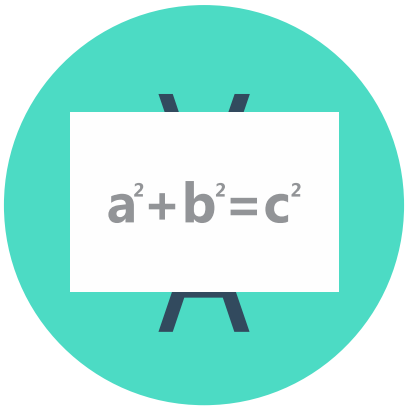

Chapitre VIII – Algorithmique

Bacomathiques — https://bacomathiqu.es

TABLE DES MATIÈRES

I - Définition

Un algorithme est une suite finie et ordonnée d'opérations ou d'instructions permettant de résoudre un problème ou d'obtenir un résultat. Ainsi, faire une recette de cuisine ou encore effectuer une division euclidienne à la main sont des exemples d'algorithmes.

Dans ce cours, nous travaillerons à la fois avec des algorithmes Python et des algorithmes en pseudo-code.

II - Instructions

1. Création de variables

Créer une variable permet de réserver un espace pour y stocker des données quelconques.
On donne un nom à chaque espace pour le repérer : ce sont les noms de variables. Dans certains langages, on leur donne également un type (entier, réel, ...) pour travailler avec (ce qui n'est pas le cas dans Python).

À RETENIR

En python

```
nombre = 0 # On crée la variable "nombre" et on lui assigne la valeur
    0.
chaine = 'Bonjour' # On crée la variable "chaine" et on lui assigne
   la valeur 'Bonjour'.
```

À LIRE

Exemple (tiré du sujet de Pondichéry 2017)

Variables
  R et S sont des réels
  n et k sont des entiers
Traitement
  S prend la valeur 0
  Demander la valeur de n
  Pour k variant de 1 à n faire
    R prend la valeur $\frac{2,5}{n} \times f\left(\frac{2,5}{n} \times k\right)$
    S prend la valeur $S + R$
  Fin Pour
  Afficher S

Ici nous avons quatre variables : R et S qui sont des réels et n et k qui sont des entiers.

2. Affectations de valeurs

Comme dit précédemment, les variables sont des "espaces" dans lequel il est possible de stocker des informations.

Cependant, après avoir créé cet espace, celui-ci est encore vide. C'est pourquoi on doit le "remplir" : c'est **l'affectation d'une valeur à une variable**.

Il existe plusieurs manières d'affecter une valeur à une variable : soit on lui donne directement sa valeur dans l'algorithme, soit on demande à l'utilisateur d'entrer une valeur (il faut garder à l'esprit que nos algorithmes sont faits pour être utilisés par des utilisateurs).

À RETENIR

En python

```
x = int(input('Veuillez entrer une valeur : ')) # L'utilisateur va
    entrer une valeur, on la convertir en entier et on va affecter
    celui-ci à notre variable "x".
y = 2*x + 10 # Une fois fait, "y" va prendre la valeur 2 * x + 10.
    Par exemple, si l'utilisateur entre "10", "y" vaudra 30.
```

À LIRE

Exemple (tiré du sujet de Pondichéry 2017)

Variables
- R et S sont des réels
- n et k sont des entiers

Traitement
- S prend la valeur 0
- Demander la valeur de n
- Pour k variant de 1 à n faire
  - R prend la valeur $\frac{2,5}{n} \times f\left(\frac{2,5}{n} \times k\right)$
  - S prend la valeur $S + R$
- Fin Pour
- Afficher S

Ici on donne à S la valeur 0, mais on demande à l'utilisateur d'entrer la valeur de la variable n (l'utilisateur entrera un entier, car la variable n ne peut contenir que des entiers).

Une fois que l'on a affecté une valeur à une variable, il est encore possible de la changer!

3. Affichage de variables

Nos algorithmes étant faits pour être utilisés, il faut donc **retourner un résultat** sinon ceux-ci seraient inutiles. C'est pourquoi, on peut "afficher" les valeurs des variables (les montrer à l'utilisateur).

À RETENIR

En python

```
print('Voici la valeur de "maVariable" :', maVariable) # Permet d'
    afficher la valeur de "maVariable".
```

À LIRE

Exemple (tiré du sujet de Métropole 2017)

Variables	N et A des entiers naturels
Entrée	Saisir la valeur de A
Traitement	N prend la valeur 0 Tant que $N - \ln(N^2+1) < A$ N prend la valeur $N+1$ Fin tant que
Sortie	Afficher N

Une fois l'algorithme terminé, on affiche la valeur de la variable N (on remarque que N a pris plusieurs valeurs différentes au cours de l'algorithme mais qu'on affiche uniquement la valeur finale de la variable).

III - Blocs d'instructions

1. Définition

Les blocs d'instructions sont des parties de l'algorithme (ce sont des "algorithmes dans l'algorithme") qui s'exécutent suivant certaines conditions propres aux différents blocs d'instructions.

2. Les blocs SI et SINON

Les blocs **SI** et **SINON** sont des blocs d'instructions très utilisés qui permettent de tester une condition : si elle est réalisée, on va exécuter les instructions se situant sous le bloc SI et sinon, on va exécuter celles se situant sous le bloc SINON.

À RETENIR

En python

```
x = 2 # On attribue à "x" la valeur 2.
if x == 3: # Si "x" est égal à 3...
  print('"x" est égal à 3.') # ... Alors on affiche ce message. Mais
   ici, "x" vaut 2 donc ce message ne sera jamais affiché.
else: # Sinon...
  print('"x" n\'est pas égal à 3.') # ... On affiche ce message.
```

À LIRE

Exemple (test de parité)

Variables	N et R sont des entiers
Entrée	Saisir la valeur de N
Traitement	SI E(N/2) = N/2 : R = 0 SINON : R = 1
Sortie	Afficher R

Si la partie entière de $\frac{N}{2}$ est égale à $\frac{N}{2}$ (ce n'est vrai que pour les entiers pairs), alors on donne à R la valeur 0. Sinon on lui donne la valeur 1.
En fin d'algorithme, on affiche la valeur de R : soit 0 si N est pair, soit 1 si N est impair.

3. La boucle POUR

La **boucle POUR** est un bloc d'instruction qui s'exécute et qui va faire prendre à une variable toutes les valeurs comprises dans un ensemble d'entiers.

À RETENIR

En python

```
for i in range(-5, 6): # Pour chaque entier entre -5 (inclus) et 6 (
    exclu)...
  print(i) # ... On affiche cet entier.
```

À LIRE

Exemple (calcul des termes d'une suite)

Variables	U est un réel N et n sont des entiers
Entrée	Saisir la valeur de N
Traitement	Affecter à u la valeur 1 Pour n allant de 1 à N : Affecter à u la valeur u + 1/n
Sortie	Afficher n Afficher u

Cet algorithme permet de calculer les termes d'une suite $(u_n)_{n\in\mathbb{N}}$ définie par récurrence :

$\begin{cases} u_0 = 1 \\ u_{n+1} = u_n + \frac{1}{n} \end{cases}$ pour n entier.

On demande à l'utilisateur d'entrer une variable N, et pour n variant de 1 jusqu'à N (n prendra tour à tour les valeurs 1, 2, 3, ..., $N-1$, N), on va calculer les termes de la suite.

4. La boucle TANT QUE

Cette boucle, différente de la boucle POUR, permet d'exécuter son bloc d'instructions tant qu'une certaine condition est valable.

À RETENIR

En python

```
x = 100 # On affecte à "x" la valeur 100.
while x > 10: # Tant que x est supérieur à 10...
  x = x / 2 # On divise x par 2 (i.e. on affecte à "x" la valeur x/2)
   .
  print(x) # On affiche la valeur de "x".
```

À LIRE

Exemple (tiré du sujet de Métropole 2017)

Variables	N et A des entiers naturels
Entrée	Saisir la valeur de A
Traitement	N prend la valeur 0 Tant que $N - \ln(N^2+1) < A$ N prend la valeur $N+1$ Fin tant que
Sortie	Afficher N

Ici, tant que $N - \ln(N^2+1)$ est inférieur à A, on affecte une nouvelle valeur à la variable N.

IV - Algorithmes sur l'ordinateur

Il est possible de tester et de vous entraîner aux algorithmes sur votre ordinateur, voire directement sur votre smartphone!

À LIRE

Quelques logiciels d'algorithmique

Divers logiciels à télécharger, dont certains ne nécessitant pas d'installation sont disponibles :

- Python
- Algogo
- AlgoBox
- Scratch
- Proglab