



Interagir avec un programme et effectuer des choix

Objectif

Convertir une saisie.

Découvrir les instructions conditionnelles.

Méthode de travail. Pour chaque expérience :

1. prévoir le résultat sans lancer Python ;
2. exécuter le code et relever le résultat exact ;
3. expliquer les éventuels écarts et ne pas hésiter à m'appeler ;
4. conserver les notes et les codes validés pour plus tard.

1 La fonction input

La fonction `input ( . . . )` affiche un message dans la console et attend une saisie au clavier. Exécuter le script suivant avec plusieurs âges.

```
age = input("Entrer votre age : ")
print("Votre age est :", age)
print(type(age))
age_nombre = int(age)
print("L'annee prochaine :", age_nombre + 1)
```

Question 1 : Déterminer le type renvoyé par `input`, puis expliquer la nécessité de la conversion `int(age)` avant le calcul.

Réponse 1 : `input` renvoie toujours une chaîne de caractères de type `str`, même lorsque l'utilisateur saisit des chiffres. La conversion `int(age)` construit l'entier correspondant et permet l'addition numérique `age_nombre + 1`.

1.1 La magie des maths

Multiplier le numéro du mois de naissance par 2, ajouter 5, multiplier le résultat par 50, ajouter 1776, puis retrancher l'année de naissance. Le résultat est ... étonnant.

Question 2 : Écrire un script utilisant `input` qui réalise ces opérations. Reproduire avec le mois et l'année de naissance de quelqu'un d'autre, observer la structure du résultat et déterminer les informations qu'il contient.

Réponse 2 :

```
mois = int(input("Mois de naissance : "))
annee = int(input("Annee de naissance : "))
resultat = (mois * 2 + 5) * 50 + 1776 - annee
print(resultat)
```

Le calcul se simplifie en $100 \times \text{mois} + 2026 - \text{année de naissance}$. Les derniers chiffres donnent donc l'âge atteint en 2026 et les premiers donnent le numéro du mois.

2 | Échanger deux valeurs

Question 3 : Écrire un script qui échange les valeurs de deux variables a et b. Proposer une méthode utilisant une variable intermédiaire, puis une affectation simultanée.

Réponse 3 : Avec une variable intermédiaire :

```
tampon = a
a = b
b = tampon
```

Python permet aussi l'affectation simultanée (élément à élément) :

```
a, b = b, a
```

3 | Découvrir les instructions conditionnelles

Une instruction conditionnelle exécute un bloc de code seulement lorsqu'une expression booléenne vaut True. En Python, les deux points terminent la ligne du if et l'indentation (tabulation 4 espaces), délimite le bloc.

```
if expression_booleenne:
    instructions_si_vrai
else:
    instructions_si_faux
```

3.1 Premiers essais avec True et False

```
if True:
    print("Ce bloc 1 est execute")
else:
    print("Ce bloc 1 est ignore")

if False:
    print("Ce bloc 2 est ignore")
else:
    print("Ce bloc 2 est execute")

if True:
    print("Le else est facultatif")
```

Question 4 : Prévoir les trois messages affichés, exécuter le programme, puis déterminer le rôle de else et de l'indentation.

Réponse 4 : Le programme affiche Ce bloc 1 est execute, Ce bloc 1 est execute, puis Le else est facultatif. Le bloc indenté sous if est exécuté si la valeur booléenne est vraie. Le bloc placé sous else est exécuté dans le cas contraire. La partie else est facultative.

3.2 Utiliser une variable booléenne

```
allumee = True
if allumee:
    print("La lumiere est allumee")
else:
    print("La lumiere est eteinte")
```

Question 5 : Tester successivement allumee = True et allumee = False, puis déterminer l'expression évaluée par le if.

Réponse 5 : Le if évalue directement la valeur de la variable booléenne allumee. Il n'est pas nécessaire d'écrire allumee == True.

3.3 Condition définie par une expression booléenne

Il n'est pas nécessaire de stocker d'abord le résultat d'une comparaison dans une variable. Une expression comme `a == b` ou `a > b` peut être placée directement après `if`. En effet, comme nous l'avons vu précédemment les expressions de comparaison booléennes renvoient un booléen et les conditions nécessitent soit `True` soit `False`.

```
a = 7
b = 7

if a == b:
    print("a et b sont égaux")

if a > b:
    print("a est strictement supérieur a b")
else:
    print("a n'est pas strictement supérieur a b")
```

Question 6 : Prévoir l'affichage pour $(a, b) = (7, 7)$, puis tester $(9, 4)$ et $(2, 5)$.

Réponse 6 : Pour $(7,7)$, le premier message et le dernier sont affichés. Pour $(9,4)$, seul `a est strictement supérieur a b` est affiché. Pour $(2,5)$, seul le dernier message est affiché. Chaque expression est évaluée lorsque Python atteint le `if` correspondant; son résultat est `True` ou `False`.

3.4 Enchaîner plusieurs if indépendants

Plusieurs instructions `if` peuvent se suivre. Chacune est alors testée, même si une expression précédente était déjà vraie.

```
note = int(input("Entrer une note entiere : "))

if note >= 10:
    print("Objectif 10 atteint")
if note >= 12:
    print("Objectif 12 atteint")
if note >= 16:
    print("Objectif 16 atteint")
```

Question 7 : Prévoir les messages affichés pour les notes 9, 11, 13 et 17, puis vérifier expérimentalement. Expliquer pourquoi plusieurs messages peuvent être affichés lors d'une seule exécution.

Réponse 7 : Les nombres de messages sont respectivement 0, 1, 2 et 3. Les trois `if` sont indépendants : Python évalue donc les trois expressions. Pour la note 17, elles sont toutes vraies.

3.5 Choisir un seul cas avec if, elif et else

Le mot-clé `elif`, contraction de *else if*, permet d'ajouter des cas. Dès qu'une expression est vraie, son bloc est exécuté et les cas suivants ne sont plus testés.

```
note = int(input("Entrer une note entiere : "))

if note >= 16:
    print("Niveau 3")
elif note >= 12:
    print("Niveau 2")
elif note >= 10:
    print("Niveau 1")
else:
    print("Niveau 0")
```

Question 8 : Tester les notes 9, 11, 13 et 17, puis comparer le nombre de messages avec l'enchaînement précédent. Justifier l'ordre décroissant des seuils.

Réponse 8 : Un seul message est affiché pour chaque note : niveaux 0, 1, 2 et 3. Les seuils sont rangés du plus exigeant au moins exigeant. Avec le seuil 10 en premier, toute note supérieure ou égale à 10 déclencherait immédiatement le niveau 1 et les niveaux suivants ne seraient jamais atteints.

4 | Calcul d'IMC

L'IMC (*Indice de Masse Corporelle*) est un indicateur des risques pour la santé en fonction du poids et de la taille; la formule est la suivante :

$$\text{IMC} = \frac{\text{poids(kg)}}{\text{taille}^2(\text{m})}$$

Question 9 : Rédiger un script qui demande la taille (en cm) et le poids (en kg) à l'utilisateur *libre à vous d'y répondre ce que vous voulez* et qui affiche dans la console l'IMC correspondant.

Réponse 9 :

```
taille_cm = input("Entrer la taille en cm : ")
poids = input("Entrer le poids en kg : ")
taille_m = int(taille_cm)*10**-2

IMC = float(poids)/taille_m**2
print("L'IMC correspondant est : ", IMC)
```

Question 10 : De quoi aurait-il fallu se méfier si on avait demandé la taille en mètre? *Peut être avez-vous eu ce problème en voulant entrer le poids.*

Réponse 10 : Il aurait fallu prendre en compte que les floats en Python utilisent le . et non la ,. Soit penser à répondre en utilisant un . soit remplacer la , par un ..

L'interprétation en fonction de la valeur de l'IMC est présentée dans le tableau 1. Cet indicateur peut être pratique mais il faut être conscient de ses limites : il ne prend pas en compte la répartition du poids entre muscle et graisse, devrait s'adapter en fonction du sexe, l'âge...

Valeur de l'IMC	Interprétation
IMC < 18,5	Maigre
18,5 ≤ IMC < 25	Corpulence normale
25 ≤ IMC < 30	Surpoids
30 ≤ IMC	Obésité

TABLEAU 1 – Interprétation de l'IMC

Question 11 : Compléter le script avec une structure if/elif/else qui affiche le message associé à l'intervalle contenant l'IMC. Tester au moins une valeur dans chaque intervalle.

Réponse 11 :

```
if IMC < 18.5:
    print("Insuffisance pondérale")
elif IMC < 25:
    print("Corpulence usuelle")
elif IMC < 30:
    print("Surpoids")
else:
    print("Obésité")
```

Dans chaque elif, les seuils précédents ont déjà été écartés. Il est donc inutile de répéter la borne inférieure de chaque intervalle.

5 | Construire des expressions booléennes

Question 12 : Écrire des expressions Python déterminant si chacune des deux quantités suivantes est strictement positive, puis relever les booléens obtenus.

$$\sqrt{1 + \sqrt{1 + \sqrt{1}}} - \frac{1 + \sqrt{5}}{2} \quad \text{et} \quad \frac{\sqrt{3}}{2} - \frac{2,7}{3,14}.$$

Réponse 12 :

```
test_1 = (1 + (1 + 1**0.5)**0.5)**0.5 - (1 + 5**0.5) / 2 > 0
test_2 = 3**0.5 / 2 - 2.7 / 3.14 > 0
print(test_1, test_2)
```

On obtient False puis True.

Question 13 : Écrire une expression booléenne utilisant a, b, c et d, vraie uniquement lorsque a est supérieure ou égale aux trois autres valeurs.

Réponse 13 :

```
a >= b and a >= c and a >= d
```

Question 14 : Soient $\Omega = (X, Y)$, $A = (x_A, y_A)$ et $r > 0$. Écrire une expression booléenne vraie lorsque A est strictement à l'intérieur du cercle de centre Ω et de rayon r . Tester $\Omega = (0, 0)$, $r = 2$, $A = (1, 1, 5)$, puis $\Omega = (1, 3)$, $r = 4$, $A = (5, 2)$.

Réponse 14 :

```
(xA - X)**2 + (yA - Y)**2 < r**2
```

Le premier test renvoie True, car $3,25 < 4$. Le second renvoie False, car 17 n'est pas strictement inférieur à 16.

6 | Défi de synthèse : qualifier une mesure de capteur

Cet exercice reprend le contexte du TP précédent et une partie des questions. L'objectif est de les traiter avec les conditions *if*.

Créer un nouveau script. Un convertisseur fournit un entier brut. Une valeur est valide si elle appartient à $[0; 1023]$. Pour une valeur valide,

$$U = 5 \frac{\text{brut}}{1023} \quad \text{et} \quad T = 100(U - 0,5).$$

La température est plausible dans $[-20; 80]$ et une alerte doit être affichée lorsqu'une température plausible atteint ou dépasse 30°C .

Question 15 : Écrire un programme qui demande brut à l'utilisateur. Utiliser une structure *if/else* pour écarter une valeur invalide, puis calculer et afficher U et T lorsque le code est valide.

Réponse 15 :

```
brut = int(input("Valeur brute : "))
if 0 <= brut <= 1023:
    U = 5 * brut / 1023
    T = 100 * (U - 0.5)
    print("U =", U)
    print("T =", T)
else:
    print("Code invalide")
```

Question 16 : Dans le bloc correspondant à une valeur valide, ajouter des instructions conditionnelles qui distinguent une température non plausible, une alerte et un fonctionnement normal.

Réponse 16 :

```
if 0 <= brut <= 1023:
    U = 5 * brut / 1023
    T = 100 * (U - 0.5)
    print("U =", U, "V")
    print("T =", T, "degres C")

    if -20 <= T <= 80:
        if T >= 30:
            print("Alerte")
        else:
            print("Fonctionnement normal")
    else:
        print("Temperature non plausible")
else:
    print("Code invalide")
```

Question 17 : Tester le programme avec brut = -1, brut = 100, brut = 164 et brut = 1100, puis compléter le tableau.

brut	T si elle est calculée	Message final
-1	NA	Code invalide
100	$T \simeq -1,12^{\circ}\text{C}$	Fonctionnement normal
164	$T \simeq 30,16^{\circ}\text{C}$	Alerte
1100	NA	Code invalide

Réponse 17 : Pour -1 et 1100, le message est Code invalide et la température n'est pas calculée. Pour 100, $T \simeq -1,12^{\circ}\text{C}$ et le fonctionnement est normal. Pour 164, $T \simeq 30,16^{\circ}\text{C}$ et l'alerte est affichée.