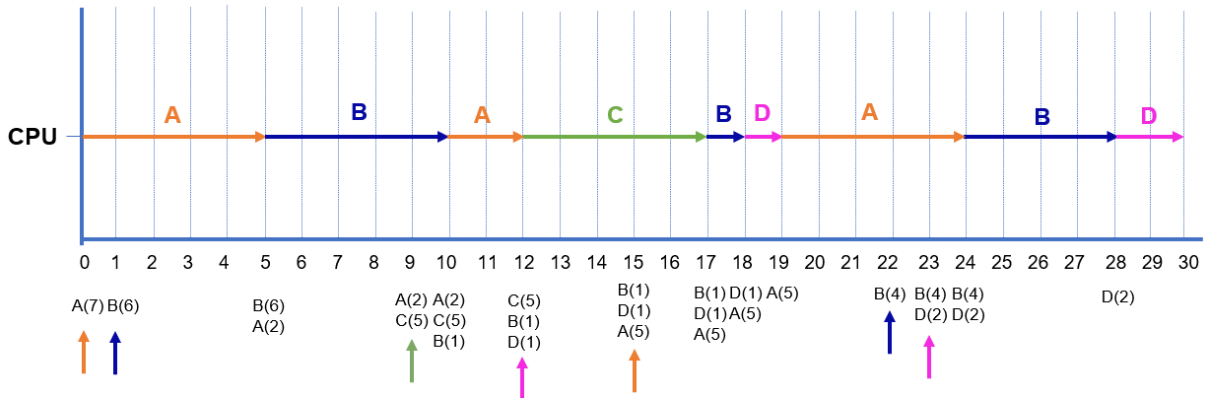
- A se présente en premier, à l'instant 0,
- B se présente à l'instant 1,
- C se présente à l'instant 9,
- D se présente à l'instant 12.

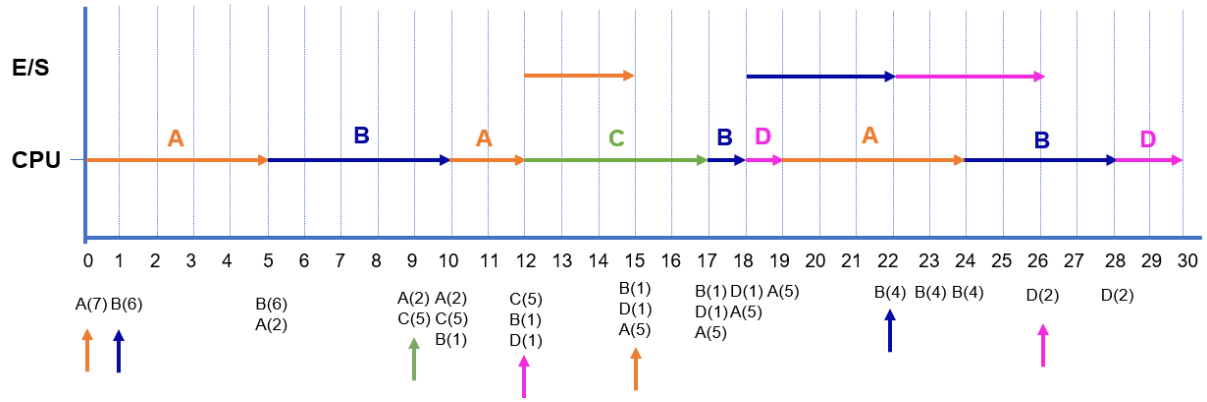
Montrez comment les 4 processus vont utiliser le processeur (avec un diagramme de Gantt) dans chacun des cas suivants :

1. Chaque processus a son propre périphérique d'E/S et l'ordonnanceur fonctionne selon Premier Arrivée Premier Servi FCFS (FIFO sans préemption).
2. Chaque processus a son propre périphérique d'E/S et l'ordonnanceur utilise l'algorithme du tourniquet (RR), avec un quantum de 5. Le temps de commutation est égal à 0.
3. Les trois processus utilisent le même périphérique d'E/S dont la file d'attente est gérée par l'algorithme FCFS (FIFO). L'ordonnanceur du processeur utilise l'algorithme du tourniquet, avec un quantum de 5. Le temps de commutation est supposé égal à 0.
4. Donnez, dans chaque cas, le temps d'exécution moyen.

======== Solution =======







$$Tm1 = \frac{(23-0)+(28-1)+(18-9)+(30-12)}{4} = 19.25$$

$$Tm2 = \frac{(24-0)+(28-1)+(17-9)+(30-12)}{4} = 21.75$$

$$Tm3 = \frac{(24-0)+(28-1)+(17-9)+(30-12)}{4} = 21.75$$

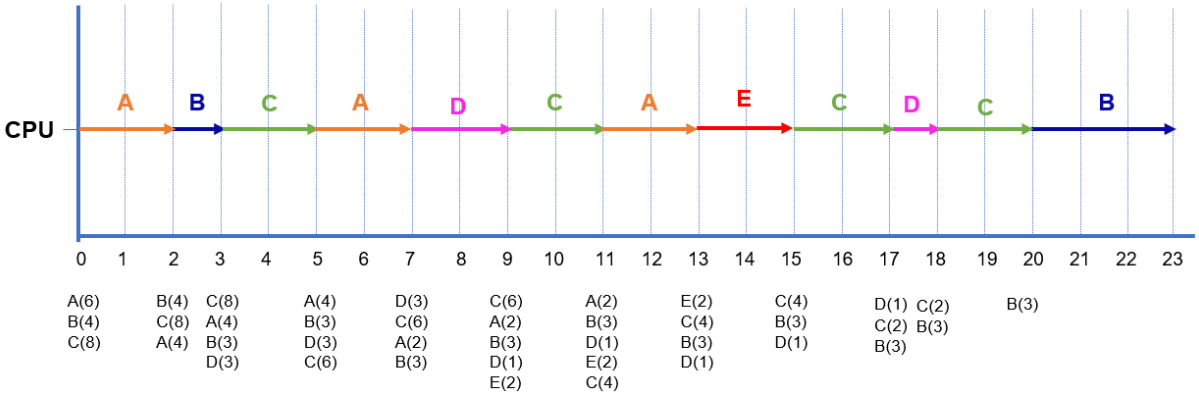
Exercice 2 :

On considère trois (3) processus A, B, C dont les durées d'exécution sont respectivement 6, 4 et 8 unités de temps. Les trois processus se présentent à l'instant 0.

On fait l'hypothèse suivante : après 1 unité de temps d'exécution, le processus B crée un processus fils (qu'on appellera D) dont la durée d'exécution est de 3 unités de temps. Le processus D après 2 unités de temps d'exécution crée à son tour un nouveau processus fils E, dont la durée d'exécution est de 2 unités de temps. On admet qu'un processus ayant créé un fils doit se bloquer jusqu'à la terminaison de son processus fils.

En supposant que tous les processus sont gérés en utilisant l'ordonnancement Tourniquet « Round-Robin » avec un quantum égal à 2 unités de temps, dessinez le digramme de Gantt.

======== Solution =======



Exercice 3 :

Un système d'arrosage automatique doit arroser trois types de plantes les plus fragiles qui doivent être arrosés pendant 1 minutes toutes les 4 minutes une deuxième catégorie qui doit recevoir de l'eau pendant 2 minutes toutes les 6 min enfin des plantes d'un troisième type qu'il faut arroser toutes les 8 minutes pendant 2 minutes.
L'arrosage peut se faire de façon fractionnée c'est à dire s'interrompre et reprendre.

Question 1 :

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

1. Définir la liste d'états s'accomplir
2. Pour les stratégies RMA. Donner un schéma d'utilisation du système d'arrosage à partir du temps zéro.

Question 2 :

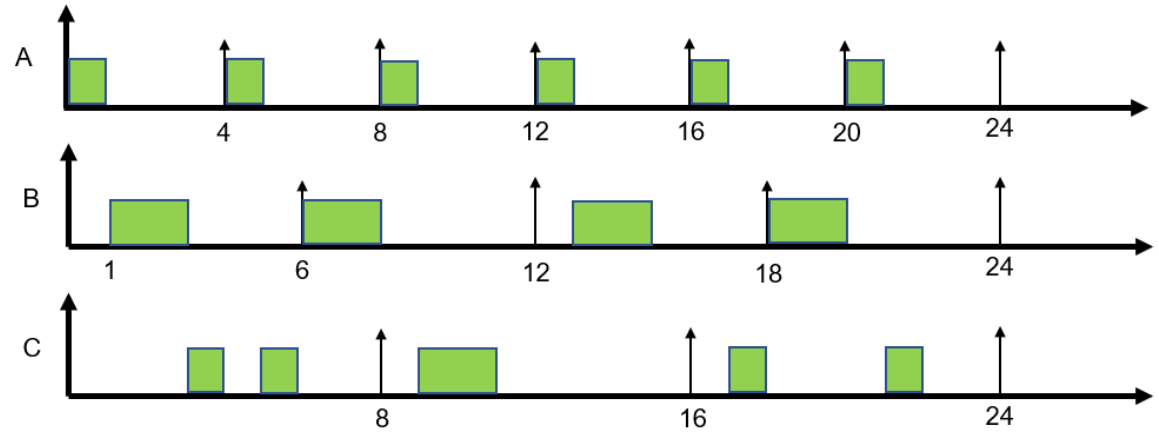
On veut maintenant se servir du système d'arrosage pour nettoyer les allées qui servent les plantations. On décide de faire cet entretien pendant 1 minute toutes les 6 minutes. Cet entretien est-il possible avec la stratégie RMA pendant les arrosages.

======== Solution =======

Question 1 :

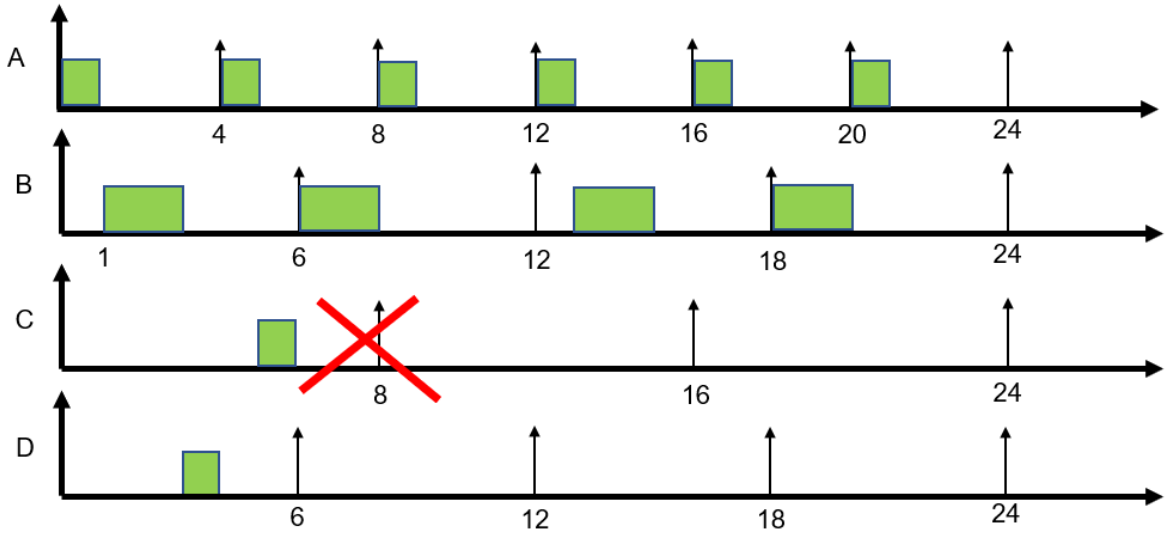
A(r=0, C=1, P=4)
B(r=0, C=2, P=6)
C(r=0, C=2, P=8)

Prio(A) > Prio(B) > Prio(C)



Question 2 :

Prio(A) > Prio(B) > Prio(D) > Prio(C)



Exercice 4 :

Combien de message Hello est affiché après l'exécution de chaque programme :

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
int main(){
    fork();
    if(fork())
        printf("Hello\n");

    printf("Hello\n");
    return 0;
}
```

6 Hello seront affichés

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
int main(){
        fork();
        fork();
        printf("Hello\n");
        return 0;
}
```

4 Hello seront affichés

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
```

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
```

```
int main(){
        fork();
        if(fork())
                fork();
        printf("Hello\n");
        return 0;
}
```

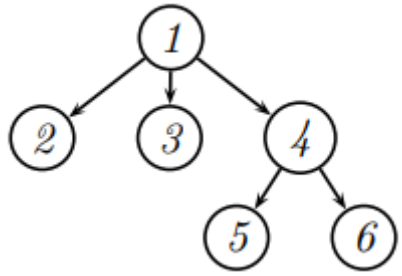
6 Hellos seront affichés

```
int main(){
        fork();
        if(fork())
             fork();
        else
             printf("Hello\n");
        return 0;
}
```

2 Hellos seront affichés

Exercice 5 :

Ecrire un programme qui engendre l'arbre généalogique suivant :



======== Solution =======

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
int main(){

    int p1,p2,p3,p4,p5,p6;

    if((p1 = fork())==0){
        printf("p1\n");

        if((p2 = fork())==0){
            printf("p2\n");
            exit(0);
        }
        if((p3 = fork())==0){
            printf("p3\n");
            exit(0);
        }
        if((p4 = fork())==0){
            printf("p4\n");
```

```
            if((p5 = fork())==0){
                printf("p5\n");
                exit(0);
            }

            if((p6 = fork())==0){
                printf("p6\n");
                exit(0);
            }
            exit(0);
        }
        exit(0);
    }
    return 0;
}
```