FICHE N°2 : LES VARIABLES

A) Qu'est-ce qu'une variable pour Python ?

Définition : Pour un langage de programmation comme Python, une **variable** est un espace de stockage dans un emplacement de la mémoire de l'ordinateur. Elle possède plusieurs caractéristiques dont en particulier :

- un **nom** : permet d'identifier la variable ;
- une **valeur** : utilisée par le programme, peut éventuellement changer ;
- un **type** : définit quelle donnée peut contenir la variable ;
- une **adresse** : l'endroit de l'ordinateur où est stockée la variable.

En mathématiques, dans les programmes écrits avec Python, la plupart du temps, on ne s'intéresse qu'au nom de la variable et à sa valeur. Pour les programmes simples, Python gère tout seul les types de variables mais il est quand même bien de savoir avec quel type on travaille.

Exemple n°1 : On souhaite écrire un programme qui calcule le prix qu'il faut payer pour faire le plein d'essence en fonction du prix au litre.
Dans ce programme, il faudra, entre autres, stocker quelque part le prix au litre. On va donc créer une variable avec les valeurs suivantes :

- nom : prix ;
- valeur : 1.52 (en informatique, les nombres décimaux s'écrivent avec des points à la place de la virgule) ;
- type : flottant (voir partie B) ;
- adresse : l'ordinateur gère tout seul cette donnée-là.

B) Les différents types des variables

En classe de seconde, il faut connaître les types de variables suivants :

- **entier :** les valeurs de la variable sont des entiers relatifs ; cela inclut donc les entiers négatifs et les entiers positifs (et donc aussi 0).
- **flottant :** les valeurs de la variable sont des nombres décimaux : ils s'écrivent avec un point (à la place de la virgule). Par exemple, la valeur 3.0 sera considérée de type *flottant* et non pas de type *entier*.
- **chaîne de caractères :** les valeurs de la variable sont des mots (ou, de façon générale, des ensembles de lettres). Avec Python, on les écrit **obligatoirement** entre guillemets. Par exemple, "Bonjour" ou "SdldFkq".
- **booléen :** les variables de type *booléen* ne prennent que deux valeurs : vrai (True) ou faux (False).

Exemple n°2 : Voici quelques valeurs de variables triées par type :
- entier : 2 ; –8 ; 12 345 ; –9 887
- flottant : 1.0 ; 3.14 ; –103.435
- chaîne : “Bonjour” ; “Comment vous appelez-vous ?” ; “Au revoir !”
- booléen : True ; False (il n’y a pas d’autres possibilités).

C) Affectation d’une variable

Définition : Avec Python, **affecter une variable** signifie lui attribuer un nom et lui associer une valeur initiale. Le type est automatiquement établi par Python et l’adresse est gérée par l’ordinateur.

Affecter une variable	
Dans un algorithme	Dans un programme Python
A ← 50 prix ← 1.52	`A = 50` `prix = 1.52`

Ici, nous avons affecté deux variables : la première s’appelle A et sa valeur, de type *entier*, est 50. La deuxième s’appelle prix et sa valeur, de type *flottant* est 1.52.
On dira : « A prend la valeur 50 » et « prix prend la valeur 1.52 ».
On remarque qu’on n’a pas eu besoin de spécifier ni le type, ni l’adresse.
Il important de retenir que les majuscules et les minuscules sont différenciées. La variable prix n’est donc pas la même que la variable Prix.

Point commun à retenir :
Le nom de la variable apparaît à gauche et sa valeur à droite.

Différence à retenir :
Dans un algorithme, on utilise le symbole ← pour affecter une variable. Dans un programme Python, on utilise le symbole =. Attention, le symbole = n’a pas le même sens ici que dans une égalité mathématique (voir Fiche n°3 bis).

Bonne pratique pour nommer une variable : Le nom d’une variable commence préférentiellement par une lettre minuscule. Généralement, le nom est explicite pour bien comprendre le programme. On peut utiliser des majuscules au milieu

du nom ou bien utiliser le symbole _ pour gagner en lisibilité. Il faut éviter les lettres accentuées mais on peut utiliser des chiffres.

Exemple n° 3 : On souhaite écrire un programme qui, connaissant le prix au litre et la contenance du réservoir, calcule le prix total pour faire le plein d'essence.

Dans un algorithme

```
contenanceReservoir ← 50
crix_au_Litre ← 1.52
total ← contenanceReservoir × prix_au_Litre
```

Dans un programme Python

```
1 contenanceReservoir = 50
2 crix_au_Litre = 1.52
3 total = contenanceReservoir * prix_au_Litre
```

Reprenons étape par étape :

- **Ligne 1 :** Affectation de la variable qui porte le nom contenanceReservoir et qui contient la valeur 50. Cette variable est de type *entier*.
- **Ligne 2 :** Affectation de la variable qui porte le nom prix_au_Litre et qui contient la valeur 1.52. Cette variable est de type *flottant*.
- **Ligne 3 :** Affectation de la variable qui porte le nom total. La valeur de cette variable est le produit de la valeur de contenanceReservoir par la valeur de prix_au_Litre. Bien que le résultat donne 76, la valeur stockée est 76.0. Cette variable est donc de type *flottant* (comme prix_au_Litre).

À retenir : Les variables de type *entier* et de type *flottant* sont compatibles pour les opérations arithmétiques. Le résultat d'une opération entre ces deux types de variables est stocké dans une variable de type *flottant*.

D) Afficher la valeur d'une variable

Le programme de l'exemple n°3 fonctionne. Cependant, il n'affiche rien. Si l'on souhaite afficher la valeur d'une variable, il suffit d'utiliser la syntaxe suivante.

Afficher la valeur d'une variable	
Dans un algorithme	Dans un programme Python
Afficher total	`print(total)`

Exemple n°3 bis : Voilà donc le programme complet répondant au problème posé. Pour calculer d'autres valeurs, il suffit de changer la valeur des variables contenanceReservoir et prix_au_Litre et de relancer le programme.

Dans un algorithme

```
contenanceReservoir ← 50
prix_au_Litre ← 1.52
total ← contenanceReservoir × prix_au_Litre
Afficher total
```

Dans un programme Python

```
1 contenanceReservoir = 50
2 crix_au_Litre = 1.52
3 total = contenanceReservoir * prix_au_Litre
4 print(total)
```

E) Opérations courantes avec les variables dans Python

Les tableaux suivants regroupent quelques opérations utiles que l'on pourra utiliser avec les variables manipulées dans des programmes écrits avec Python.
Les propriétés opératoires connues sur les réels (parenthèses, opposé d'un nombre, multiplication prioritaire sur l'addition, etc.) restent valables avec Python.

Variables de type *entier* ou *flottant* (mélange possible)		
Opérations	**Symboles**	**Exemples**

Opérations arithmétiques :
- addition
- soustraction
- multiplication
- division décimale
	
	+
	-
	*
	/
		`print(3 + 4)`
		`print(5 - 6.7)`
		`print(12*(-4))`
		`print(81/10)`

`print(5 - 6.7)` est un exemple simple pour montrer les limites de Python avec les flottants.		
Puissance	**	`print(5**2)` `print(2**10)`

Racine carrée :
- uniquement pour les nombres positifs
- nécessite la librairie *math*
	sqrt
	(pour *square root*)
		`from math import*`
		`print(sqrt(12))`
		`print(3 - sqrt(6/5))`

Variables de type *entier*		
Opérations	**Symboles**	**Exemples**
Reste de la division euclidienne de a par b	%	`print(81%10)`
Quotient de la division euclidienne de a par b	//	`print(81//10)`
Explication : Puisque $81 = 10 \times 8 + 1$ alors le reste de la division euclidienne de 81 par 10 est 1 et le quotient de cette division euclidienne est 8. Donc 81%10 renvoie 1 alors que 81//10 renvoie 8. **Remarque :** avec Python, ces instructions fonctionnent aussi sur les variables de type flottant mais, au lycée, on s'en sert généralement sur des entiers uniquement.		

Variables de type *chaîne de caractères*		
Opérations	**Symboles**	**Exemples**
Concaténation de deux chaînes	+	`print("Bonjour" + " le monde")`
Explication : La concaténation met simplement bout à bout les chaînes de caractères. Le résultat obtenu est donc une seule chaîne de caractères dont la valeur est “Bonjour le monde”. On remarquera l’espace avant le mot “le” dans la deuxième chaîne de caractères.		

Info : Pour faire un petit commentaire dans un programme, il suffit d’utiliser le symbole #. Tout ce qui sera écrit après sur la même ligne sera ignoré par Python.

F) Exercices

✓ **Exercice 1 :**

Dans le cadre ci-dessous, on a défini plusieurs variables. Pour chacune d’elles, donner leur nom, leur valeur et leur type.

A ← 5.2
B ← “Bonjour”
C ← 6
D ← “Choisir un nombre entier”
E ← False
F ← A + C
G ← C - 10
H ← (26 + 4*2 - (12 + 6*3))/2

✓ **Exercice 2 :**

Dans le programme ci-dessous, on a défini plusieurs variables. Pour chacune d’elles, donner leur nom, leur valeur et leur type.

```
1 nombre = "3"
2 autre_nombre = 5
3 texte = True
4 autre_texte = "False"
5 somme = nombre + nombre
6 autre_somme = autre_nombre + autre_nombre
```

✓ **Exercice 3 :**

On considère le programme ci-dessous.

```
1 prixBaguette = 0.9
2 nombreBaguette = 3
3 total = prixBaguette * nombreBaguette
4 print(total)
```

1) Pour chaque variable, donner leur nom, leur valeur et leur type.
2) Que fait ce programme ?
3) Écrire l'algorithme correspondant en utilisant les bonnes notations.

✓ **Exercice 4 :**

1) Écrire un algorithme qui utilise trois variables nommées cote, perimetre, aire et qui calcule et affiche le périmètre et l'aire d'un carré dont la longueur du côté est la valeur de cote.
2) Programmer cet algorithme avec Python.

✓ **Exercice 5 :**

La bibliothèque *math* permet d'accéder à la variable pi dont la valeur est fixée par Python et qui est approximativement égale à π.
1) Afficher la valeur de la variable pi dans la console Python.
2) Écrire un algorithme qui calcule et affiche le périmètre d'un cercle de rayon R donné et l'aire d'un disque de même rayon.
3) Programmer cet algorithme avec Python.

✓ **Exercice 6 :**

On considère un parallélépipède rectangle $ABCDEFGH$ tel que $AB = L$, $BC = \ell$ et $AE = h$.
1) Écrire un algorithme qui calcule et affiche le volume de ce pavé droit ainsi que sa surface latérale lorsque les valeurs de L, ℓ et h sont données.
2) Programmer cet algorithme avec Python.