Variable statique vs variable dynamique

Variable statique : caractéristiques

- Une variable déclarée (explicitement ou implicitement dans certains langages)
  - Le système d'exploitation attribue une place mémoire qui servira à contenir la valeur de la variable (demande faite par le compilateur)
  - La durée de vie de la variable est égale à celle du programme qui la contient et dans lequel elle est déclarée
- La variable statique (int k =7 ;) se caractérise par :
  - Son nom (ou désignation) i.e. ici k
  - Son type i.e. ici int
  - Son adresse mémoire (sa référence)
  - Sa valeur en tout instant donné (c'est une variable!)

Variable statiques, 1 modèle d'utilisation

- Début

  Déclaration de A et de B

  Utilisation de A

  Autres travaux

  Utilisation de A et B

  Utilisation de B

- fin

Structure statique et structure dynamique

- Variable associée à une Structure statique
  - Temps d'existence est égale ≈ à celui du programme qui la contient.
  - Mise à la disposition du programme qui la contient dès le commencement,
  - Le programme conserve son droit d'accès à cette variable jusqu'à la fin de son exécution.
- Variable associée à une Structure dynamique
  - Temps d'existence peut être inférieur à celui du programme qui la contient.
  - Mise à la disposition du programme qui la contient à la demande de ce dernier; Il conserve ses privilèges d'accès à cette variable mais peut décider de la restituer à tout moment !

Structure statique et structure dynamique

- Début

  Réservation de A

  Utilisation de A

  Réservation de B

  Utilisation de A et B

  Restitution de A

  Utilisation de B

- fin

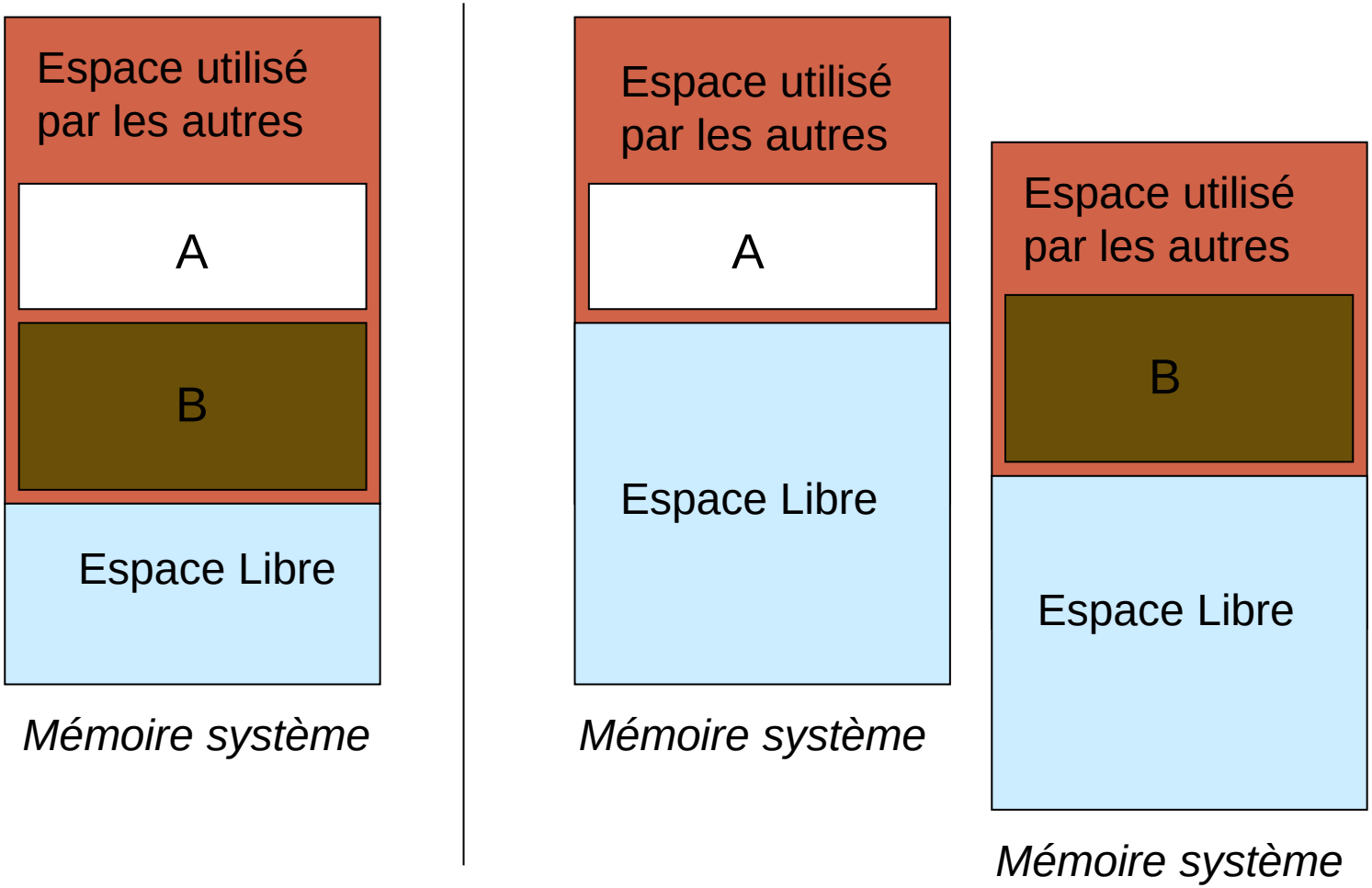
Structure dynamique

- Intérêt
  - Optimisation de l'utilisation de la mémoire du système.
  - Amélioration, à mémoires égales, de la précision de calcul.
  - Nouvelle possibilité en Programmation:
    - Augmentation (ajout d'éléments) et diminution des structures plus souple.
    - Minimisation des coûts des opérations (insertion suppression) au milieu de la structure.

```
Début
    Déclaration de A et de B
    Utilisation de A
    Utilisation de B
fin
```

```
Début
    Réservation de A
    Utilisation de A puis libération de A
    Réservation de B
    Utilisation de B puis libération de B
fin
```

Exemple: optimisation mémoire



Mémoire pour la variable dynamique

- Lorsqu'en un moment donné, on a besoin de mémoire pour « loger » une nouvelle variable dynamique, on s'adresse au système
- Par conséquent, il faut :
  - Phase 1 : formuler la demande
    - Préciser ce qu'on veut (Taille et Structuration)
  - Phase 2 : Connaître la réponse
    - Convenir d'une zone de communication où :
      - Le Système écrira sa réponse.
      - Le programme lira cette réponse.

Zone de communication ⟵ Obtenir (mémoire structurée)

Protocole d'attribution de la mémoire

- Définition : Un pointeur est une variable qui contient l'adresse d'une « autre » variable.
  - Conséquence 1 : la zone de communication est au moins constituée d'un pointeur.
  - Conséquence 2 : comme le pointeur doit être accessible en lecture et/ou en écriture par le programme donc :

    Pointeur (de zone de communication) est 1 variable du programme

- Remarque: ce pointeur peut être statique ou dynamique.

Protocole de communication

- Deux cas :
  - « Système » peut attribuer cette mémoire alors :
    - Il écrit, dans la zone de communication, l'adresse de la plage mémoire allouée au programme.
  - « Système » ne peut pas attribuer la mémoire :
    - Il écrit la valeur NULL dans la zone de communication.

Protocole de communication

- Deux cas :

Zone de communication ⟵ **Obtenir (mémoire structurée)**

Pointeur ⟵ **Obtenir (mémoire structurée)**

```
Exemple :  Algo_Exp_alloc
Début
        ptr de type pointeur sur Entier Long ;
        ptr ⟵ obtenir (mémoire pour un Entier Long)
        si   (ptr # NULL)
           écrire (« système a attribué la mémoire située à l'adresse »,  ptr)
        sinon
           écrire  (« le système n'a pas attribué de mémoire »)
fin
```

Variables dynamiques

- Inconvénients et contraintes
  - On perd l'accès direct à la variable
  - Le programmeur doit prendre à sa charge la gestion des allocations et restitutions de la mémoire qu'il veut utiliser
  - Le programmeur doit faire le gestion des exceptions.

    (Plus de liberté donc Plus de responsabilité!)

Obtentions consécutives de M.D

- Exemple

  Améliorer la fluidité de la circulation dans un carrefour en modifiant les périodes d'alternance des feux.

  - À chaque passage de véhicule, on note :
    - l'heure de son passage
    - le type du véhicule

  - Pour assurer une bonne gestion de la mémoire, on aura recours à chaque passage de véhicule à de la mémoire dynamique

Obtentions consécutives de M.D

- Exemple

Ptr : un pointeur qui contient l'adresse d'un nouvel élément la demande de M.D.

Ptr ⟵ Obtenir (mémoire pour Heure et type)

Question : Que faire lorsqu'un autre véhicule passe ?

Il faut demander à nouveau de la M.D.

Question : Quel pointeur utiliser pour formuler la demande ?

U**n pointeur autre que Ptr pour ne pas perdre l'adresse du 1er.**

Question : De combien de pointeurs aura-t-on besoin ?

Autant de pointeurs qu'il passera de voitures !

Obtentions consécutives de M.D

- Exemple

Conséquence : les pointeurs qui contiennent les adresses des élément ne peuvent pas être tous des variables statiques.

Que faire ?

Un exemple vécu :

Opérations de guichet nécessitant un imprimé:

Les accoutumés gardent toujours sur eux un imprimé en réserve, pour ne pas faire la queue deux fois !

Obtentions consécutives de M.D

- Exemple

Solution : lors de la demande de mémoire on demandera en même temps:

- De la place mémoire pour noter l'heure et le type du véhicule
- Un pointeur de réserve pour noter l'adresse de l'élément suivant: