Enseignement SNT

LYCÉE de l'ÉLORN
le choix de votre réussite

PROGRAMMATION
avec le Langage Python

I- Le langage informatique :

Un programme informatique est une suite d'opérations ou d'instructions permettant de résoudre un problème ou d'obtenir un résultat. Nous utiliserons le langage Python inventé en 1991 par Guido Van Rossum (hollandais). Python est gratuit, sous licence libre, et fonctionne sur plusieurs architectures (PC, tablettes, smartphones, ...)

II- Le b.a.-ba du Python

Les variables

Une variable est un espace mémoire dans lequel il est possible de stocker une valeur

Les variables sont définies par deux caractéristiques essentielles, à savoir :

L'identificateur : le nom de la variable.
Le type : la nature de la variable (entier, réel, caractère, chaîne de caractères …)

Le nom des variables en Python peut être constitué de lettres minuscules (a à z), de lettres majuscules (A à Z), de nombres (0 à 9) et du caractère souligné (_).

Les types de bases

Nom français	Nom Python	Désignation	Exemple
Entier	Int ou long	Nombre entier (sans virgule)	1 10 20254
Réel	float	Nombre à virgule	1,1 5425,87458 12,0
Chaîne de caractères	str	Suite quelconque de caractères	"Bonjour" "Ok"
booléen	bool	Peut prendre les valeurs True ou False (vrai ou faux)	True ou False

Il existe plusieurs fonctions en Python qui permettent de forcer le type d'une variable en un autre type :
int() : permet de modifier une variable en entier.
float() : permet la transformation en flottant.
str() : permet de transformer la plupart des variables d'un autre type en chaînes de caractères.

Exemple : Soit x une variable de type chaîne de caractères : x="50"

L'opération x+10 générera une erreur. Pour réaliser une opération arithmétique sur la variable x nous devons la convertir en type entier avec la commande suivante : x=int(x)

→Réaliser ces tests dans l'ordre avec votre Numworks en mode console Python :

```
>>> x= "50"
>>> x
>>> x+10
>>> x*10
>>> x=int(x)
>>> x
>>> x+10
<<< x*10
```

Pour connaître le type d'une variable on peut utiliser la fonction type()

```
>>> x= "50"
>>> type(x)
>>> x=int(x)
>>> type(x)
>>> x=x+1.5
>>> type(x)
>>> x=11.5
>>> type(x)
```

Les opérateurs d'entrées/sorties en Python

a) Les sorties

Pour permettre au programme en cours d'exécution d'afficher un texte ou un nombre on utilise la commande print().

```
print("Bonjour !")      # c'est la chaîne de caractère "Bonjour" qui sera affichée.
a = 3
print(a)                # c'est le contenu de la variable a qui sera affiché.
print("Le carrée de", a,"est", a * a)   # affichera « Le carrée de 3 est 9 »
```

b) Les entrées

Il est souvent nécessaire de donner une valeur en utilisant le clavier. On utilisa alors la commande Input().

```
nom = input("Quel est votre nom ?")
#nom contiendra la chaîne de caractère saisie au clavier
```

Input() est une fonction **qui renvoie toujours une chaîne de caractères**. Il est donc parfois nécessaire de changer le type de la variable rentrée.

```
n = input("Entrer un nombre")          # n est une variable de type String
n = int(n)                             # n devient une variable de type entier
print("Le carrée de", n,"est", n * n)
```

Les opérateurs de base en python

+	addition	
-	soustraction	
*	multiplication	
/	division	5/2 donne 2.5
//	division entière	5//2 donne 2
%	reste de la division entière	5%2 donne 1 -> Numworks : fmod(5,2)
**	puissance	2**10 donne 1024

==	égalité (à ne pas confondre avec l'affectation)
!=	différent
<, >, <=, >=	inférieur, supérieur, inférieur ou égal, supérieur ou égal
and	opérateur booléen ET
or	opérateur booléen OU
not	opérateur booléen NON

→Tester (en mode console sous Python) la valeur d'une variable contenant un nombre.

Soit n une variable contenant le nombre 12 (faire n=12 avant de les tests).

Test en français	***Écrit en Python 3***	***Renvoie quoi ?***
n est égal à 12	n==12	
n est égal à 10	n==10	
n est positif	n>0	
n est différent de 10	n!=10	
Si *n* est compris strictement entre 0 et 20	(n>0) and (n<20) ou alors 0<n<20	

Les structures algorithmiques fondamentales

a) La structure "*SI ALORS SINON*"

Nous utilisons cette structure dès que nous voulons exprimer une possibilité de choix simple à deux alternatives. C'est une des opérations les plus utilisées, nous pourrions tout décrire avec des opérations simples et des « SI ALORS SINON ».

Exemple :

```
If note_ds >= 10:
    print("Vous avez la moyenne")
else:
    print("Vous n'avez pas la moyenne")
```

Le bloc else: est optionnel

b) La structure de boucle *"REPETER TANT QUE" (WHILE)*

Dans cette structure on commence par tester une condition. Si elle est vérifiée, le traitement est exécuté.

Exemples :

Test d'un mot de passe

```
mot_de_passe=""

while mot_de_passe!="bidule":
    mot_de_passe=input("mot de passe svp : ")

print("le mot de passe est validée")
```

Écrit la table de multiplication de 7

```
i=1
while i<11:
    print(i, "x 7 =", i*7)
    i=i+1
```

c) La structure de boucle *"POUR … DE … A … " (FOR)*

Lorsque l'on souhaite répéter un nombre donné de fois la même instruction ou le même bloc d'instructions, la commande for est la plus appropriée.

Exemples :

Écrit tous les nombres de 0 à 99 inclus

```
for i in range(0,100):
    print(i)
```

Écrit la table de multiplication de 7

```
for i in range(1,11):
    print(i, "x 7 =", i*7)
```

Application

→Programmer en python une application demandant le prénom d'un utilisateur puis son âge afin d'écrire le compte-rendu suivant :

« Bonjour Jojo, tu es donc né(e) en 2005 » *(si la personne s'appelle Jojo et qu'elle a 16 ans).*

→Ajouter au programme précédent un affichage mentionnant si l'utilisateur est majeur ou mineur.

« Bonjour Jojo, tu es donc né(e) en 2005, tu es mineur(e)» *(si la personne s'appelle Jojo et qu'elle a 16 ans).*

→Programmer en python une application demandant la saisie d'un nombre puis d'afficher la table de multiplication de ce nombre de 0 à 10.