Types abstraits et représentation concrète des données

1 – Définition:

Les types abstraits sont définis par leur interface, c'est à dire, comment on s'en sert, plutôt que par leur implémentation, c'est à dire, comment ils fonctionnent.

Ils permettent d'étudier des algorithmes indépendamment du langage utilisé.

Pour mieux comprendre ce qu'est un type abstrait, il faut d'abord s'intéresser a:

- La structure de données: C'est une manière de stocker les données, d'y accéder et de les manipuler comme on le ferait pour les types "list" ou "dict" de Python.
- L'interface: C'est la description de la structure de données. De quelle manière on va pouvoir la manipuler, par exemple, en utilisant .append() pour le type liste Python, ou encore .get() pour le type dict Python.
- L'implémentation: C'est la partie qui contient le code de ces méthodes, comment fait Python pour les utiliser. Il n'est pas nécessaire de connaître l'implémentation pour manipuler la structure de données.
- Un type abstrait: C'est la partie qui décrit essentiellement une interface, indépendamment du langage de programmation utilisé, avec éventuellement, des précisions sur la complexité en temps et en espace de ces opérations (se rappeler de la complexité lors du cours sur la récursivité).

Nous verrons un peu plus tard la manipulation des types de données (liste, pile et file) par des algorithmes.

Le but de réfléchir à des algorithmes est souvent, à un moment ou un autre, de "traduire" ces algorithmes dans un langage compréhensible pour un ordinateur (Python, Java, C,...).

On dit alors que l'on implémente un algorithme. Il est donc aussi nécessaire d'implémenter les types de données comme les listes, les piles ou les files afin qu'ils soient utilisables par les ordinateurs.

Les listes, les piles ou les files sont des "vues de l'esprit" présent uniquement dans la tête des informaticiens, on dit que ce sont des types abstraits de données (ou plus simplement des types abstraits).

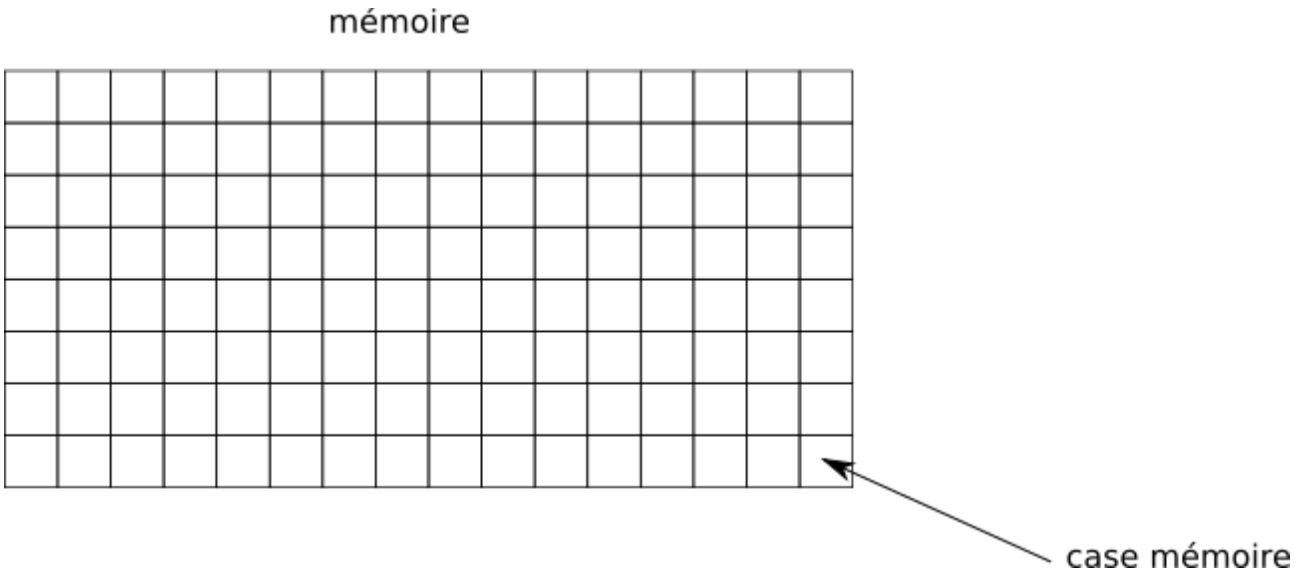
L'implémentation de ces types abstrait, afin qu'ils soient utilisables par une machine, est loin d'être une chose banal. L'implémentation d'un type de données dépend du langage de programmation. Il faut que, quel que soit le langage utilisé, le programmeur retrouve les fonctions qui ont été définies pour le type abstrait (pour les listes, les piles et les files cela correspond aux fonctions que l'on verra plus tard).

Certains types abstraits ne sont pas forcément implémentés dans un langage donné, si le programmeur veut utiliser ce type abstrait, il faudra qu'il le programme par lui-même en utilisant les "outils" fournis par son langage de programmation.

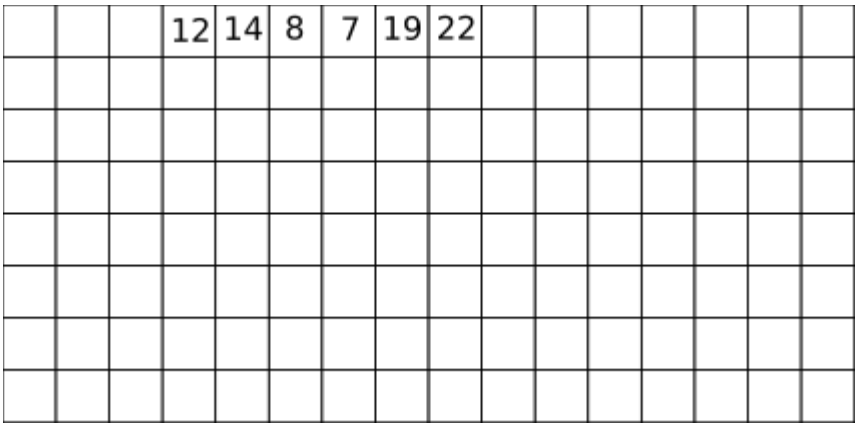
Pour implémenter les listes (ou les piles et les files), beaucoup de langages de programmation utilisent 2 structures : les tableaux et les listes chaînées.

2 – Les tableaux:

Un tableau est une suite contiguë de cases mémoires dont les adresses se suivent :



Le système réserve une plage d'adresse mémoire afin de stocker des éléments.



La taille d'un tableau est fixe : une fois que l'on a défini le nombre d'éléments que le tableau peut accueillir, il n'est pas possible modifier sa taille. Si l'on veut insérer une donnée, on doit créer un nouveau tableau plus grand et déplacer les éléments du premier tableau vers le second tout en ajoutant la donnée au bon endroit.

Dans certains langages de programmation, on trouve une version "évoluée" des tableaux : les tableaux dynamiques, qui ont une taille qui peut varier. Il est donc relativement simple d'insérer des éléments dans le tableau. Ce type de tableaux permet d'implémenter facilement le type abstrait liste (de même pour les piles et les files)

À noter que les "listes Python" (listes Python) sont des tableaux dynamiques. Attention de ne pas confondre avec le type abstrait liste que l'on verra plus tard, ce sont de "faux amis".

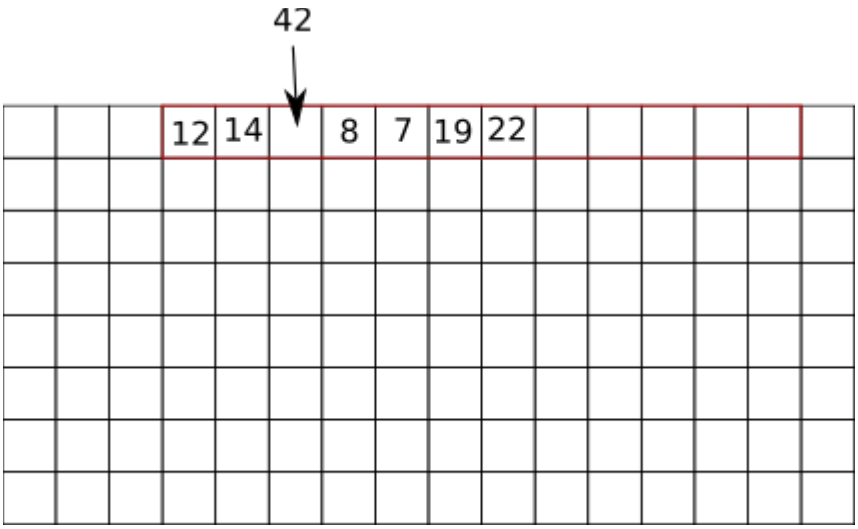
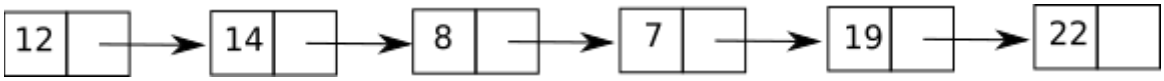


tableau dynamique

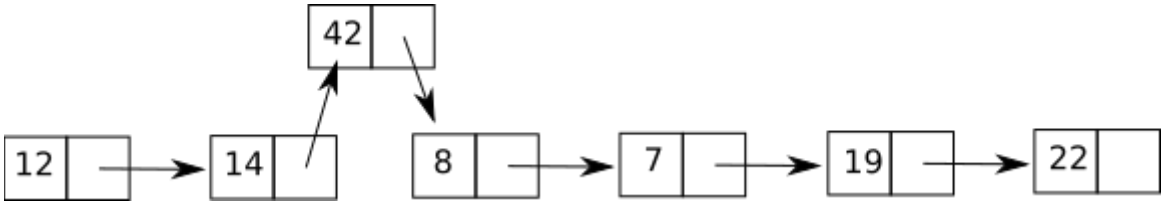
3 – Les listes chainées:

Autre type de structure que l'on rencontre souvent et qui permet d'implémenter les listes, les piles et les files : les listes chaînées.

Dans une liste chaînée, à chaque élément de la liste on associe 2 cases mémoire : la première case contient l'élément et la deuxième contient l'adresse mémoire de l'élément suivant.



Il est relativement facile d'insérer un élément dans une liste chaînée :



Il est aussi possible d'implémenter les types abstraits en utilisant des structures plus complexes que les tableaux et les listes chaînées. Par exemple, en Python, il est possible d'utiliser les tuples pour implémenter le type abstrait liste.

A faire: se renseigner sur les listes chainées.

Sources:

- https://pixees.fr/informatiquelycee/n_site/nsi_term_structDo_liste.html
- Prépabac Tle générale spécialité NSI, édition Hatier