Systèmes d’exploitation

Plan

- Introduction aux systèmes d'exploitation
- Fonctions de base
  - Gestion des fichiers
  - Gestion des processus
  - **Gestion de la mémoire**

Gestion de la mémoire

- Introduction
- Système mono-tâche
- Système multi-tâche
  - Allocation contiguë
    - Partitions variables
    - Partitions fixes
    - Swapping
    - Compactage
  - Allocation non contiguë
    - Segmentation
    - Pagination
    - Mémoire virtuelle

Gestion de la mémoire

- Introduction
- Système mono-tâche
- Système multi-tâche
  - Allocation contiguë
    - Partitions variables
    - Partitions fixes
    - Swapping
    - Compactage
  - Allocation non contiguë
    - Segmentation
    - Pagination
    - Mémoire virtuelle

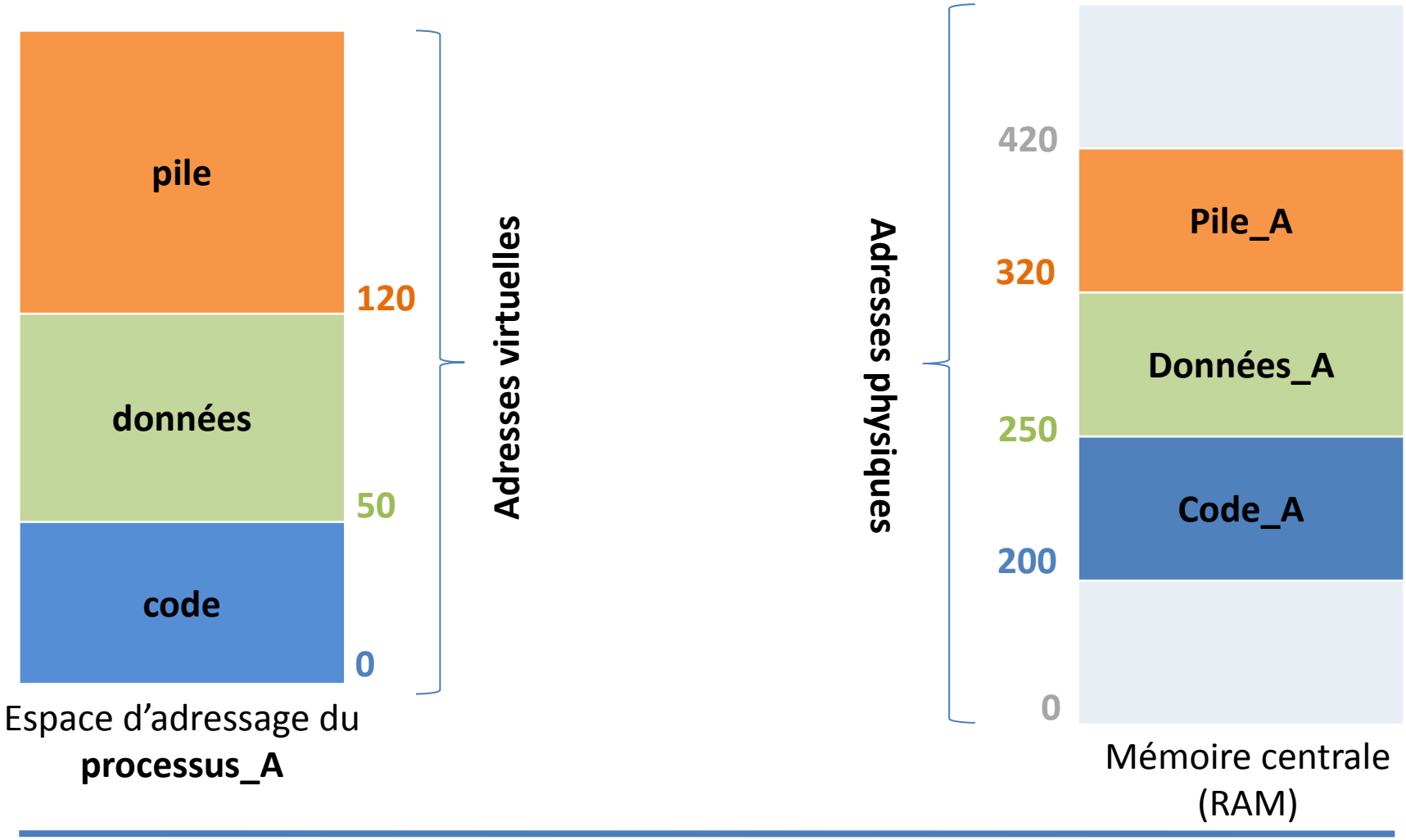
Introduction

- La mémoire centrale est une ressource requise par tout processus
- Un processus doit être chargé dans la mémoire centrale pour être exécuté
- Opérations sur la mémoire :
  - Demarrer_processus(p)=>Allouer(taille(p))
  - Terminer_processus(p)=>Liberer(zone_allouée_à (p))

Objectifs de la gestion de la mémoire

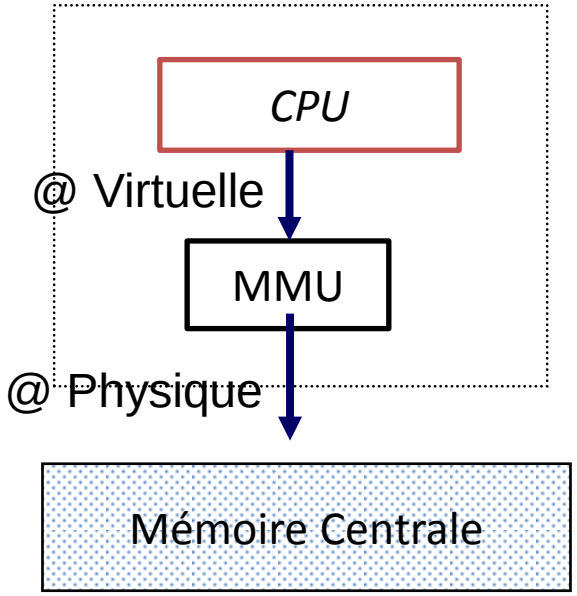
- Protection : par défaut, un processus ne doit pas pouvoir modifier les informations d'un autre,
- Partage : s'ils le demandent, plusieurs processus peuvent partager une zone mémoire commune,
- Réallocation/Mapping : chaque processus voit son propre espace d'adressage, numéroté à partir de l'adresse 0 (adresse virtuelle). Cet espace est physiquement implanté à partir d'une adresse quelconque (adresse physique).

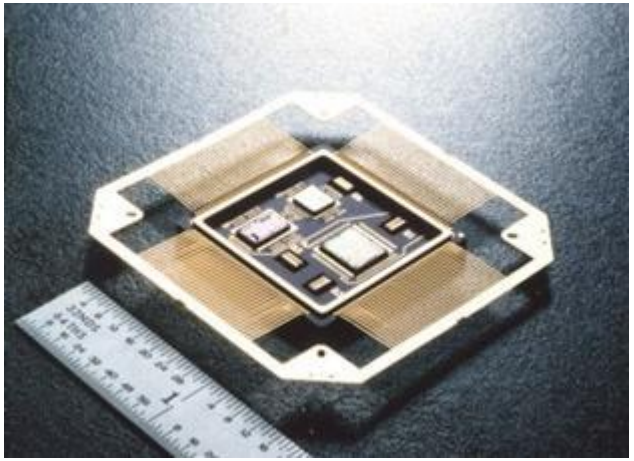
Objectifs de la gestion de la mémoire



Objectifs de la gestion de la mémoire

- Adresse virtuelle : générée par la CPU;
- Adresse physique : adresse reçue par la mémoire centrale
- **Memory Management Unit (MMU)** : qui fait transcodage de l'adresse virtuelle vers l'adresse physique



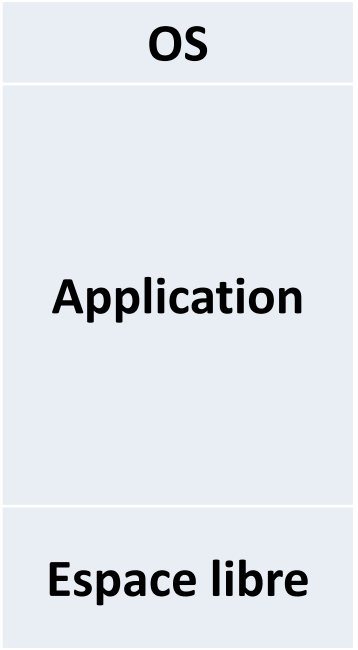


Gestion de la mémoire

- Introduction
- Système mono-tâche
- Système multi-tâche
  - Allocation contiguë
    - Partitions variables
    - Partitions fixes
    - Swapping
    - Compactage
  - Allocation non contiguë
    - Segmentation
    - Pagination
    - Mémoire virtuelle

Gestion de la mémoire pour système mono-tâche

Dans le cas des systèmes mono-tâche, la gestion de la mémoire est assez simple. Il suffit de réserver une partie de la mémoire au système d'exploitation. L'application est ensuite rangée dans l'espace restant qui est libéré sitôt que l'application est terminée.



Gestion de la mémoire

- Introduction
- Système mono-tâche
- Système multi-tâche
  - Allocation contiguë
    - Partitions variables
    - Partitions fixes
    - Swapping
    - Compactage
  - Allocation non contiguë
    - Segmentation
    - Pagination
    - Mémoire virtuelle

Gestion de la mémoire pour système multi-tâche

Il existe plusieurs méthodes d'allouer la mémoire aux processus :

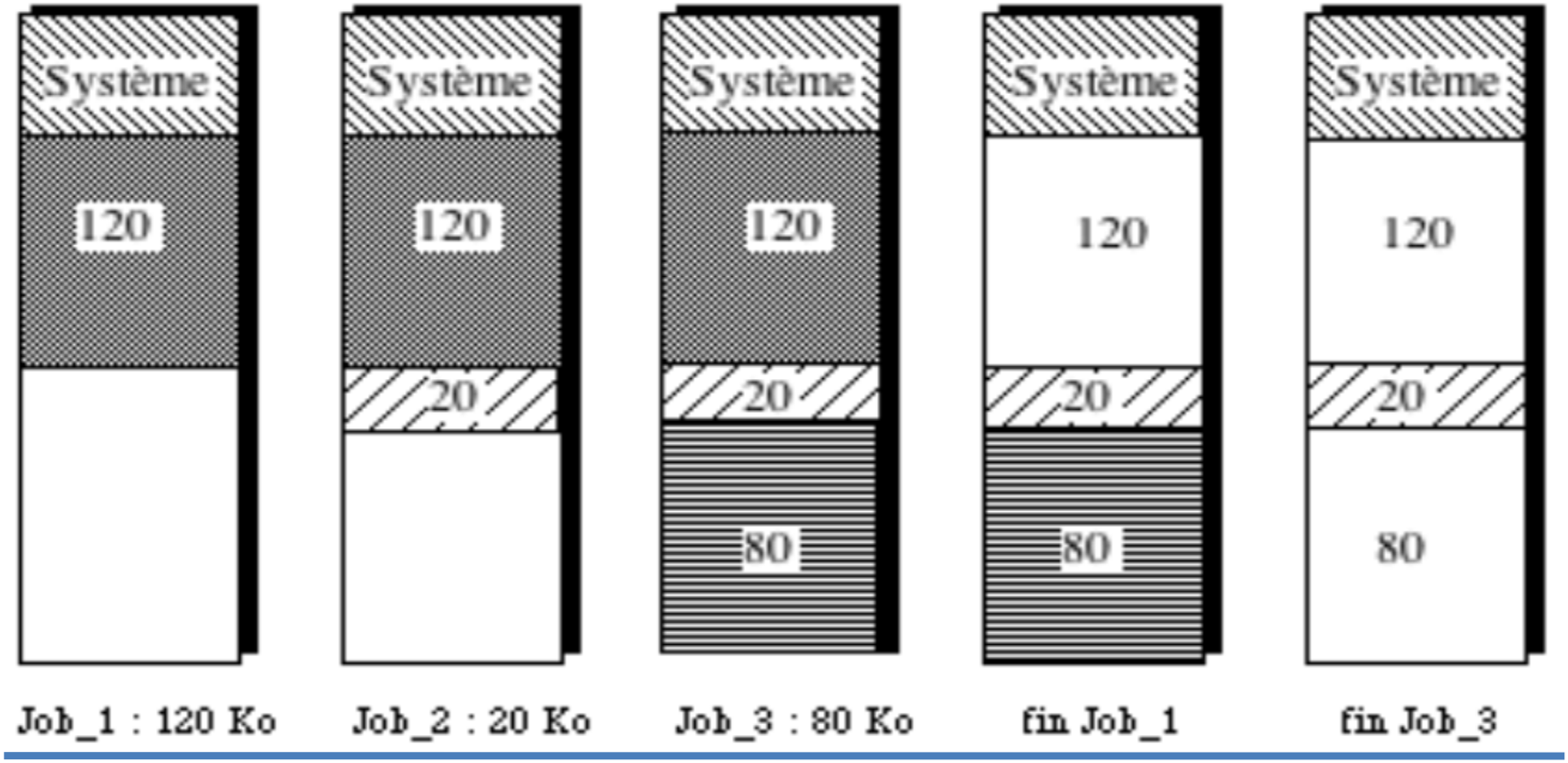
- Allocation contiguë (le processus est considéré comme un seul bloc indivisible)
  - Partitions variables et partitions fixes
- Allocation non contiguë (le processus peut être stocker sur des zones mémoire éparpillées)
  - Pagination et segmentation

Gestion de la mémoire

- Introduction
- Système mono-tâche
- Système multi-tâche
  - Allocation contiguë
    - Partitions variables
    - Partitions fixes
    - Swapping
    - Compactage
  - Allocation non contiguë
    - Segmentation
    - Pagination
    - Mémoire virtuelle

Partitions variables

L'espace mémoire est alloué dynamiquement, de façon **contiguë**, lors de l'implantation des processus en mémoire



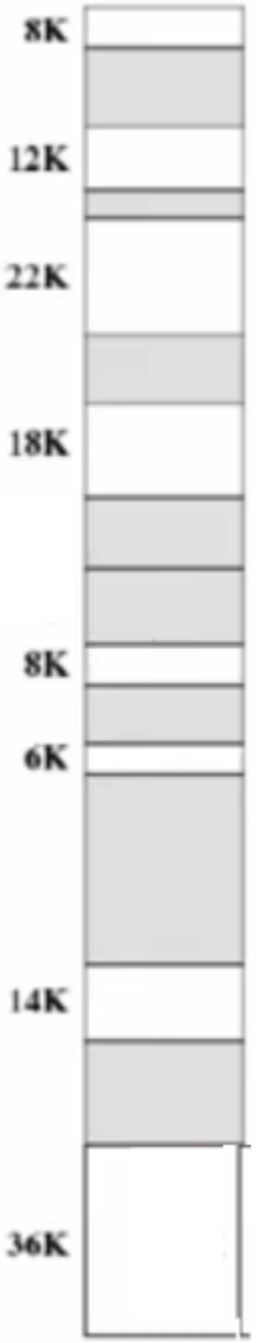
Partitions variables

Si plusieurs partitions sont susceptibles de recevoir un processus, on peut adopter l'une des stratégies de **placement** suivantes :

- **First fit** : on range le processus dans la première partition libre et suffisamment grande trouvée
- **Best Fit** : on va chercher la partition dont la taille approche au mieux celle du processus à charger en mémoire.
  - Inconvénient : création de bloc minuscules non réutilisables (fragmentation externe)
- **Worst Fit** : on met le processus dans la plus grande partition libre.

Partitions variables

Selon chacun des trois algorithmes, First Fit, Best Fit et Worst fit, où sera placé un block de 16 k?

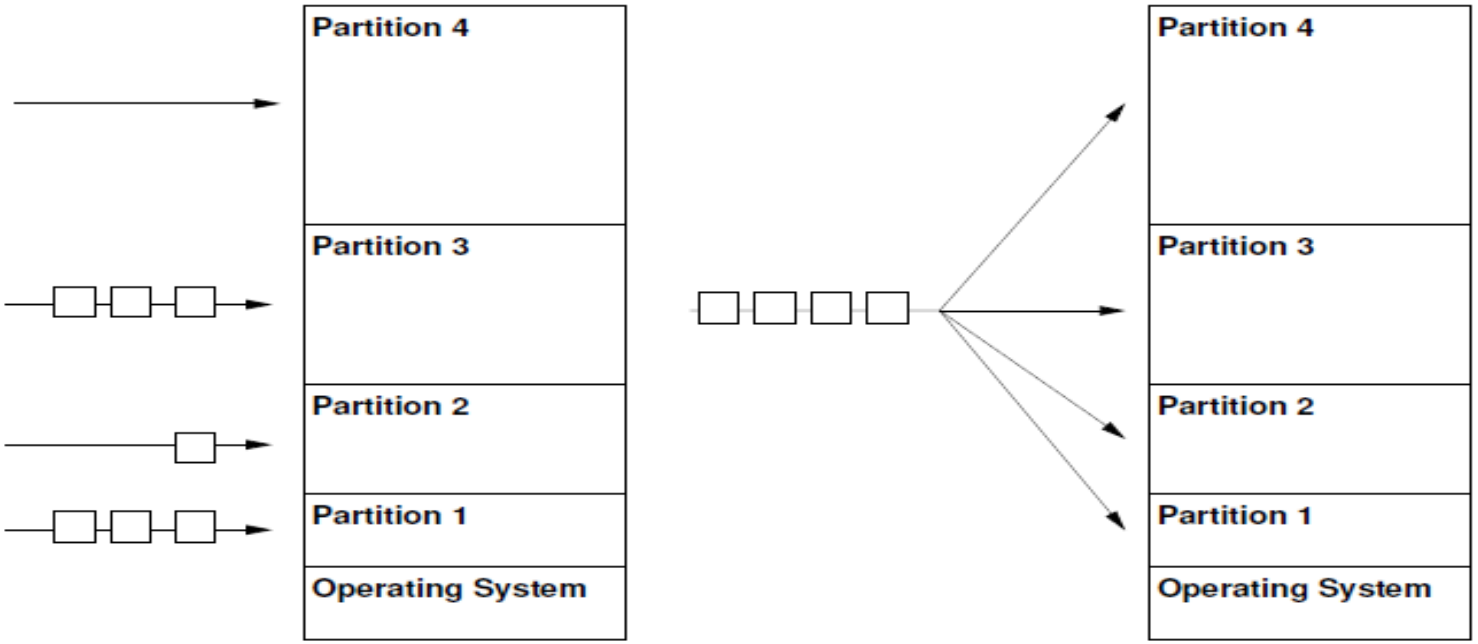


Gestion de la mémoire

- Introduction
- Système mono-tâche
- Système multi-tâche
  - Allocation contiguë
    - Partitions variables
    - Partitions fixes
    - Swapping
    - Compactage
  - Allocation non contiguë
    - Segmentation
    - Pagination
    - Mémoire virtuelle

Partitions fixes

- La mémoire est divisée en partitions fixes dès le démarrage du système.
- Les partitions sont de différentes tailles
- On peut définir une file d'attente par partition, ou une file commune à toutes les partitions

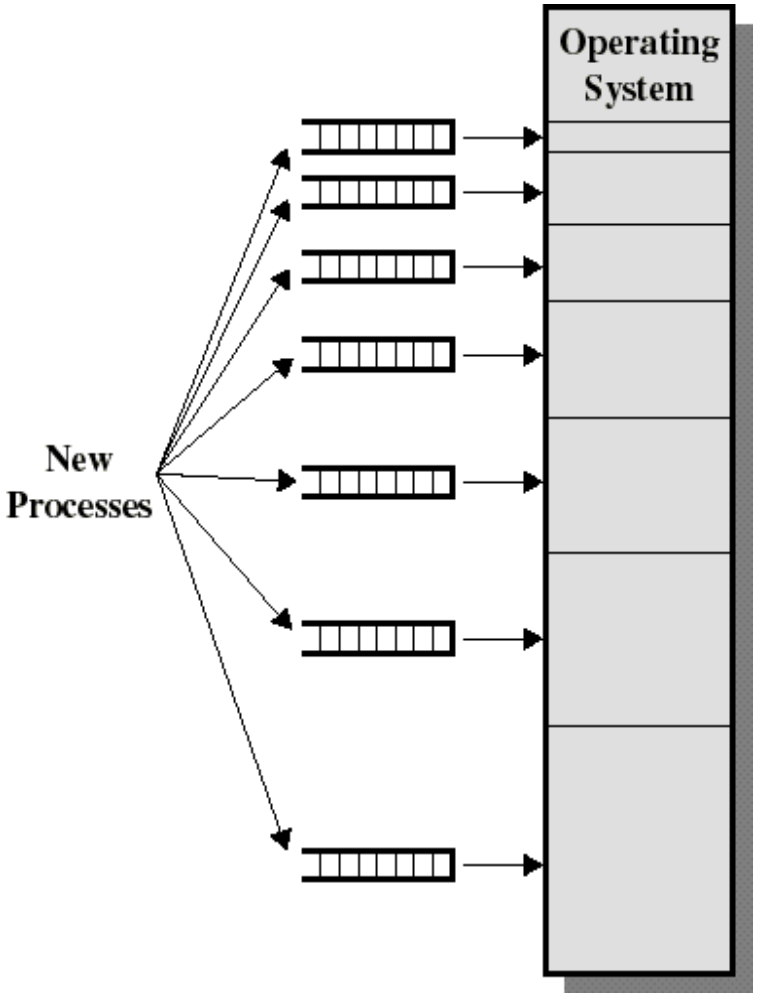


Partitions fixes

- 1 file par partition
- Choisir la file en fonction de l'algorithme de placement utilisé : First fit, Best fit ou Worst fit.

Problèmes:

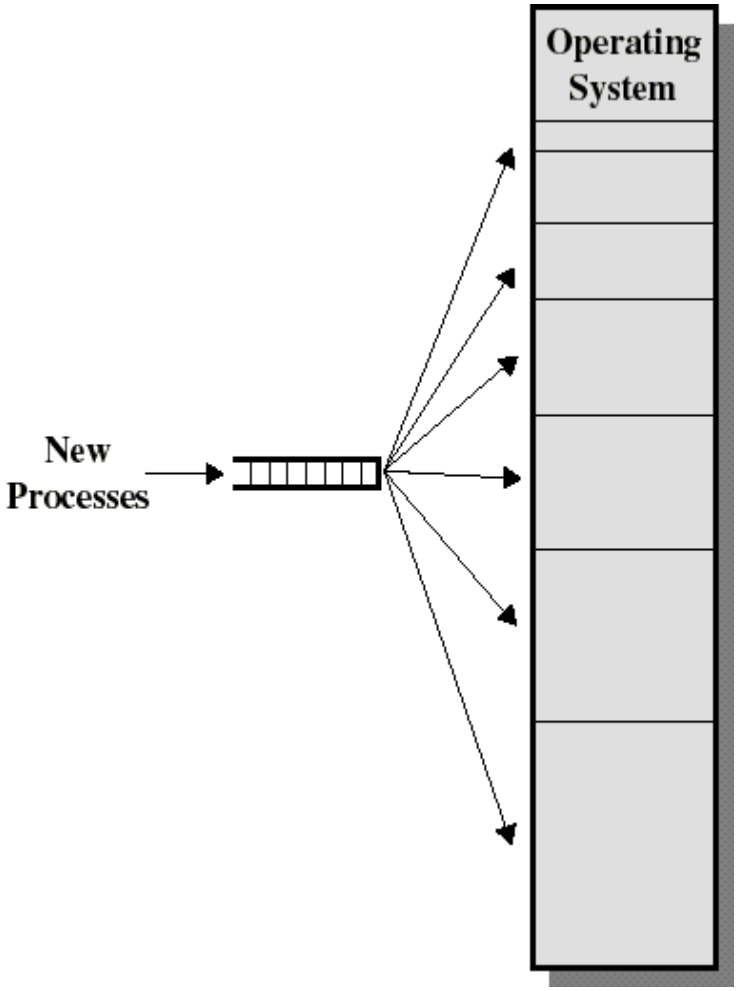
- certaines files seront vides, alors que d'autres surchargées.
- Espace inutilisé à l'intérieur des partitions (fragmentation interne).



Partitions fixes

- 1 file pour toutes les partitions
- Choisir la partition en fonction de l'algorithme de placement utilisé : First fit, Best fit ou Worst fit.

- Problème: On pourrait allouer trop de mémoire à un programme (fragmentation interne)

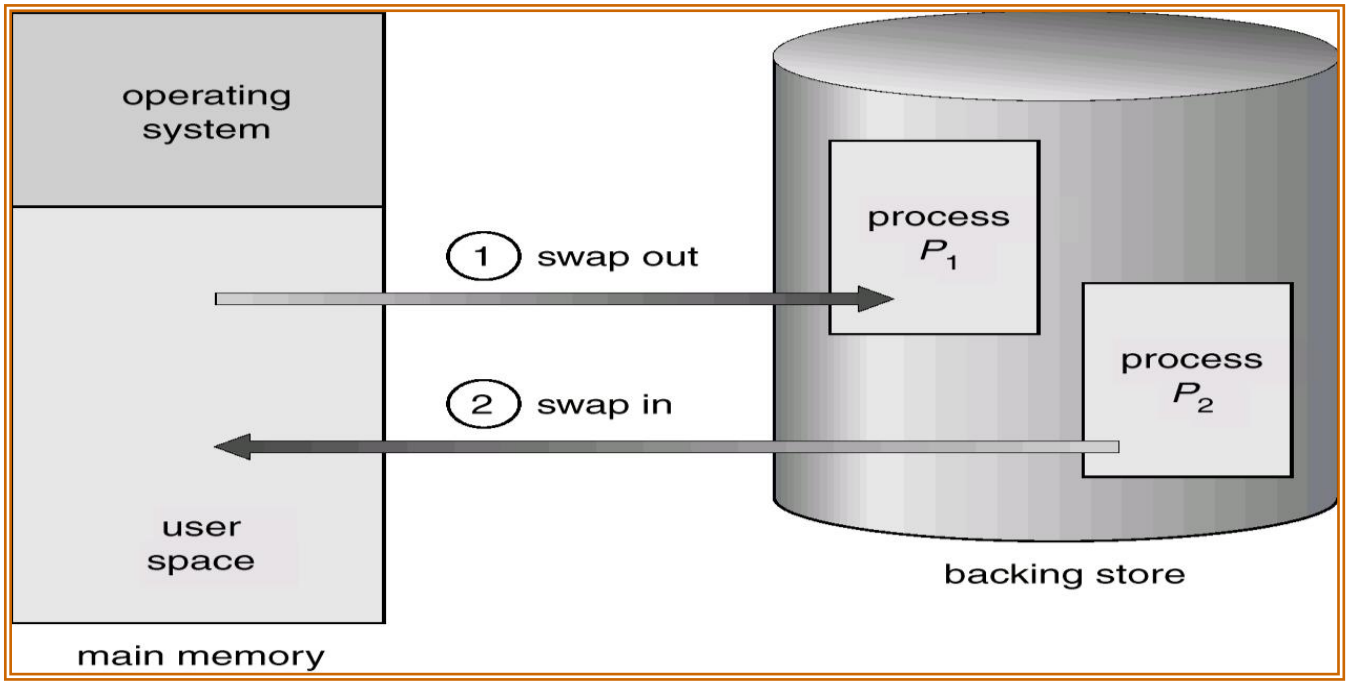


Gestion de la mémoire

- Introduction
- Système mono-tâche
- Système multi-tâche
  - Allocation contiguë
    - Partitions variables
    - Partitions fixes
    - Swapping
    - Compactage
  - Allocation non contiguë
    - Segmentation
    - Pagination
    - Mémoire virtuelle

Va-et-vient (Swapping)

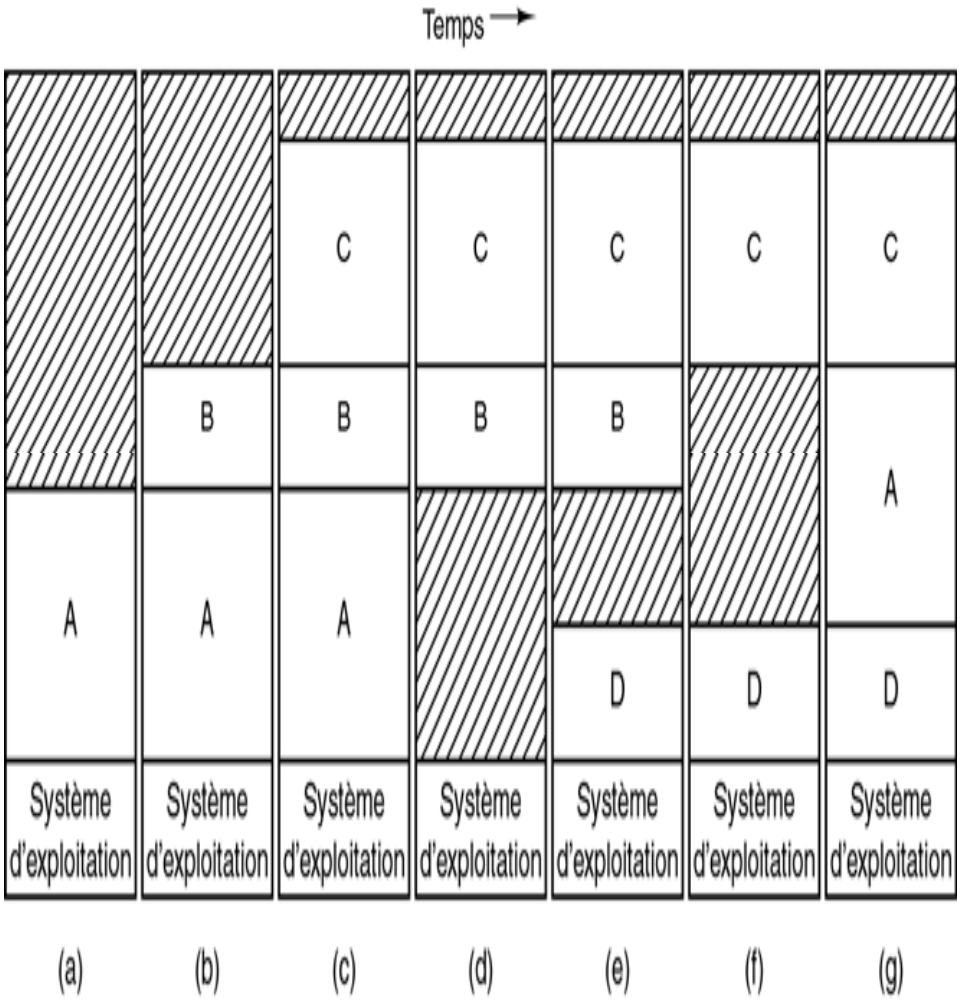
Un processus (dans son intégralité) peut être **swappé** temporairement en dehors de la mémoire centrale vers un stockage secondaire (**swap file** sur le disque dur), et puis remis en mémoire pour continuer son exécution



N.B : Pour optimiser l'utilisation du CPU, on swap un processus en attente d'E/S

Va-et-vient (Swapping)

i. (a) A est en mémoire
ii. (b,c) B et C sont crées ou chargés depuis le disque
iii. (d) A est transférée sur le disque
iv. (e,f) D arrive, tandis que B s'en va
v. (g) A revient



Allocation contiguë

Inconvénients :

- Fragmentation **interne** : partitions fixes
- Fragmentation **externe** : partitions variables
- Impossibilité d'exécuter un programme de taille supérieure à celle de la mémoire centrale

N.B : on parle de fragmentation interne, lorsque l'espace inutilisé est dans les partitions. Tandis que pour la fragmentation externe, l'espace inutilisé est entre les partitions.

Gestion de la mémoire

- Introduction
- Système mono-tâche
- Système multi-tâche
  - Allocation contiguë
    - Partitions variables
    - Partitions fixes
    - Swapping
    - Compactage
  - Allocation non contiguë
    - Segmentation
    - Pagination
    - Mémoire virtuelle

Compactage

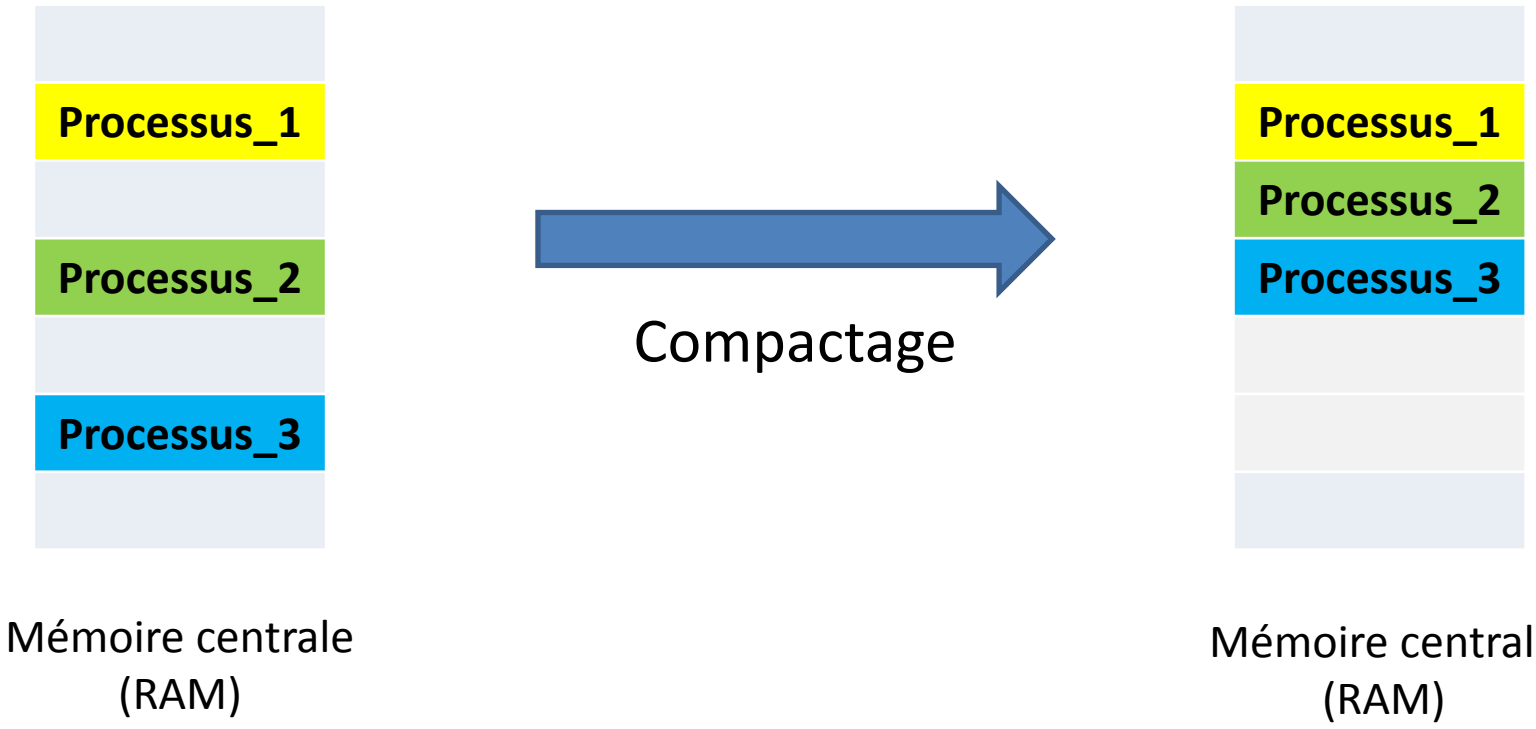
Une solution pour la fragmentation externe (partitions variables)

Définition : Le compactage (ou défragmentation) est une opération réalisée par le système d'exploitation consistant à déplacer tous les processus vers des emplacements contigües pour avoir un grand espace libre.

Inconvénients

- On consomme du temps CPU pour faire le transfert qui risque d'être long.

Compactage


Processus_1
Processus_2
Processus_3
Compactage
Processus_1
Processus_2
Processus_3
Mémoire centrale (RAM)
Mémoire centrale (RAM)

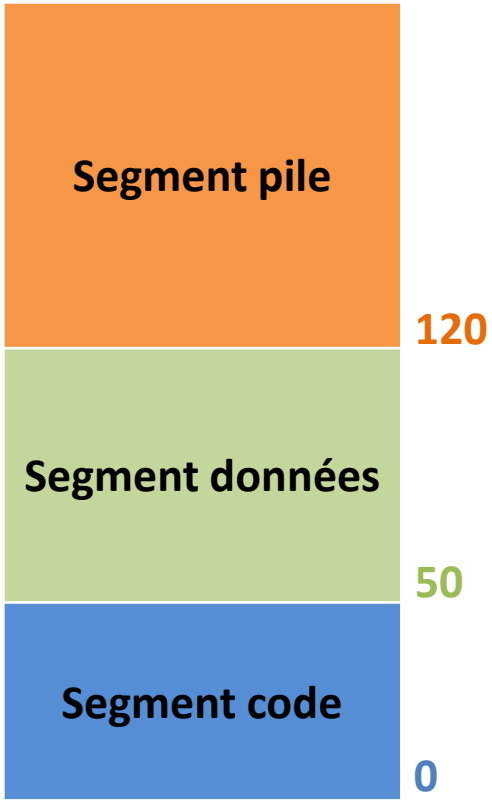
Gestion de la mémoire

- Introduction
- Système mono-tâche
- Système multi-tâche
  - Allocation contiguë
    - Partitions variables
    - Partitions fixes
    - Swapping
    - Compactage
  - Allocation non contiguë
    - Segmentation
    - Pagination
    - Mémoire virtuelle

Segmentation

- Un segment est une suite d'emplacements contiguës
- Les segments correspondent à un découpage **logique** d'un programme : segment de données, segment de code, segment de pile, etc.
- Chaque segment peut croître indépendamment des autres
- L'adresse virtuelle a la forme (segment, déplacement). Elle est traduite en adresse physique par le biais d'une table de segments.
- Une table de segments contient ces champs :
  - **Base** : l'adresse de début du segment
  - **limite** : la longueur du segment.
  - **Privilèges** de l'utilisateur sur le segment: lecture, écriture ou exécution.

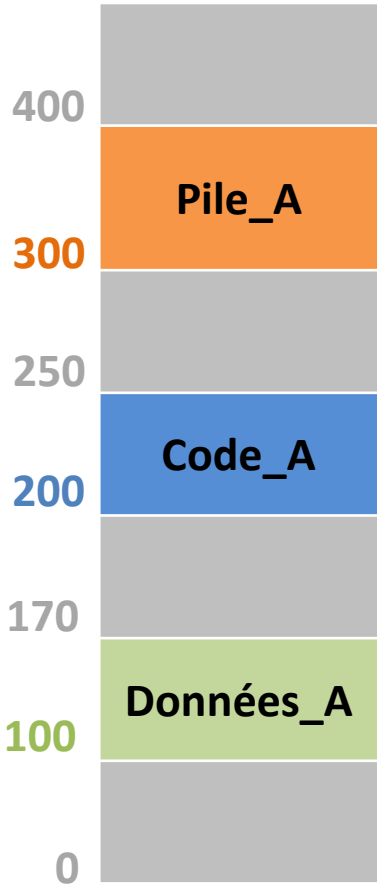
Segmentation



Espace d'adressage du **processus_A**

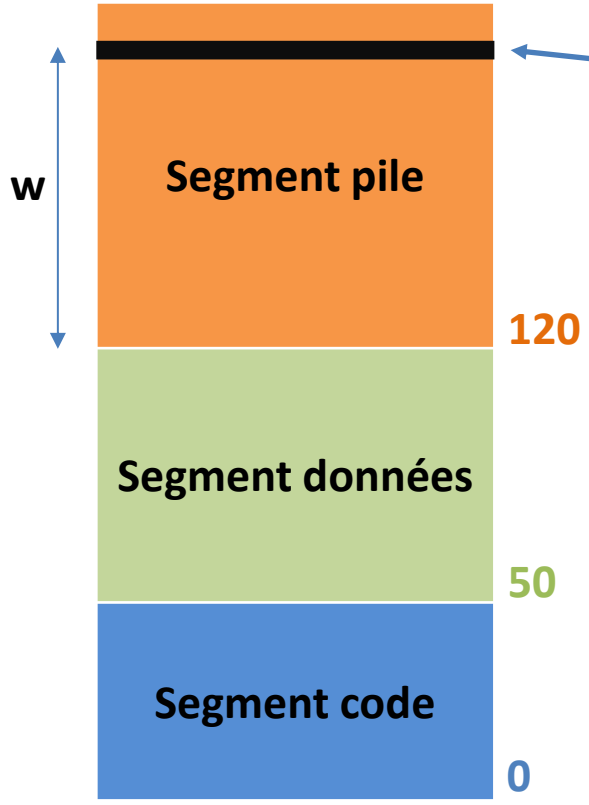
	Limite/taille (octets)	Adresse début du segment sur la RAM
C	50	200
D	70	100
P	100	300

Table de segment du **processus_A**



Mémoire centrale (RAM)

Segmentation

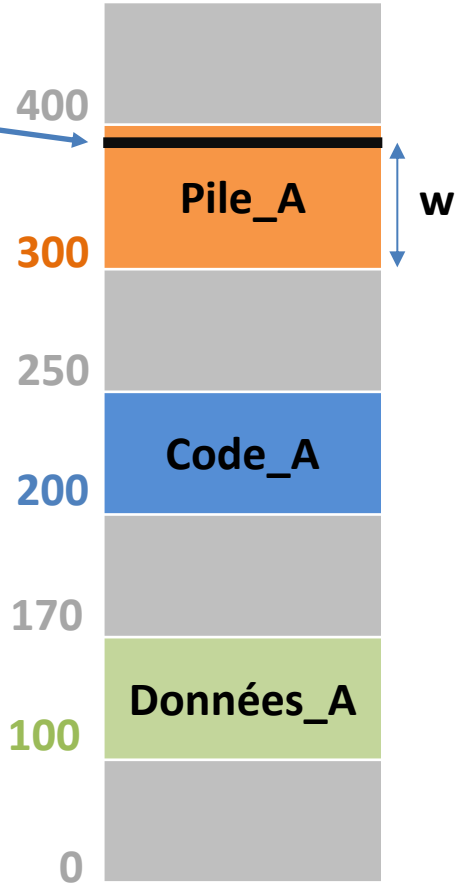


Espace d'adressage du **processus_A**

Quelles sont les adresses virtuelle et physique de ce bloc ?

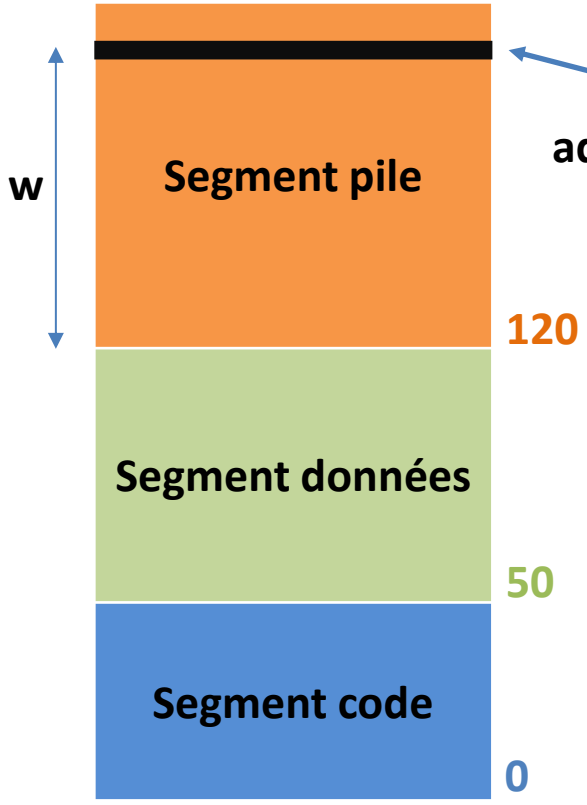
	Limite/taille (octets)	Adresse début du segment sur la RAM
C	50	200
D	70	100
P	100	300

Table de segment du **processus_A**



Mémoire centrale (RAM)

Segmentation



Espace d'adressage du **processus_A**

Adresse virtuelle = **(pile, w)**

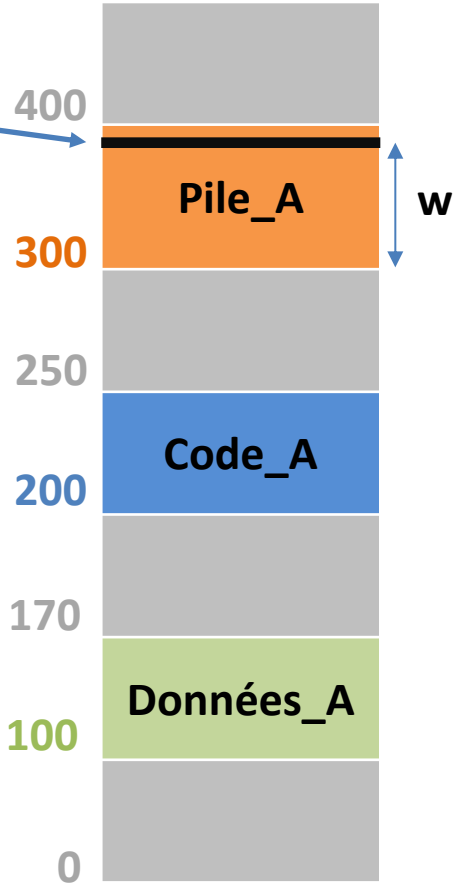
Adresse physique =

adresse physique du début du segment pile + w =

300 + w

	Limite/taille (octets)	Adresse début du segment sur la RAM
C	50	200
D	70	100
P	100	300

Table de segment du **processus_A**



Mémoire centrale (RAM)

Gestion de la mémoire

- Introduction
- Système mono-tâche
- Système multi-tâche
  - Allocation contiguë
    - Partitions variables
    - Partitions fixes
    - Swapping
    - Compactage
  - Allocation non contiguë
    - Segmentation
    - Pagination
    - Mémoire virtuelle

Pagination

- L'espace d'adressage du processus est divisé en pages, la mémoire «centrale» est divisée en cadres,
- Taille de la page = taille du cadre,
- Table des pages : elle fait correspondre une page logique à un cadre physique. Chaque processus a sa propre table des pages.

Pagination

Chaque entrée de la table des pages contient les informations suivantes:

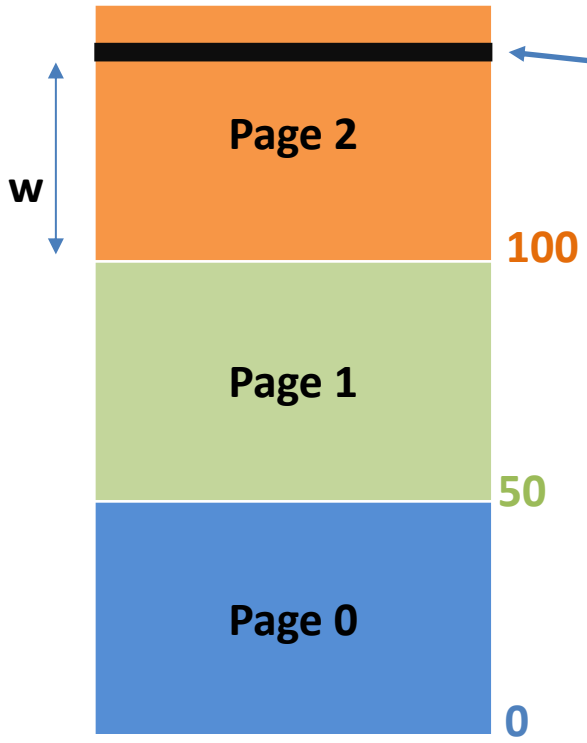
- Le **numéro de cadre** alloué à cette page.
- Le bit de **présence** indique s'il y a un cadre alloué à cette page.
- Le bit de **protection** de la page indique les opérations autorisées sur cette page par le processus.
- Le bit de **référence** est positionné lors d'un accès quelconque à la page.
- Le bit de **modification** est positionné lors d'un accès en écriture à la page.

Pagination

Passage de l'adresse virtuelle à l'adresse physique :

- L'adresse virtuelle est divisée en deux champs : numéro de page virtuelle et déplacement dans cette page (ici appelés p et w)
- Le contenu de l'entrée p de la table de pages TP[p] donne le numéro de cadre physique p' où est chargée p.
- Pour trouver l'adresse physique, il faut combiner le déplacement dans la page (w) au numéro de cadre physique trouvé (p').

Pagination

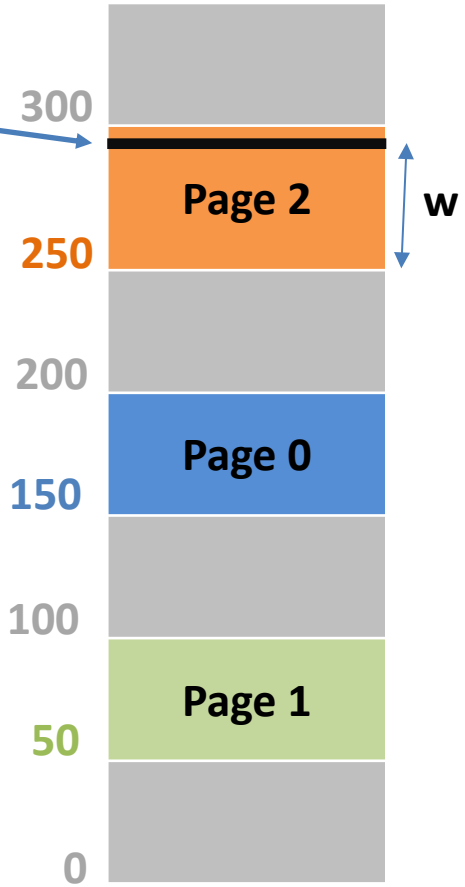


Espace d'adressage du **processus_A**

Quelles sont les adresses virtuelle et physique de ce bloc ?

N° page	N° cadre
0	3
1	1
2	5

Table des pages du **processus_A**



Mémoire centrale (RAM)

Pagination

Espace d'adressage du **processus_A**

Adresse virtuelle = **(2, w)**

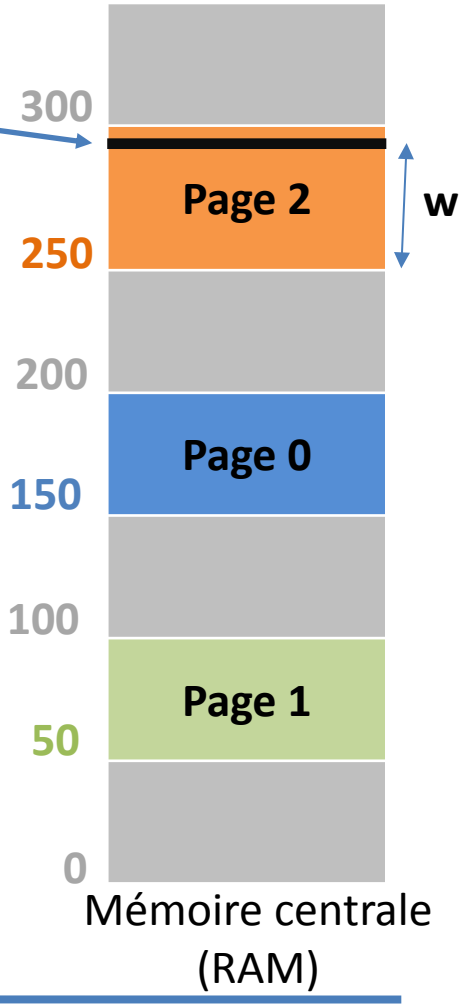
Adresse physique =

N° cadre x taille page + w =

5 x 50 + w

N° page	N° cadre
0	3
1	1
2	5

Table des pages du **processus_A**



Mémoire centrale (RAM)

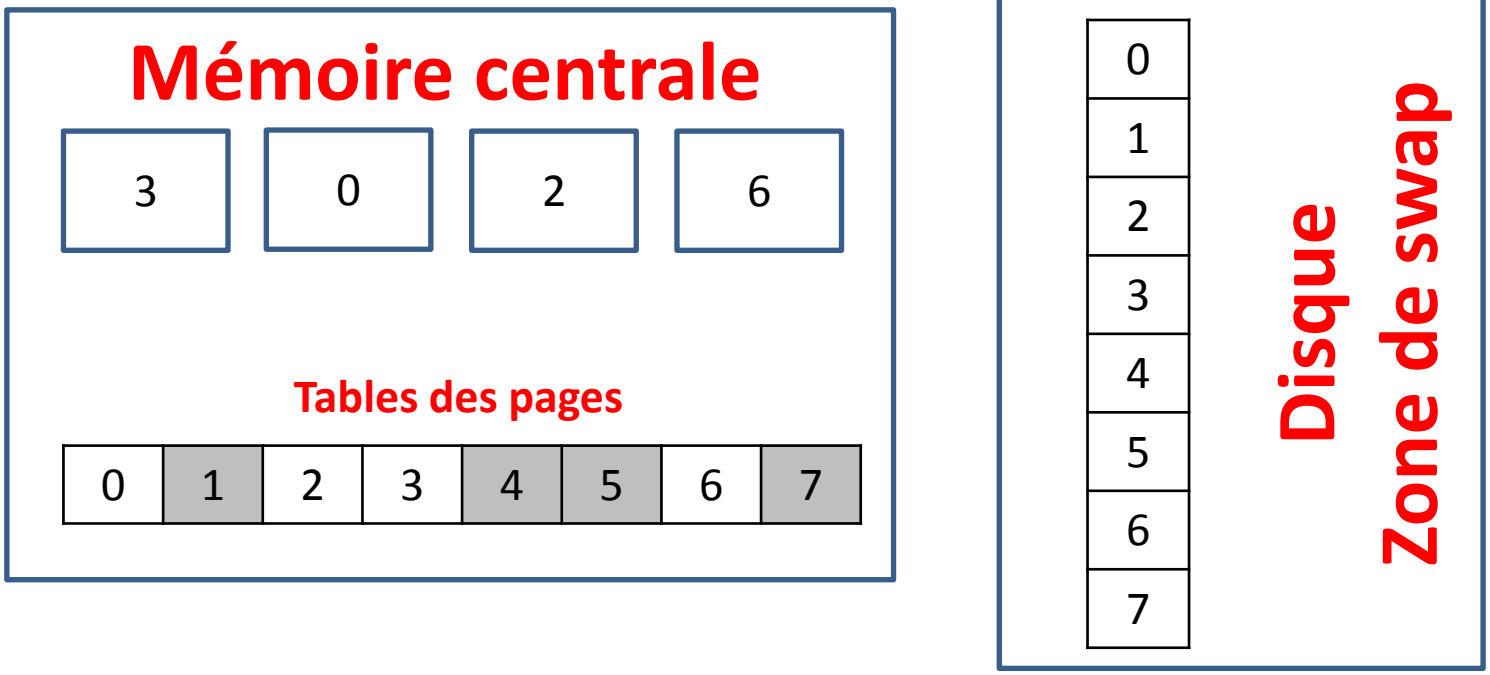
Gestion de la mémoire

- Introduction
- Système mono-tâche
- Système multi-tâche
  - Allocation contiguë
    - Partitions variables
    - Partitions fixes
    - Swapping
    - Compactage
  - Allocation non contiguë
    - Segmentation
    - Pagination
    - Mémoire virtuelle

Mémoire virtuelle

- Soit M la taille de la mémoire, physique, libre, sur la machine. Soit T la taille de l'espace d'adressage du processus :
  - Si T < M, on parle simplement de pagination en mémoire réelle. Dans ce cas toutes les pages du processus sont présentes en mémoire centrale.
  - Si T > M, on parle de mémoire virtuelle et dans ce cas une partie des pages est présente en mémoire centrale l'autre réside sur la mémoire secondaire (**swap file**). Lorsque le processus a besoin d'une page qui est sur la mémoire secondaire, celle-ci est copiée en mémoire centrale pour être exécutée.

Mémoire virtuelle (exemple)



- Au démarrage d'un processus, son espace d'adressage est copié dans la zone de swap.
- Dans la tables des pages, le bit de présence =1 pour les pages qui existent en mémoire centrale (0,2,3 et 6)
- Dans cet exemple, si le processeur souhaite accéder à la page N°1, une interruption **défaut de page** est générée.

Mémoire virtuelle

Pour résoudre un **défaut de page**, le système d'exploitation réalise les actions suivantes :

1. Cherche la page manquante dans le swap,
2. Copie la page manquante dans la mémoire centrale
   a) S'il existe un cadre libre, Copie la page manquante dans ce cadre
   b) Sinon **remplace** une page déjà chargée en mémoire.
      N.B : Si la page à remplacer a été modifiée, il faut la copier dans le swap avant de l'écraser
3. Met à jour la table des pages

Mémoire virtuelle

Il existe plusieurs stratégies de **remplacement** des pages (indiquent quelle page doit être remplacée) :

- **Moins Récemment Utilisée (MRU)** : implique que l'on conserve une trace des dates d'accès aux pages.

Algorithme le plus utilisé car il réduit le nombre de défaut de page

- **Moins Fréquemment Utilisée (MFU)** : implique que l'on conserve une trace du nombre d'accès aux pages.

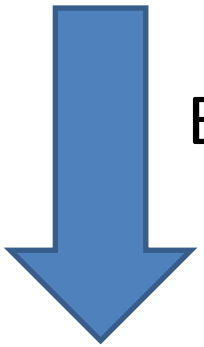
Problème des pages récemment chargées qui vont être remplacées même si elles sont fréquemment utilisées.

- **La Plus Ancienne (FIFO)** : Implique que l'on conserve une trace de l'ordre de chargement.

Problème des pages anciennes fréquemment utilisées.

Allocation non contigue

- Inconvénients :
  - Fragmentation: interne en pagination, externe pour le mode segmentation



En pratique ?

Segmentation paginée avec swap

Mémoire divisée en segments de taille fixe. Chaque segment est divisé en pages et chaque segment a sa propre table de pages.