Structure dynamique de données

Algo – Chapitre 5

I. Structure dynamique de données

1. Notions de création et de fonctionnement d'un programme

a. La compilation

La compilation permet de transformer du code humainement compréhensible vers du code machine. De plus, il permet souvent de changer de paradigme et peut rajouter du code.

b. Segments mémoire et allocation

Les entités utilisées sont placées dans la mémoire vive dans divers segments :

- **statique ou *text* :** programmes et sous-programmes
- ***bss* :** variables globales
- ***data* :** constantes
- **tas ou *heap* :** espaces alloués dynamiquement
- **pile ou *stack* :** espaces alloués statiquement

- **Allocation statique :** Allocation prévue à la compilation (variables locales, paramètres, …)
- **Allocation dynamique :** Allocation non prévue à la compilation, écrite par le programmeur.

2. Les pointeurs

Un *pointeur* **p** est une variable de type *« pointeur sur T »* noté **^T** référençant une zone mémoire permettant de stocker une information de type **T**. Un pointeur à **NIL** ne pointe sur rien.

p^ permet d'accéder à l'espace mémoire pointé par p. **@var** permet d'obtenir un pointeur vers la variable var.

Exemple :

```
p: ^Entier
i: Entier

p ← NIL
p ← @i
p^ ← 3
```

3. Allocation dynamique

p : ^T	Pseudo-code
Allouer p	`allouer(p)`
Libérer p	`liberer(p)`

4. Egalité et identité d'une liste chaînée

- **Listes égales :** même contenu
- **Listes identiques :** même adresse mémoire