

ALGORITHMIQUE

↔ Instructions de base et types ↔

I. Introduction

Intro

Lorsque l'on veut faire effectuer quelque chose à une machine, il faut **découper** le problème en tâches simples organisées logiquement. C'est la présentation de ces tâches dans l'ordre où elles doivent être exécutées que l'on appelle un **algorithme**.

Une fois ce travail accompli, il faut **traduire** le tout dans un langage compréhensible par la machine, en tenant compte des règles propres à chaque langage.

Ainsi, l'écriture de l'algorithme permet d'évacuer dans un premier temps les problèmes spécifiques des langages pour se concentrer uniquement sur l'analyse de la tâche à accomplir.

Méthode

D'une façon générale on peut dire qu'un algorithme se présentera en trois parties :

- *Préparation*
- *Traitement*
- *Édition*

La partie demandant le plus de travail étant bien sûr le ALGO traitement. La conception d'un algorithme se déroulera grosso modo en deux étapes :

- compréhension et analyse du problème ;
- écriture de l'algorithme en « ALGO pseudo-langage ».

Remarque

Pour être compréhensible par d'autres, réutilisable et modifiable un algorithme devra donc être accompagné de ALGO commentaires.

Méthode - Analyse d'un problème

Avant de pouvoir se lancer dans l'écriture d'un programme, il est nécessaire de *comprendre parfaitement* le (ou les) problème(s) qu'il doit résoudre. Le *programmeur* doit bien savoir ce que l'*utilisateur* attend.

Il faudra :

- définir d'abord les résultats que votre programme devra fournir (les ALGO sorties), avant d'avancer plus ;
- ALGO décomposer le problème jusqu'à ce que l'on puisse le réduire à des tâches que la machine peut exécuter, et organiser ces tâches (c'est l'algorithme).

Cette démarche s'appelle l'ALGO analyse descendante.

II. Variables et constantes

II.1. Définitions

Définition

Un programme va manipuler des éléments de base : ALGO variables et ALGO constantes.

De manière très schématique :

- les constantes sont les valeurs **fixes** tout au long de l'algorithme ;
- les variables peuvent être amenées à voir leur valeur **évoluer** au long de l'algorithme.

Vocabulaire

Chaque élément de base (variable ou constante) a trois qualités : le nom, le type, la valeur.


- Le **nom**, ou identificateur : permet de repérer et réserver un emplacement de la mémoire de l'ordinateur afin de lire ou modifier (la variable). Utiliser des noms explicites pour faciliter la compréhension.
- Le **type** : désigne l'ensemble dans lequel la variable est prise. (Nombre entier, nombre réel, caractère ou chaîne de caractères, booléen)
À chaque type correspondent des limitations à l'utilisation de la variable, seules certaines fonctions ou opérations seront permises avec des variables de type donné.
- La **valeur** : on ne peut pas modifier la valeur d'une constante, fixée dans la partie « préparation » de l'algorithme. Mais lors de l'exécution de l'algorithme, les valeurs de certaines variables seront modifiées.

II.2. Sous Python

Point histoire

python est un langage de programmation interprété, multi-paradigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramasse-miettes et d'un système de gestion d'exceptions; il est ainsi similaire à Perl, Ruby, Scheme, Smalltalk et Tcl.



Guido van Rossum (1956-, ) est un développeur connu pour être le créateur et leader du projet du langage de programmation python. Il est également l'auteur du navigateur web Grail entièrement programmé en python; il n'a pas été actualisé depuis 1999. Le nom fait référence au film *Monty Python and the Holy Grail*

Méthode

Les noms de variables doivent obéir à quelques règles simples :

- Un nom de variable est une séquence de lettres (**a** à **z**, **A** à **Z**) et de chiffres (**0** à **9**), qui doit toujours commencer par une lettre.
- Seules les lettres ordinaires sont autorisées. Les lettres accentuées, les cédilles, les espaces, les caractères spéciaux tels que **\$**, **#**, **@**, etc. sont interdits, à l'exception du caractère **_** (souligne).
- La casse est significative (les caractères majuscules et minuscules sont distingués).

Ainsi : **Joseph**, **joseph**, **JOSEPH** sont donc des variables différentes.

On écrit généralement l'essentiel des noms de variables en caractères minuscules (y compris la première lettre). Il s'agit d'une simple convention, mais elle est largement respectée.

On n'utilise les majuscules qu'à l'intérieur même du nom, pour en augmenter éventuellement la lisibilité, comme dans **tableDesMatières**.

En plus de ces règles, il faut encore ajouter que l'on ne peut pas utiliser comme nom de variables les 33 « mots réservés » ci-dessous (ils sont utilisés par le langage lui-même) :

and **as** **assert** **break** **class** **continue** **def** **del** **elif** **else** **except**
False **finally** **for** **from** **global** **if** **import** **in** **is** **lambda** **None** **nonlocal**
not **or** **pass** **raise** **return** **True** **try** **while** **with** **yield**

II.3. Typage

Propriété

Les **opérations** que l'on pourra effectuer sur les variables dépendront de leur *type*.

Définition - Le type entier

Ce sont les entiers, positifs et négatifs. Les opérateurs sur les entiers sont :

$\boxed{+}$ (addition); $\boxed{-}$ (soustraction); $\boxed{*}$ (multiplication); $\boxed{**}$ (élévation à une puissance);
 $\boxed{/}$ (division : quotient réel); $\boxed{//}$ (division : quotient entier); $\boxed{\%}$ (reste de la division entre entiers);
 $\boxed{==}$ (égale); $\boxed{!=}$ (différent); $\boxed{<}$ (inférieur); $\boxed{<=}$ (inférieur ou égal); $\boxed{>}$ (supérieur); $\boxed{>=}$ (supérieur ou égal).

Définition - Le type réel

Les opérateurs sur les réels sont : $\boxed{+}$; $\boxed{-}$; $\boxed{*}$; $\boxed{**}$; $\boxed{/}$; $\boxed{//}$; $\boxed{==}$; $\boxed{!=}$; $\boxed{<}$; $\boxed{<=}$; $\boxed{>}$; $\boxed{>=}$.

Définition - Le type chaîne de caractères

Un opérateur sur les chaînes de caractères : $\boxed{+}$ (concaténation); $\boxed{*}$ (concaténation répétée).

Définition - Le type booléen

Les booléens sont des variables logiques : ils peuvent prendre deux valeurs : $\boxed{\text{VRAI}}$ ou $\boxed{\text{FAUX}}$.

Les opérateurs sur les booléens sont : $\boxed{\text{ET}}$, $\boxed{\text{OU}}$, $\boxed{\text{NON}}$, $\boxed{==}$, $\boxed{!=}$.

Méthode

En  python, les opérateurs définis précédemment sont :

- $\boxed{+}$; $\boxed{-}$; $\boxed{*}$; $\boxed{**}$; $\boxed{/}$; $\boxed{//}$; $\boxed{==}$; $\boxed{!=}$; $\boxed{<}$; $\boxed{<=}$; $\boxed{>}$; $\boxed{>=}$;
- $\boxed{\text{True}}$; $\boxed{\text{False}}$; $\boxed{\text{and}}$; $\boxed{\text{or}}$; $\boxed{\text{not}}$.

Méthode

Sous  python, le type de la variable est donné par le type de la valeur qu'on lui affecte.

On peut changer de type (on parle de $\boxed{\text{transtypage}}$) à l'aide de fonctions :

- Type **entier** en Python : $\boxed{\text{int}}$.
- Type **réel** en Python : $\boxed{\text{float}}$ (Le séparateur décimal est un point!).
- Type **chaîne** en Python : $\boxed{\text{str}}$.
- Type **booléen** en Python : $\boxed{\text{bool}}$.

III. Entrées, sorties, affectation

III.1. Entrées : lecture ou saisie

Définition

La $\boxed{\text{Saisie}}$ consiste pour l'ordinateur à « lire » une donnée (par exemple tapée par l'utilisateur sur le clavier), et à ranger cette valeur dans une variable.

L'instruction $\boxed{\text{Saisir}}$ doit donc obligatoirement faire apparaître un nom de variable.

À l'exécution du programme, il y aura *interruption* jusqu'à ce que l'utilisateur ait tapé un nombre au clavier.

De même que pour la $\boxed{\text{Déclaration}}$, on peut faire saisir plusieurs variables dans la même instruction.

III.2. Sorties : écriture (ou affichage)



Définition

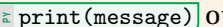
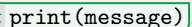
A l'inverse de la saisie où l'utilisateur donne des informations à la machine, l'instruction $\boxed{\text{Afficher}}$ permet à la machine de communiquer des résultats à l'utilisateur.



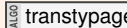


$\boxed{\text{Afficher}}$ est suivi du nom d'une variable ou d'une constante, et sa valeur est alors donnée par la machine.

III.3. En Python

Méthodes

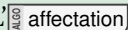
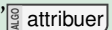
La transcription en  python de «  » sera, en fonction du type de commande :

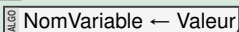
- ou ou ;
-  ou .


Le message figurant entre les parenthèses de l'instruction  est facultatif mais elles doivent apparaître. Il est à noter que la fonction  renvoie toujours en Python un type chaîne de caractères d'où la nécessité d'effectuer un  lorsque l'on veut saisir des nombres ( : entier,  : réel).

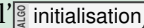
III.4. Affectation

Définition

L' est l'action qui permet d' une valeur à une variable.


Elle est symbolisée par \leftarrow selon la syntaxe : .


Cette instruction signifie que la quantité située à droite de la flèche est évaluée puis stockée dans la variable , remplaçant ainsi son ancien contenu.

L'instruction qui stocke une **première** valeur dans une variable est l'.

À gauche doit apparaître le nom de la variable à modifier, et à droite une expression ayant une valeur de même type que la variable. La valeur de la variable de gauche est modifiée, contrairement à celle de droite qui est évaluée mais non modifiée. L'affectation n'est *pas symétrique*!

Méthode




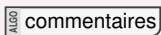


En  python, l'affectation est symbolisée par le signe .

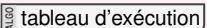
Dans les algorithmes, on gardera toujours la notation  afin d'éviter toute confusion.

IV. Conclusion

Bilan

L'écriture d'un algorithme obéit à des règles précises et doit comporter :

- un  (répondant aux mêmes contraintes que les noms de variables) ;
- son  (un commentaire décrivant ce que fait l'algorithme et éventuellement sa façon de procéder) ;
- la  des variables utilisées (noms et types) ;
- un début et une fin encadrant la suite d'instructions constituant l'algorithme ;
- des  afin d'être facilement lisible et compréhensible (on signalera un commentaire en  par le symbole  ; tout ce qui suit ce symbole sur la même ligne est ignoré par la machine).

On peut faire un  pour « dérouler » l'algorithme et analyser l'évolution des variables.

Algorithme

Un algorithme en  se présente donc sous la forme suivante :

Pseudo-Code

Algorithme :

Rôle :

Variables :

Début

.....

.....

Fin