Chapitre – ALGORITHME à Python

SCRATCH

AVANT IL Y AVAIT SCRATCH :
RAPPEL et directives pour la suite.

A présent, il y aura Python : Action !

Les principales instructions en Python

1- Variables : types de variables et leurs manipulations...

LES VARIABLES

Type	Pour qui ?	Exemples
int	Les entiers	..., –5, –1, 0, 2, 6, ...
float	Les réels	–5.28, 13.0, 18.7
str	Les chaînes de caractères (texte)	'Janvier', '28b6', '13'
bool	Les booléens : uniquement deux valeurs possibles	True (Vrai) ou False (Faux)
list	C'est une liste ou tableau dans lequel on peut stocker plusieurs valeurs idéalement de même type	[0, 1, 2, 4], ['Janvier', 'Février', 'Mars']

Pour visualiser le type d'une variable *x* en python, il suffit d'utiliser la fonction ***type(x)***

L’affectation

L’affectation est l’opération qui permet de définir la valeur ou le contenu d’une variable. En python l’affectation est définie avec le signe =

`longueur = 7.5` #signifie que l’on affecte la valeur 7.5 à la variable appelée longueur

`Prenom_joueur = “Noémie”` #signifie ……

Manipulation des variables

Les principaux opérateurs arithmétiques (sur des nombres 'int' ou 'float')

x + y	*Addition*
x – y	*Soustraction*
*x * y*	*Multiplication*
x / y	*Division décimale de x par y*
x // y	*Quotient de la division euclidienne de x par y*
x % y	*Reste de la division euclidienne de x par y (Modulo)*
*x ** y*	*Puissance : x^y*

opérations sur les chaînes de caractères (les chaînes de caractères sont des suites de signes, comme un mot qui est une suite de lettres)

Les principaux opérateurs sur des chaînes de caractères

ch1 + ch2	*Concaténation (les chaînes ch1 et ch2 sont mises bout à bout)*
len(ch1)	*Nombre de caractères de la chaîne ch1*
ch1[i]	*(i +1)ème caractère de la chaîne ch1*

IMPORTANT- L'insertion de commentaires : une nécessité

En python (mais dans d'autres langages également), et dès que vous écrivez un programme un tout petit peu complexe, il faut commenter ce que vous faites.

Ex. ce petit programme très simple

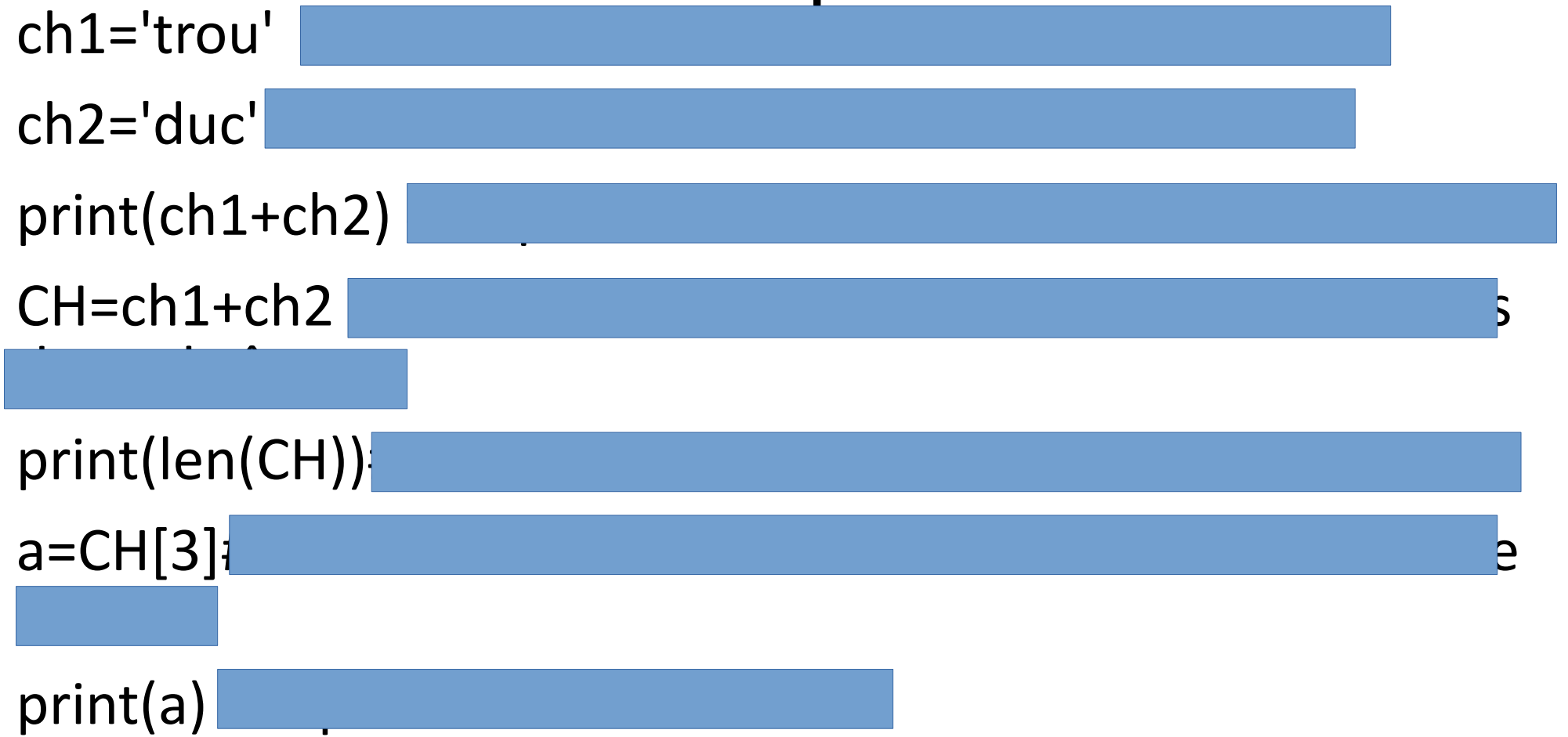
→ Exercice d'entraînement

```
age = int(56)     #attribue une valeur d’entier à la variable âge

print(‘Vous avez’, age, ‘ans’) #affiche une phrase avec insertion de la variable
```

exemple:

```
ch1='trou'
ch2='duc'
print(ch1+ch2)
CH=ch1+ch2
print(len(CH))
a=CH[3]
print(a)
```



exemple:

```
ch1='trou'  #attribue une chaîne trou à la variable ch1

ch2='duc' #attribue une chaîne duc à la variable ch1

print(ch1+ch2) # imprime la concaténation les deux chaînes

CH=ch1+ch2 #attribue à la variable CH la concaténation les deux chaînes

print(len(CH))#imprime/affiche la longueur de la chaîne CH

a=CH[3]#attribue à la variable a le 4° caractère de la chaîne CH

print(a) #imprime la variable a
```

2- Interaction avec l'utilisateur (entrées/sorties)

Comme on vient de le voir, la plupart du temps, dans les programmes, on demandera à l'utilisateur d'entrer des valeurs. Pour ce faire, en python, on utilise l'instruction `input()`.

Dans les parenthèses, on peut mettre, entre guillemets, une phrase explicitant ce que l'on attend de l'utilisateur

Ex.

```
Prénom = input("Quel est votre prénom? ")
```

Cette instruction prendra les signes entrés par l'usager comme une chaîne de caractères. Si c'est une valeur numérique que vous attendez, il faudra combiner cette instruction `input()` à d'autres instructions. Voici ce que nous allons rencontrer le plus fréquemment :

Input() → Par défaut, ce qui est saisi par l'utilisateur est une chaîne de caractères.

int(input())→ Permet de recueillir un entier entré par l'utilisateur (c'est utile pour l'âge, pour le nombre de personnes présentes ou absentes, etc.)

float(input())-→ Permet de recueillir un réel entré par l'utilisateur (c'est utile pour la température, , etc.)

Mini projet express

Afficher la table de multiplication de x après demande à l'utilisateur.

3- Instructions conditionnelles : IF et ELSE, opérateurs logiques OR, AND, NOT, cas multiples: ELIF

En algorithmique, on peut vouloir effectuer des instructions selon qu'une condition soit satisfaite ou non. On parle alors d'instruction conditionnelle. En français, on peut résumer cette situation de la façon suivante “Si la condition est satisfaite, Alors on réalise une action, Sinon on réalise une autre action”

opérations que l'on dit booléennes (des tests qui peuvent se faire sur n'importe quel type de variable):

$x < y$	*Tester si x est inférieur à y*
$x <= y$	*Tester si x est inférieur ou égal à y*
$x > y$	*Tester si x est supérieur à y*
$x >= y$	*Tester si x est supérieur ou égal à y*
$x == y$	*Tester si x est égal de y*
$x != y$	*Tester si x est différent de y*

```
age = int(input("Entrez votre âge : "))
if age<18:
   print("Vous n'êtes pas majeur!") #remarquez le décalage de la ligne
Else:
   print ("vous avez l'âge requis")
```

Les conditions peuvent se combiner entre elles

x and y	Intersection
x or y	Union
not y	Négation

```
age = int(input('Donnez votre âge'))   #attribue une valeur d'entier à la variable âge
if age<18:                             #Réalise un test sur la valeur entrée en âge
    print("Vous n'êtes pas majeur")    #Donne une réponse pour non-majeur
else:
  print ("vous avez l'âge requis")     #Donne une réponse dans le cas contraire
```

Mini projet express

EXO:

Afficher la table de multiplication de x après demande à l'utilisateur. Et après lui avoir demandé un mot de passe(qu'il connaît!!!)

```python
def main():
    pass

if __name__ == '__main__':
    main()
nbre=int(input("Donne un chiffre pour afficher sa table")) #pour demander quelle
code=str(input("rentre ton code"))
if code=="1234":
    print("la table de multiplication de ", nbre)
    print(0,nbre, nbre*2,nbre*3, nbre*4, nbre*5) #affichage résultats
else:
    print("rentre chez ta mère!!!")
```

4- Boucles : boucles bornées, boucles non bornées

Pour exécuter plusieurs fois un bloc d'instructions ((souvenez-vous du dessin d'un carré par exemple), on utilise ce que l'on appelle une boucle.

Il existe des boucles dont on connaît le nombre de répétitions, les boucles bornées, on utilise alors l'instruction `for` et la fonction `range()`. Cette fonction admet 3 paramètres (le début, la fin, le pas). Par défaut et si vous ne précisez rien, le début est 0, il est impératif de saisir un paramètre de fin et le pas est par défaut 1, autrement dit la fonction va par défaut aller de un en un.

`for i in range (5):` `  print(i)`	Voici un programme qui va vous afficher 0 puis 1 puis 2 puis 3 puis 4 (en commençant par défaut à 0 et tant que i<5)
`for i in range (0,10,2):` `  print(i)`	Voici un programme qui va vous afficher 0 puis 2 (mais pas 1 parce que on a demandé de sauter une valeur sur 2) puis 2 puis 4 puis 6 puis 8 (et s'arrête juste avant que i atteigne la valeur 10)

Mais il existe aussi des boucles dont on ne connaît pas le nombre de répétitions, on les introduit par l'instruction `while` suivie d'un test (la boucle sera répétée tant que le test est vrai)

```
while condition-de-continuation-de-la-boucle:
    Instruction(s)  # corps de la boucle
```

Une seule exécution du corps de la boucle est appelée une **itération** (ou répétition) de la boucle.

Exercice: voici un programme, quel sera le résultat?

```
somme = 0
i = 1
while i < 10:
    somme = somme + i
    i = i + 1
print("la somme est ", somme)
```

Mini projet express

EXO:

Afficher la table de multiplication de x après demande à l'utilisateur. Et après lui avoir demander un mot de passe(qu'il connait!!!)

Le programme doit comporter une boucle de votre choix afin de réduire sa taille...

5- Fonctions et bibliothèques de modules

Fonctions

Au sens algorithmique, une fonction est une sorte de boîte noire qui admet en entrée des valeurs que l'on appelle des paramètres et qui retourne un résultat stockable dans une variable.

Nous verrons plus tard comment créer des fonctions dans Python même si nous en avons déjà manipulé (la fonction `range()`, la fonction `print()` par exemple)

Activité 01-02 annotation des instructions pythons, analyses de programmations.

Pour créer une fonction:

```
def NomDeLaFonction(variable1, variable2...):
    A=variable1*variable2
    return (A)
```

pour l’utiliser:

NouvelleVariable=NomDeLaFonction(4,78)

#permet d’affect à la variable NouvelleVariable une nouvelle valeur de 312...

- Bibliothèques et modules
- Une bibliothèque ou un module est un paquet de fonctions regroupées parce qu'elles se rapportent à une thématique commune. Il existe ainsi dans Python des modules de fonctions créés par des utilisateurs. Pour accéder à un module, on utilise le mot-clé `import` suivi du nom du module que l'on veut utiliser, nous reverrons cela en détail.
- Voici cependant le nom de modules que nous mobiliserons dès cette année :
- `tkinter` : afficha ge graphique (nous utiliserons aussi `turtle`)
- `math` : calcul mathématique
- `random` : traitement aléatoire
- `time` : gestion du temps
- `pillow` : traitement d'images numériques