Algorithmique 2 : Python sur Numworks

Nous avons parlé dans la première partie des algorithmes écrits en langage naturel puis en pseudo-code. Pour pouvoir faire exécuter un tel algorithme par une machine, il faut le traduire dans un langage de programmation ; nous utiliserons ici le langage Python, à la fois simple et puissant.

Les affectations s'écrivent avec un = en Python, ainsi :

- en langage naturel, on écrit : a prend la valeur 300
  ou : Affecter à a la valeur 300
- en pseudo-code, on écrira ceci : a ← 300
- en Python, on écrira cela : a = 300

Utilisation de la console Python

La console Python (ou interpréteur) permet l'exécution de quelques instructions simples.

Pour accéder à la console Python dans la Numworks :

- dans la page d'accueil, à l'aide des flèches et du bouton OK, lancez l'application Python ;
- descendez jusqu'à "Console d'exécution" puis OK
- vous pouvez ensuite taper des instructions puis taper sur EXE pour lancer leur exécution.

Exercice 1 : utilisation de la console Python

1°) Essayez de prévoir la valeur finale de a à la fin du programme suivant puis tapez les instructions pour vérifier votre réponse :

Instructions	Commentaires	Comment faire

`>>> a = 5`
`>>> b = 3`
`>>> a = a - b`
`>>> b = a ** 2`
`>>> a`
	affecter 5 à a
	affecter 3 à b
	affecter la valeur de a – b à a
	affecter la valeur de a^2 à b
	afficher la valeur de a
		shift π pour le =
		
		le * s'obtient avec la touche ×

** sert au calcul des puissances en Python

Dans la console il suffit de taper le nom d'une variable pour voir sa valeur.

2°) Tapez la suite d'instructions suivante. Quel commentaire pouvez-vous faire sur celle-ci ?

```
>>> f = 4
>>> g = 1
>>> g = 2 * f
>>> g
```

Utilisation de l'éditeur Python

L'éditeur Python permet la sauvegarde et l'exécution de programmes plus complexes.

Pour accéder à l'éditeur Python dans la Numworks :

- dans la page d'accueil, à l'aide des flèches et du bouton OK, lancez l'application Python ;
- descendez jusqu'à "Ajouter un script" puis OK, donnez un nom au nouveau script, par exemple "algo", en utilisant les touches de la calculatrice (les lettres y apparaissent en gris) puis OK pour valider le nom et OK pour **éditer** le programme ;
- vous taperez plus tard un programme ;
- utilisez  pour sortir de l'éditeur ;
- pour exécuter un programme : avec la flèche droite, allez sur les trois points puis OK  et choisissez « Exécuter le script » ;

Exercice II : utilisation de l'éditeur Python

Voici un exemple de traduction d'un algorithme en programme :

Algorithme en langage naturel	**Algorithme en pseudo-code**	**Programme en Python**

• étant donné un nombre ;
• lui ajouter 2 et multiplier le tout par 3 ;
• mettre au carré le résultat ;
• afficher le résultat.
	y ← x + 2
	z ← y*3
	a ← z^2
	Afficher a
		`y = x + 2`
		`z = y*3`
		`a = z**2`
		`print(a)`

1°) Entrez ce programme dans la Numworks. Quelques indications :

pour …	utiliser l'alphabet	valider une proposition (en gris) / refuser	"	print()	passer à la ligne
faire …	appuyer deux fois sur alpha (et une fois pour en sortir) ou une fois pour une seule lettre	flèche droite / flèche gauche	 en mode alphabet	la taper ou utiliser le Catalogue dans  en mode non alphabet	

2°) Le programme amène à une erreur : lisez-là et comprenez-là. Corrigez l'erreur en donnant une valeur au nombre initial…

3°) Exécutez votre programme pour chacun des nombres suivants :

Nombre choisi	0	1	4	5
Résultat du programme				

La fonction print

En mode console, pour afficher la valeur d'une variable, il suffit de taper son nom (comme dans l'exercice I). En mode éditeur, il faut pour cela utiliser la fonction :

print(…)

Par exemple, pour afficher la valeur d'une variable nommée *dist*, on taperait :

print(dist)

(il faut que la variable *dist* ait été créée avant…).

Pour afficher un texte, on utilise des guillemets :

print("Ce programme calcule des distances.")

Pour mixer les deux, on sépare avec des virgules :

print("La distance calculée est :", dist)

Exercice III : sorties

Tapez dans l'éditeur Python, en les complétant, les programmes suivants de façon à ce qu'ils affichent ce qui est écrit dans la console quand on les exécute ; vous ne ferez aucun calcul…

	Programme (éditeur)	**Résultat de l'exécution (console)**

1°)	`long = 230230`
	`larg = 120120`
	`print(.....)`
		`Aire = 27655227600`

2°)	`print(.....)`	`2 puissance 1000 =` `10715086071862673209484250490600018105614048117055336074437503883703510511249361224931983788156958581275946729175531468251871452856923140435984577574698574803934567774824230985421074605062371141877954182153046474983581941267398767559165543946077062914571196477686542167660429831652624386837205668069376`

3°)	`moi = "Dupont"`
	`print(.....)`
	`print(.....)`
	`print(.....)`
		`Dupont est le plus intelligent`
		`Le plus fort : Dupont…`
		`Dupont ! Dupont ! Dupont !`
		`Dupont ! Dupont ! Dupont !`

Consigne ; vous pourrez changer le nom à la première ligne seulement et le reste du programme devra s'adapter…
Astuce : "to" * 2 donne "toto" !

Types de données

L'informatique est la science de la manipulation des informations, ou des données.
Celles-ci peuvent se présenter sous différents aspects, entre autres :

- des nombres entiers, tels que : 4 ; 236 ; – 21 etc.
- des nombres réels, tels que : 3.14 ; – 1.732 etc.
- du texte , par exemple : "Bonjour tout le monde !"
- et bien d'autres types de données…

Il est parfois nécessaire de passer d'un type à l'autre, on appelle cela un *trans-typage*.

Exercice IV : types de données

1°) Tapez dans la console et observez bien les réponses de Python :

```
>>> a = 30
>>> type(a)
>>> b = "test !"
>>> type(b)
```

La commande type(…) demande le type de donnée d'une variable.
En Python, les entiers sont de type *int* (integer = entier) et les textes de type *str* (string = chaîne).

2°) Comment Python appelle-t-il les nombres à virgules ?

3°) Tapez dans la console et observez bien les réponses de Python :

```
>>> nb1 = 10
>>> nb2 = 20
>>> nb1+nb2
>>> chn1 =  "encore un "
>>> chn2 = "test"
>>> chn1+chn2
>>> chn2+nb1
```

On ne peut pas ajouter un texte et un nombre. Si on veut obtenir le texte "test10", il faut convertir le nombre 10 en le texte "10". Ceci se fait à l'aide d'une fonction de trans-typage, ici str(…).
Il existe également les commandes int(…) et float(…) pour convertir en entier ou en réel.

3°) Tapez donc dans la console :

```
>>> chn2+str(nb1)
```

et testez également les instructions suivantes :

```
>>> str(nb1)
>>> float(nb1)
>>> int(nb1)
>>> int(5.25)
```

4°) Tapez et complétez, dans l'éditeur, le programme suivant.
Les nombres 38 et 7 ne doivent pas apparaître dans les lignes cachées.

Programme (éditeur)	Résultat de l'exécution (console)

`a = "38"`
`b = 7.0`
`print(.....)`
`print(.....)`
`print(.....)`
	`Le produit de 38 et de 7 est 266.`
	`La division de 38 par 7 donne`
	`5.428571428571429 environ.`
	`Donc dans 38 il y a 5 fois le nombre 7.`

Remarquez que la conversion du type entier vers le type réel est automatique (exemple : 4 + 7.02).

Entrées de l'utilisateur

Que se passe-t-il lorsqu'un utilisateur veut retirer de l'argent à un distributeur bancaire ?

- le distributeur affiche une zone d'entrée de code et attend que l'utilisateur entre son code ;
- le distributeur traite ensuite cette information (vérification du code puis actions adaptées).

De même, pour qu'un utilisateur puisse entrer des données dans un algorithme existant, on utilise la commande Saisir (ou Lire ou Demander) en langage naturel.

Cette commande se traduit en langage Python en

input(…)

Par exemple, les instructions :

```
print("Quel est votre code ?")
votre_code = input()
```

ou, en plus court :

```
votre_code = input("Quel est votre code ?")
```

affichent le texte « Quel est votre code ? », attendent la réponse de l'utilisateur et place la réponse dans la variable votre_code (qu'on pourra utiliser plus tard, pour vérification par exemple).

Exercice 5 : entrées

1°) Écrivez un programme en Python qui demande à un utilisateur son prénom (par exemple l'utilisateur tape : Toto) et qui affiche le message :

Bonjour *prénom_choisi* (par exemple : Bonjour Toto)

2°) Testez le programme suivant et essayez de le réparer…

```
nb = input()
print("Le double de ", nb, " est ", 2* nb)
```

La commande input(…) renvoie toujours du texte. Il sera ainsi parfois nécessaire de convertir ce texte en entier ou en réel, comme dans cet exemple :

```
larg = input("Largeur du rectangle :")
larg = float(larg)
```

ou directement :

```
larg = float(input("Largeur du rectangle :"))
```

Exercice 6 : quelques petits programmes en Python

Créez les programmes suivants dans Python (mode éditeur) et lancez-les pour vérifier qu'ils fonctionnent. Imaginez que ces programmes sont destinés à un autre utilisateur, qui n'a pas accès à vos programmes. Faîtes vérifier (valider) chaque programme par votre professeur.

Programme 1 :

La machine demande à un utilisateur son année de naissance (exemple : l'utilisateur choisit 2007) et l'année actuelle (exemple : 2023) et affiche le message :

C'était il y a … années (exemple : C'était il y a 16 années)

Programme 2 :
La machine demande à un utilisateur un nombre entier et affiche le message :

2 ^… = …

(exemple : l'utilisateur entre 3 et la machine répond 2^3 = 8)

Programme 3 :
La machine demande à un utilisateur la longueur du côté d'un carré et affiche le périmètre et l'aire de ce carré.

Exercice 7 : traduction en pseudo-code puis en Python

Voici un algorithme :

- choisir deux nombres entiers ;
- multiplier le premier nombre par 2 ;
- multiplier le second nombre par 3 ;
- ajouter les deux résultats précédents ;
- afficher le dernier résultat.

1°) Que donne cet algorithme quand on y entre les nombres 4 et 7 ?

2°) Traduisez l'algorithme en pseudo-code.

3°) Traduisez l'algorithme en Python, exécutez-le (vérifiez la réponse à la question 1°) pour voir si votre programme fonctionne bien).

Exercice 8 : Le magicien

Un magicien demande à un spectateur :

- de penser à un nombre entier ;
- de le multiplier par 5 ;
- d'ajouter 7 au résultat ;
- de multiplier par 4 le résultat ;
- d'ajouter 6 au résultat ;
- de multiplier par 5 le résultat ;
- d'annoncer le résultat final obtenu.

1°) Le spectateur pense au nombre 4, quel nombre annonce-t-il à la fin ? 2°) Le magicien trouve à chaque fois le nombre choisi au départ par le spectateur ! Soit il est très fort en calcul mental, soit il a un truc de magicien… Écrivez un programme Python qui : – demande un nombre entier ; – effectue les opérations demandées par le magicien ; – affiche le résultat final (celui que le spectateur annonce).	3°) Entrez ce programme dans l'éditeur Python puis, à l'aide de ce programme, vérifiez la réponse à la question 1°). Relancez plusieurs fois votre programme, choisissez d'autres valeurs de départ et cherchez un lien entre le nombre choisi par le spectateur et celui qu'il annonce. 4°) On appelle x le nombre choisi par le spectateur. Écrivez en fonction de x le résultat qu'il annonce. Prouvez alors la remarque faîte au 3°).