Année 2014-2015
Cité scolaire Paul Valéry
Mathématiques - F. Gaunard
`http://frederic.gaunard.com`

INTRODUCTION À L'ALGORITHMIQUE

Dans le but d'apprendre à *programmer* des instructions sur une machine (ordinateur, calculatrice...), il est au préalable important de donner une structure à cet ensemble d'instructions et nous commencerons donc par nous familiariser avec un peu de "théorie" via la notion d'*algorithme*.

---

1. Notion d'algorithme

Définition 1.1. Un **algorithme** est une suite d'*instructions détaillées* qui, si elles sont correctement exécutées, conduit à un résultat donné.

Une recette de cuisine ou une notice de montage d'un meuble peuvent être par exemple considérées comme des algorithmes. La suite d'instructions suivante:

1. `choisir un nombre`
2. `le multiplier par lui même`
3. `afficher le résultat obtenu`

est un algorithme permettant d'obtenir le carré d'un nombre.

Remarque 1.2. Dans la définition précédente, "détaillées" signifie que les instructions sont suffisamment précises pour pouvoir être mises en oeuvre correctement par l'exécutant (homme ou machine).

2. Pseudo-Code

Les instructions doivent être formulées dans un *langage compréhensible* par l'exécutant. Dans le cas d'un humain, il s'agira du langage courant (langue maternelle); dans le cas d'une machine, il faudra recourir à un langage de *programmation* (comme par exemple C, Java, Python...).

En algorithmique, nous utiliserons un langage situé à mi-chemin entre le langage courant et un langage de programmation appelé *pseudo-code*. Il n'y a pas de norme concernant ce pseudo-code qui peut varier légèrement d'un enseignant à l'autre. Le but est surtout de mettre l'accent sur la *logique de l'algorithme*. L'avantage du pseudo-code est qu'il permet de rester proche d'un langage informatique sans qu'il soit nécessaire de connaître toutes les règles et spécificités d'un langage particulier.

3. Variables

Un algorithme (ou un programme informatique), agit sur des nombres, des textes, ... Ces différents éléments sont stockés dans des *variables*. On peut se représenter une variable comme une boîte portant une étiquette ("le nom de la variable") à l'intérieur de laquelle on peut placer un contenu.

En informatique, les variables sont des emplacements réservés dans la mémoire de l'ordinateur auxquels on attribue une étiquette.

Définition 3.1. Déclarer une variable c'est indiquer le nom et le type (nombre texte, tableau,...) d'une variable que l'on utilisera dans l'algorithme. La déclaration des variables se fait au début de l'algorithme avant la première instruction.

Les principaux types de variables que nous utiliserons seront : entier, nombre (=réel), texte (=chaîne de caractères), tableau de nombres ou de textes.

Exemple 3.2. Dans notre pseudo-code, on déclarera les variables de la façon suivante:

```
variables
   x : nombre
   y : texte
   a, b, c : entiers
```

On définit 5 variables : `x` du type nombre (réel), `y` du type texte, et `a,b,c` de type entier.) Nous distinguerons la déclaration des variables en plaçant le reste de l'algorithme entre les instructions "début algorithme" et "fin algorithme".

Définition 3.3. Affecter une variable, c'est attribuer une valeur à cette variable. Si la variable contenait déjà une valeur, cette ancienne valeur est effacée.

Dans notre pseudo code, on écrira parfois "`x prend la valeur` 3" mais on utilisera aussi souvent le symbole ←. On écrira donc par exemple `x` ← 3.

4. Entrée-Sortie

Faire effectuer un calcul à une machine c'est bien... Mais il faut au moins être capable d'entrer des valeurs et il faut aussi que la machine puisse afficher un résultat! Les instructions qui permettent de "dialoguer" avec une machine s'appellent les instructions "d'entrée/sortie" ou de "lecture/écriture"

4.1. **Lecture.** Dans notre pseudo-code nous utiliserons l'instruction *Lire* suivie du nom d'une variable pour pouvoir saisir une valeur (en anglais cette instruction se nomme généralement *input*).

Lorsqu'elle rencontre une telle instruction, la machine **s'arrête et attend** que l'utilisateur entre une valeur. Une fois la valeur saisie, la machine affecte la valeur saisie à la variable dont le nom figure après *lire*. Ensuite, elle passe à l'instruction suivante.

Exemple 4.1. L'algorithme suivant demande d'entrer un nombre entier, stocke la valeur de ce nombre dans la variable `x`, puis calcule le double du nombre entré et affecte ce double à la variable y. Le résultat n'est pas affiché (d'où le paragraphe suivant...)

```
variables
   x : entier
début algorithme
   lire x
   y ← 2*x
fin algorithme
```

4.2. **Ecriture.** Dans notre pseudo-code nous utiliserons l'instruction *Afficher* suivie du nom d'une variable ou d'une constante (nombre, texte ...) pour afficher une valeur (on peut également utiliser "écrire" ou "print" en anglais).

Il est fréquent d'afficher un texte pour donner des consignes ou des informations à l'utilisateur. Pour afficher le contenu d'une variable on fait suivre "afficher" du nom de la variable sans guillemet.

Exemple 4.2. L'algorithme suivant calcule puis affiche l'âge qu'aura une personne en 2100 :

```
variables
   année, âge, âge_en_2100 : entiers
début algorithme
   afficher "Entrer l'année actuelle"
   lire année
   afficher "Entrer votre âge"
   lire âge
   âge_en_2100 ← âge + 2100 - année
   afficher "En 2100, vous aurez ", âge_en_2100, " ans."
fin algorithme
```