Chapitre F.1 - Construction de programme

I. Introduction

"**Un programme** informatique est un ensemble d'instructions et d'opérations destinés à être exécutées par un ordinateur."
Source : Wikipedia
Pour écrire un programme, on utilise un langage particulier appelé **langage de programmation**.
En Numérique et Sciences Informatiques, nous utiliserons le langage **Python**.

> **Python** est un langage de programmation crée en **1989** par **Guido Van Rossum**.

II. L'environnement de développement

A. Définition

Pour lancer un programme python, il faut écrire le code du programme dans un fichier python (**.py**) puis lancer l'interpréteur Python. Il est fastidieux d'écrire et d'exécuter son code à partir de deux logiciels différents, puisqu'il faut jongler entre les deux logiciels en permanence.
Un **environnement de développement** est un logiciel permettant d'écrire du code et de l'exécuter dans un seul et même programme. De plus, il est généralement accompagné d'outils permettant de faciliter l'écriture du code et la correction d'erreurs.

B. Thonny / Edupython

III. Les bases de la programmation

A. Les opérateur python

1. Opérateurs de calcul

Opérateur	Opération
+	_____
-	_____
*	_____
/	_____
//	_____(division entière)
%	Modulo (reste de la division entière)
**	_____
+	_____ (mise bout à bout de chaine de caractères)

2. Opérateurs de comparaison

Opérateur	opération
`<`	_____
`<=`	_____
`==`	____
`>=`	_____
`>`	_____

B. Les types de données

1. Les types de bases

Type Python	Type représenté
`int`	_____
`float`	_____(Flottant)
`bool`	_____(Deux valeurs `True` (Vrai) `False`(Faux))
`str`	_____

La fonction python `type` permet de connaître le type d'une expression.

Exemple :

```
>>> type(5+6)
<class 'int'>
```

2. Changer le type d'une expression

Il est parfois nécessaire de changer le type d'une expression. Pour changer le type d'une expressions on utilise les fonction suivantes :

- La fonction `str` permet de changer le type de la valeur en str.
- La fonction `int` permet de chagner le type de la valeur en int.
- La fonction `float`permet de changer le type de la valeur en float.

Exemple :

```
>>> int("55")
55
```

Lorsque la conversion n'est pas possible, le programme déclenche une erreur et s'arrête.

C. Les variables

Une **variable** est _____

1. Le **nom** de la variable : c'est ce qui permet de reconnaître la variable. Il nous permet de ne pas retenir l'adresse mémoire dans laquelle est stocké la valeur.

> Le nom d'une variable doit être uniquement composé par :
>
> - Des lettres (minuscule ou majuscule).
> - Des chiffres (⚠ Le nom d'une variable ne peut pas commencé par un chiffre.).
> - Le caractère undescore (_).

2. La **valeur** : c'est la donné stocké.

1. Créer une variable

Pour créer une variable, on écrit son nom puis on lui affecte sa valeur en utilisant l'opérateur `=`.

Exemple :

```
ma_variable = 10
```

2. Modifier la valeur d'une variable

Pour modifier la valeur d'une variable, on utilise également l'opérateur `=`.

Exemple :

```
v = 10 # On affecte 10 à la variable v
v = 5 # On modifie la valeur de la variable v
```

3. Utiliser une variable

Pour utiliser la valeur conserver dans une variable, il faut écrire le nom de la variable à la place de la valeur.

Exemple :

```
annee_naissance = 2010
age = 2024 - annee_naissance
```

D. Interaction avec l'utilisateur

1. Afficher un texte dans la console

Pour afficher des informations dans la console, on utilise la fonction `print`.

On écrit le mot `print` suivie du texte à afficher dans la console entre parenthèse. Il est possible de mettre plusieurs valeurs dans les parenthèses pour afficher plusieurs valeurs à la suite.

Exemple 1 :

```
print("Hello World !")
```

Exemple 2 :

```
nom = "Wotte"
prenom = "Ca"
print("Vous vous appelez ", nom, " ", prenom)
```

2. Demander une valeur à l'utilisateur

Pour demander une valeur à l'utilisateur, on utilise la fonction `input`.
On écrit le mot `input` suivie de la demande faite à l'utilisateur entre parenthèse. Pour réutiliser la valeur obtenue plus tard dans le programme, on l'affecte à une variable.

Exemple :

```
age = input("Quelle est votre age ?")
print("Vous avez ", age, "ans")
```

⚠ Le type de donnée obtenu par `input` est `str`. Pour pouvoir faire un traitement des données il faudra parfois modifier le type de la donnée.

Exemple :

```
valeur = int(input("Quelle nombre multiplier par 10 ?"))
resultat = valeur * 10
print(resultat)
```

IV. Commenter son programme

Lorsque l'on écrit un programme, celui-ci a pour vocation d'être lu. Que ce soit par une autre personne ou par soit-même. Il faut alors analyser le programme et le tester pour comprendre comment celui-ci fonctionne.
Il peut être difficile de comprendre ce que fait un programme rapidement un programme qui a été écrit par quelqu'un d'autre. Pour résoudre ce problème, il faut commenter sont code pour expliquer comment il fonctionne.
Pour écrire un commentaire en python on utilise le caractère `#`. Les éléments situé après le caractère `#` ne seront pas interprété par Python.

Exemple :

```
# Cette ligne est un commentaire, elle ne sera pas exécutée.
print("Bonjour") # Seul le début de la ligne sera exécuté.
```

> ⚠ Il est difficile de comprendre un code qui ne contient pas de commentaire mais un code contenant trop de commentaires devient illisible.
> Il est donc important d'utiliser les commentaires pour expliquer les parties de code plus difficile à comprendre.