NSI 1ère - Algorithmique - 2 - Structure de données

QK

Structures de données

2 types de personnes

- En cuisine il y a deux types de personnes :
  - les chefs écrivent les recettes
  - et les cuisiniers qui préparent les plats.

  Les chefs ne font pas la cuisine.
- En informatique c'est pareil, il y a :
  - ceux qui écrivent les algorithmes
  - et ceux qui les implémentent
- Problème : il faut se comprendre.

Se comprendre...

- Pour se comprendre on va **améliorer le langage.**
  - On définit des choses bien précises (comme en cuisine)
  - On essaie de regrouper certaines méthodes ou techniques
- En informatique :
  - Langage : variable, boucle, incrémentation…
  - Regroupement : structures de données, types abstraits…

Variable

- Une variable sert à mémoriser de l'information
- Ce qui est dans une **variable** est mis dans **une partie de la mémoire**

Type de données

Type de données : le genre de contenu d'une donnée et les opérations pouvant être effectuées sur la variable correspondante

- Par exemple :
  - **int** représente un entier. On peut faire des additions, multiplications…
  - **Date** représente une date. On peut aussi faire des additions, on peut l'écrire d'une certaine manière (jj/mm/aaaa) ou "10 septembre 2049" (ma date de naissance).
- Les types que nous rencontrerons souvent sont : int, float (~ nombre réel), booléen (V / F), chaîne etc.

Exemples en Python

Accéder au type d'un objet avec `type( )`

```
>>> n = 5; f = 3.14; b = True; nom = "Robert"
>>> type(n); type(f); type(b); type(nom)
<class 'int'>
<class 'float'>
<class 'bool'>
<class 'str'>
```

Python a un typage dynamique

Il n'est pas nécessaire de préciser le type de données.

En C ou en java, on écrirait :

```
int n = 5;
```

Dans Python il suffit d'écrire `n = 5`

- Avantage : simplicité
- Inconvénients : nombreux

Un danger du typage dynamique en Python

```
b = 5
if b == 1:
  a = 1 - "marcel" # jamais exécutée
else:
  a = "bonjour " + "marcel"
```

L'opération `1 - "marcel"` est impossible.

Pas d'erreur tant que cette instruction n'est pas **exécutée**

Structure de données

Structure de données : une manière particulière de stocker et d'organiser des données dans un ordinateur de façon à pouvoir être utilisées efficacement

Différentes structures de données existent pour résoudre des problèmes précis :

- tableaux (listes en Python)
- B-arbres dans les bases de données (BTrees en Python)
- tables de hachage dans les compilateurs (~ dictionnaires en Python)

Structures de données

- Ingrédient essentiel pour l'efficacité des algorithmes
- Permettent d'organiser la gestion des données
- Une structure de données ne regroupe pas nécessairement des objets du même type

Type abstrait de donnée

- Un **type abstrait** de données ou une **structure de données abstraite** est une spécification mathématique d'un ensemble de données et des opérations qu'elles peuvent effectuer. C'est un cahier des charges qu'une structure de données doit ensuite implémenter.

Exemple de type abstrait : la pile

Défini par sa structure et 2 opérations :

- Une pile contient des éléments.
- **Push** : insère un élément à la fin
- **Pop** : extrait le dernier élément inséré de la structure

La pile en Python

Les **listes** permettent d'effectuer les opérations des piles :

Contenir, Push et Pop :

```
>>> pil = [1, 2, 3]
>>> pil.append(4) # push en Python
>>> pil
[1, 2, 3, 4]
>>> pil.pop()
4
>>> pil
[1, 2, 3]
```

Structures de données : résumé

- Permettent de gérer et d'organiser des données
- Sont définies à partir d'un ensemble d'opérations qu'elles peuvent effectuer sur les données

Structures de données et langage à objets

- Les structures de données permettent d'organiser le code.
- une SDD correspond à une **classe** contenant 2 parties
  - Une **visible** : les opérations que la structure peut effectuer. En langage objets on parle de **partie publique**
  - Une **cachée** : les méthodes internes permettant de réaliser les opérations de la SDD. En langage objets on parle de **méthode privée**
- La partie visible de la SDD est parfois appelée **API** de la SDD :

  Application Programming Interface, autrement dit l'interface de programmation de la SDD.

Exemple en Python : la classe `str`

Extrait de la doc : *Les données textuelles en Python sont manipulées avec des objets str ou strings. Les chaînes sont des listes immuables* (on ne peut les modifier) *de caractères Unicode.* (lettres, nombres, symboles etc.)

Quelques méthodes

`find` : renvoie la première position de la sous chaîne, `-1` sinon

```
>>> 'Python'.find('yth')
1
>>> 'Py' in 'Python' # test d'appartenance
True
```

`format` : permet de formater une chaîne :

```
>>> name = "Robert"
>>> 'bonjour {}'.format(name)
'bonjour Robert'
```

Méthodes publiques, privées

Les méthodes présentées ci-dessus sont toutes **publiques**

Pour connaître les méthodes privées il faut consulter le code source de Python

La deuxième phrase de la doc nous dit : **ce sont des listes !**

Voilà pourquoi on peut tester `'Py' in 'Python'` (on en reparlera plus tard)