FICHE 101

Mémo Python

Variables

Affecter une valeur à une variable

`a=2`	La variable `a` prend la valeur 2.
`a=a+1`	La variable `a` prend la valeur `a+1`, ici 2+1 donc 3.
`a="texte"` ou `a='texte'`	Le mot « texte » est affecté à la variable `a`.
`a=float(a)`	`a` est converti en réel.

Affecter une valeur saisie par l'utilisateur

`a=input("Saisir un mot :")`	Le programme affiche « Saisir un mot : », attend la frappe de l'utilisateur et affecte la saisie à la variable `a`.

Afficher le contenu d'une variable

`print(a)`	Le programme affiche le contenu de la variable `a`.
`print("a =",a)`	Le programme affiche le texte « a = » suivi du contenu de la variable `a`.

Instructions conditionnelles

Une **instruction conditionnelle** n'est exécutée que si une certaine condition est vérifiée.

La commande « `else:` » est facultative.

`p=float(input("Saisir un nombre :"))`
`if p>0:`
`  print("Le nombre saisi est positif.")`
`else:`
`  print("Le nombre saisi est négatif ou nul.")`
`print("Bonne journée")`
	Si l'utilisateur rentre –5, le programme affiche « Le nombre saisi est négatif ou nul. » puis « Bonne journée ». S'il rentre 5, le programme affiche « Le nombre saisi est positif. » puis « Bonne journée ».

Boucle bornée

On utilise une **boucle bornée** lorsqu'on veut exécuter un nombre de fois déterminé un même bloc d'instructions.

La boucle démarre à `i=1` et s'arrête à `i=n`.

`u=1`
`for i in range(1,21):`
`  u=u+2*u`
`print(u)`
	La boucle ajoute le double de `u` à la variable `u` pour `i` allant de 1 à 20. À la fin de la boucle, le programme affiche `u`.

Boucle non bornée

On utilise une boucle **non bornée** lorsqu'on veut répéter un même bloc d'instructions tant qu'une certaine condition est vérifiée.

`u=0`
`while u<1000:`
`  u=u+20`
`u=u-20`
`print(u)`
	La boucle ajoute 20 à la variable `u` tant que `u` est plus petit que 1 000.
	Le programme affiche ainsi le plus grand multiple de 20 inférieur à 1 000.

Écrire une fonction

Il peut être utile de définir une **fonction**, c'est-à-dire un bloc d'instructions qui ne sera exécuté que s'il est appelé.

Une fonction possède généralement des **paramètres**.

`def difference(x,y):`
`  dif=x-y`
`  return dif`
	On définit la fonction différence de variable `x` et `y` qui renvoie la variable `dif` au programme principal.
	`print(difference(3,5))` renvoie –2.

Liste

Une **liste** est un tableau de valeurs dont les **éléments** sont indexés à partir de 0.

`L=[5,"pair", 8.1]`	On définit une liste nommée `L` à 3 éléments.
`print(L[1])`	Affiche le terme n° 1 de la liste, ici « pair ». Attention, une liste est indexée à partir de 0.
`L.append(7)`	Ajoute 7 à la fin de la liste.
`L.insert(i,"bob")`	Insère `bob` à l'indice `L[i]` et décale le reste de la liste vers la droite.
`len(L)`	Renvoie la taille de la liste `L`.
`del(L[i])`	Supprime l'élément d'indice `i` et décale les suivants vers la gauche.
`L.count(x)`	Renvoie le nombre d'occurrences de `x` dans la liste `L`.
`L=M+N`	La liste `L` est constituée des éléments de `M` puis des éléments de `N` à la suite.
`L.sort()`	Range les éléments de `L` par ordre croissant.
`if x in L:`	Teste si `x` est un élément de `L`.
`if x not in L:`	Teste si `x` n'est pas un élément de `L`.
`for i in L:`	Pour `i` prenant successivement les valeurs des éléments de `L`.