Sciences de l'ingénieur

LYCÉE de l'ÉLORN
le choix de votre réussite

PROGRAMMATION

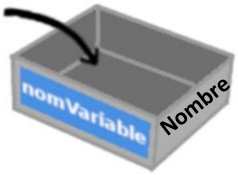
Rappel des structures de base en Python

1. Les variables

1.1. Les types de bases :

Une variable est un espace mémoire dans lequel il est possible de stocker une valeur. C'est un peu comme une « boite » sur laquelle il y aurait une étiquette désignant le nom de son contenu. Les algorithmes manipuleront le nom de la variables (nom des boites) plutôt que leur contenu.

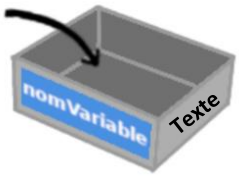
2023



- Le **nom** de la variable correspond à l'étiquette collée sur la boîte.
- La **valeur** de la variable est l'information stockée. Cette valeur peut changer au cours de l'exécution d'un programme.

Les variables peuvent être de différents types. Il faut prendre soin d'adapter le contenu au contenant et vice-versa.

"Toto"



Dans cet exemple le contenu de la variable C sera 25 après l'exécution du programme

```
a = 10
b = 15
c = a + b
```

1.2. 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)	

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.
long() : transforme une valeur en long.
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).

Pour connaître le type d'une variable on peut utiliser la fonction type() -> type(x) donne Str si x="50".

Précision sur les variables de type chaîne de caractère :

Les chaînes sont des séquences de caractères. Pour manipuler un caractère d'une chaîne, il suffit d'accoler au nom de la variable qui contient cette chaîne, son indice entre crochets :
Exemple :
nom = "Lycée de l'Elorn ''
nom[0] renvoie "L"
nom[3] renvoie "é"
Les chaînes sont non mutables (lecture seulement) -> nom[1]= "Y" est impossible

len(nom) renvoie 16 (nombre de caractère constituant la variable)

2. Les opérateurs d'entrées/sorties en Python

2.1. 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("Le carrée de", a,"est", a * a)   #affichera « Le carrée de -3 est 9 »
```

2.2. 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.

```
nombre = input("Entrer un nombre")                  # ou directement : nombre = float(input("Entrer un nombre"))
n = float(nombre)
print("Le carrée de", n,"est", n * n)
```

3. Les opérateurs de base en python

+	addition	
-	soustraction	
*	multiplication	
/	division	2/2 donne 2.5
**	puissance	2**10 donne 1024
//	division entière	5//2 donne 2

% reste de la division entière 5%2 donne 1
== 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

Soit n une variable contenant le nombre 12

Test en français	***Test en Python 3***	***Renvoie***
n est égal à 12	n==12	True
n est égal à 10	n==10	False
n est positif	n>0	True
n est différent de 10	n!=10	True
Si *n* est compris strictement entre 0 et 20	0<n<20 (n>0) and (n<20)	True
Si *n* est divisible par 6	n%6==0	True
Si *n* est divisible par 5	n%5==0	False

4. Les structures algorithmiques fondamentales

4.1. La structure "*SI ALORS SINON" (IF THEN ELSE)*

Notation algorithmique

```
si  condition  alors
  action1
sinon
  action2
finsi
```

Exemple en Python :

```
note_ds = 12

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

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

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

```
tant que condition
  action1
  action2
  …
Fintant que
```

Exemples en python :

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.

```
pour compteur de début à fin
   action1
   action2
  …
finpour
```

Exemples en python :

Écrit tous les nombres de 0 à 100 inclus

```
for i in range(0,101):
    print(i)
```

Écrit la table de multiplication de 7

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