Python

Première version en 1991 par Guido Van Rossum

langage interprété (# langage compilé) : les instructions sont interprétées ligne par ligne au fur et à mesure.

**//** : division entière

**/**: division normale (à virgule)

On met un **#** devant une ligne de commentaire

la fonction **help(‘’** mot\_clé **‘’)** permet de se renseigner sur une fonction, un module ou une variable dans python

Notes importantes

la commande « **pip** **install** Module » pour installer les modules python (**pip3** pour python3)

Pour exécuter un programme Python à partir d’un fichier

créer le fichier.py => ajouter y la ligne indiquant le chemin de l’interpréteur de commande (ici **#!usr/bin/python3.5** ) et en dessous la ligne correspondant à l’encodage utilisé si jamais on veut faire des accents (sous linux c’est Utf-8 : **# -\*-coding :Utf-8 -\*** ) => changer les droits du fichier => ecrire le code et exécuter le fichier.

Les variables

Syntaxe : nom\_de\_la\_variable = valeur (python est sensible à la casse)

les types : int, float, string (entre coats, apostrophes ou 3 guillemets. NB: Dans une chaîne de caractère, on met un **\** avant **‘**,**\** …), boolean (valeur == True ou False)

NB : a,b = b, a => faire une permutation entre a et b.

x = y = 3 => affectation conjointe.

On peut faire des conversions de type : variable = int(variable)

Les fonctions

Syntaxe :

**def** nom\_de\_la\_fonction **(**param1, …, param n**) :**

#Bloc d’instructions

NB :

[autre façon de déclarer une fonction] nom\_de\_la\_fonction = **lambda** param1**,**… **,**param n **:** instruction de retour

**return** permet à une fonction de renvoyer une valeur

la fonction **type(**variable ou valeur**)** renvoie le type de la variable (ou valeur) passée en paramètre

la fonction **print()** pour afficher

ex : print (‘’a est égale à’’**,** a **,** ‘’et b= ‘’**,** b)

le fonction **imput()** permet de demander à l’utilisateur de saisir (la saisie est toujours vue comme une chaîne de caractère donc il faut souvent faire une conversion de type)

Les boucles

Syntaxe :

#Boucle if

**if (**condition**) :**

#instructions

**elif :**

#instructions

**else :**

#instructions

#Boucle While

**while(**condition**) :**

#instructions

#Boucle for

**for** element **in** sequence:

#instuctions

NB: Séquence peut être une chaîne de caractère (dans ce cas, élément représente chaque lettre).

NB : Lorsqu’on teste une condition sans if, l’interpréteur de commande renvoie **True** ou **False** selon la véracité de cette condition.

Pour associer les conditions, on utilise les mots clés : **and**, **or**, **not**

**break**:permet d’arrêter l’exécution d’une bouble (sortir avant la fin)

**continue**: permet de revenir à la condition de la boucle sans exécuter le reste de lignes qui suivent

Les modules ( comme les bibliothèques)

Syntaxe

**import** module ou **import** module **as** nom\_voulu

utilisation : **module.fonctio**n() ou **module.variable**

**ou**

ou encore **from** module **import** fonction\_ou\_variable\_précise

utilisation : **fonction()** ou **variable**