Structure générale d'un algorithme

```
#!/usr/bin/python
# coding= latin1

# Bloc de déclaration des variables
# Programme principal
```

Déclaration des variables

Attention :

- *les variables ne sont pas typées ;*
- *c'est l'affectation d'une valeur qui définit la variable.*

Types

Pas de type pour une variable en Python...

On peut néanmoins demander le type d'une valeur à l'aide de la fonction `type()` :

```
>>> type("a")
<type 'str'>
>>> type(10)
<type 'int'>
```

Commentaires

Il vont du caractère **#** jusqu'à la fin de la ligne :

```
# Ceci est un commentaire
```

Séquence

Pour définir, un « bloc d'instruction », on ***indente*** *les instructions qui en font partie :*

```
instruction 1
instruction 2
 instruction a
 instruction b
 instruction c
```

Action de lecture/écriture

```
ma_chaine = raw_input("Entrez une chaîne de caractères")
print "le contenu est ", ma_chaine
mon_entier = int(raw_input("Entrez un entier")
mon_flottant = float(raw_input("Entrez un flottant")
print "Mon entier vaut ",mon_entier," et mon flottant vaut ",mon_flottant
```

On utilise la fonction `raw_input()` qui renvoie toujours une chaîne de caractère.

Python v3

On utilisera `input()` au lieu de `raw_input()` qui a été supprimé.

Affectation

```
1 ma_variable = 10
```

Action conditionnelle

```
1 if (condition1) :
2  instruction A
3  instruction B
4 else :
5  instruction 1
6  instruction 2
```

Ne pas oubliez les « : »

Choix multiples

```
1 if x == valeur1:
2  instruction 1
3 elif x == valeur2:
4  instruction 2
5 elif x == valeur3:
6  instruction 3
7 else:
8  instruction 4
```

Ne pas oubliez les « : »

Actions itératives

```
1 for i in range(indice_debut,indice_fin_non_compris):
2  instruction1
3  instruction2
```

```
1 while (condition):
2  instruction1
3  instruction2
```

Ne pas oubliez les « : »

Sous-programmes

```
1 def ma_fonction():
2  instruction1
3  instruction2
4  return valeur
```

```
1 def ma_fonction(parametre1, parametre2):
2  instruction1
3  instruction2
4  return valeur
```

Fichiers

```
1 mon_fichier = open("fichier.txt","r")
2 for une_ligne in mon_fichier: # lecture ligne par ligne
3  print une_ligne  # sous forme de chaîne de caractères
4 mon_fichier.close()
```

Pour se déplacer à une position précise dans le fichier : `mon_fichier.seek(decalage,reference)` où

- `reference`, prend la valeur 0 pour « depuis le début du fichier », 1 « en relatif » et 1 pour « depuis la fin ».
- `decalage` indique le nombre d'octets.

On ne lit et on n'écrit dans un fichier que des types simples, ***mais pas de structure*** *(voir support Python pour connaître la solution p.34).*

Structures

Une solution dégradée :

```
1 class ma_structure:
2  pass
3 ma_variable = ma_structure() # on crée une structure vide
4 ma_variable.x = 15.2 # on remplit la structure
5 ma_variable.y = 23.6 # on remplit la structure
6 ma_variable.nom = "A" # on remplit la structure
7 print "Le point ", ma_variable.nom, " a pour coordonnees (", ma_variable.x, ";", ma_variable.y,
")"
```

Pour une version améliorée, voir p.54 du support Python.

Vecteurs et tableaux

Ils n'existent pas sous Python.

On utilise des *listes*.

```
1 mon_tableau = [0] * 10 # Crée une liste de 10 cases contenant chacune 0
2 mon_tableau[2] = 42
3 print mon_tableau
```

Ce qui donne :

```
[0, 0, 42, 0, 0, 0, 0, 0, 0, 0]
```

Pour deux dimensions :

```
1  nb_lignes = 5
2  nb_colonnes = 4
3  tableau2D = []
4  for i in range(0, nb_lignes):
5   tableau2D.append([])
6   for j in range(0, nb_colonnes):
7    tableau2D[i].append(0)
8  tableau2D[2][3] = 'a'
9  tableau2D[4][3] = 'b'
10 print tableau2D
```

Ce qui donne :

```
[[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 'a'], [0, 0, 0, 0], [0, 0, 0, 'b']]
```

Opérations diverses

- modulo : `%`
- division entière : `//`
- pour des valeurs aléatoires :

```
1 import random
2 alea = random.random() # renvoi une valeur entre 0 et 1
```

Pointeurs

Toutes les données sont manipulées par référence.

Si on veut connaître l'adresse d'une valeur, on peut utiliser la fonction `id()`.

Pour savoir si deux variables référencent la même valeur, on utilise l'opérateur `is` pour les comparer.