SNT Programmation 3 – Variable

I – Affectation

1.1 Affectation

Définition : l'affectation est le nom qu'on donne au mécanisme permettant de créer une liaison entre un NOM et un CONTENU MEMOIRE.

En Python : on réalise une affectation en utilisant l'opérateur = .

Exemple : `a = 10` On crée une variable nommée a et y stocker la valeur 10.

Dans les algorithmes : on utilise plutôt la représentation suivante utilisant une flèche : `a ← 10`

Visualiser une variable :

Dans la console interactive, il suffit de taper le nom de la variable. Exemple : `>>> a` va afficher 10 .
Dans un programme, il faut le demander explicitement à l'aide de la fonction `print()`. Exemple : `print(a)`

1.2 Exemples d'affectation

Déroulé de l'affectation

On commence par évaluer ce qui est à droite de =, puis on affecte cette valeur à la variable de gauche.

Exemple 1

```
1   a = 20 + 5
```

TRADUCTION : affecte 25 à la variable a.

Exemple 2

```
1   a = 20
2   b = a + 100
```

TRADUCTION : évalue a + 100 et affecte cette nouvelle valeur de 120 à la variable b.

II – Récupérer les entrées clavier : input()

2.1 - But des variables : stocker une future information

Comprendre mentalement un programme

En tant qu'humain, vous avez juste besoin que de COMPRENDRE ce que réalise le programme. Vous devriez comprendre ce que le programme va réaliser, un jour, lorsqu'il aura accès à des valeurs précises.

```
1   note1 = ...
2   note2 = ...
3   moyenne = (note1 + note2) / 2
4
5   print(moyenne)
```

L'utilisation d'une variable permet de COMPRENDRE un programme sans savoir EXACTEMENT ce que seront les données : on doit simplement connaître quel type d'informations elles contiennent.

2.2 La fonction native input()

Principe : En Python, une fonction permet d'"écouter le clavier" : la fonction **input()**. Elle attend que l'utilisateur tape sur ENTREE et renvoie alors sa réponse sous forme d'un string (si vous tapez 17, Python reçoit "17").

Exemple : pour comprendre le programme d'exemple ci-dessous, il vous suffit de savoir que la variable **reponse** va contenir un string une fois que l'utilisateur aura validé avec la touche ENTREE. Le texte reçu correspondra aux touches enfoncées par l'utilisateur. Vous ne pouvez bien entendu par savoir ce que l'utilisateur va vraiment taper dans le futur.

```
1   print("Donnez votre nom puis validez avec la touche ENTREE : ")
2   reponse = input()
3   print("Merci. Votre nom est :")
4   print(reponse)
```

On comprend que :

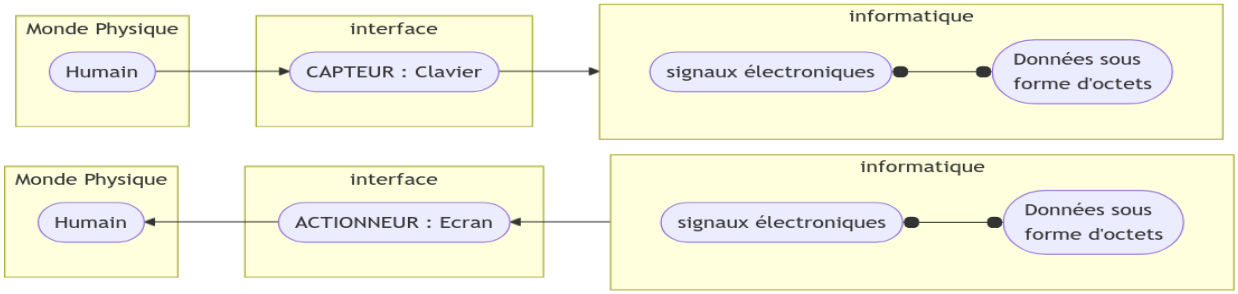
- on demande son nom à l'utilisateur en ligne 2;
- on stocke sa réponse dans la variable nommée reponse en ligne 2 également;
- on affichera sa réponse en ligne 4.

Récupérer un entier ?

Tout ce que vous tapez va être transmis à Python sous forme d'un string. Mais on peut utiliser la fonction native **int()** qui permet de tenter de créer un entier à partir de la donnée qu'on lui fournit. Exemple d'utilisation :

```
1   print("Donnez votre note puis validez avec la touche ENTREE : ")
2   reponse = int( input() )
3   print("Comme je suis gentil, je vous rajoute 3 points :")
4   print(reponse + 3)
```

2.3 Objets connectés : IMH, actionneur, capteur



IHM : C'est l'acronyme d'**Interface Homme Machine.**
Nous allons le retrouver dans tous systèmes où un utilisateur interagit avec un système informatique.

Capteur : Dispositif qui récupère une information du monde extérieur et l'envoie au système informatique sous forme d'un signal physique : clavier, souris...

Actionneur : Dispositif qui récupère une donnée informatique et l'envoie au monde extérieur sous forme d'un signal physique : écran, haut-parleur...

III – Incrémentation

3.1 Vocabulaire

Définition : Incrémenter veut dire utiliser la valeur actuelle d'une variable pour calculer sa nouvelle valeur.
Incrémenter de 1 en partant de 45 veut dire qu'on passe à 46.
Incrémenter de 2 en partant de 45 veut dire qu'on passe à 47.
Incrémenter de 5 en partant de 45 veut dire qu'on passe à 50.
Incrémenter de -1 en partant de 45 veut dire qu'on passe à 44.
On dit plutôt **décrémenter**.
Incrémenter (sans autre information donnée) veut dire qu'on incrémente de +1. il s'agit de la valeur par défaut.

Faire une incrémentation en Python

```
1   a = 10    # après cette ligne, a référence 10
2   a = a + 1 # après cette ligne, a référence 11
```

La demande s'effectue bien de la droite vers la gauche :

1. a = **a + 1** : évaluer ce que donne cette valeur + 1
2. **a =** a + 1 : affecter le résultat de cette évaluation à a.

3.2 Exemple d'incrémentation

```
1   a = 20
2   a = a + 100
3   print(a)
```

On calcule a+100 puis affecte 120 à **a**. On affiche 120.

```
1   a = 10
2   a = a + 3
3   print(a)
```

On calcule a+3 puis affecte 13 à **a.** On affiche 13.

```
1   a = 10
2   a = a + 10
3   a = a + 50
4   print(a)
```

On calcule a+10 puis affecte 20 à **a**. Ensuite, on calcule a+50 et on affecte 70 à **a**. On affiche finalement 70.