

Plaquette 1 - Variables, tests et fonctions

Algorithmique et Python — Première STMG

Corrections détaillées

Boîte à outils

Affectation. $x = x + 1$ calcule d'abord le membre de droite avec la valeur actuelle de x , puis range le résultat dans x . En langage naturel, on écrit $x \leftarrow x + 1$.

Instruction conditionnelle.

```
if condition:
    instructions
elif autre_condition:
    instructions
else:
    instructions
```

Comparaisons : $=$, $\neq$, $<$, $\leq$, $>$, $\geq$; on combine avec *and*, *or*.

Fonction. Les **entrées** sont les paramètres entre parenthèses, la **sortie** est la valeur renvoyée par *return*.

```
def ttc(ht, taux):
    return ht * (1 + taux / 100)
```

Une fonction qui **renvoie** sa valeur peut être réutilisée dans un calcul ; *print* ne fait qu'afficher.

Opérations utiles. // quotient entier, % reste, ** puissance, *round*(x , 2) arrondi au centième.

$$\text{Prix TTC} = \text{Prix HT} \times \left(1 + \frac{t}{100}\right)$$

Correction de l'exercice 1 - Suivre des affectations

On exécute les lignes une par une, en notant les valeurs.

Instruction	prix	remise
prix = 80	80	—
remise = 0.25	80	0,25
prix = prix * (1 - remise)	60	0,25
prix = prix * 1.2	72	0,25

À la ligne 3, $80 \times (1 - 0,25) = 80 \times 0,75 = 60$; à la ligne 4, $60 \times 1,2 = 72$. Python affiche 72.0, car le résultat d'un calcul avec des décimaux est un nombre décimal.

Le programme applique une remise de 25 % à un prix hors taxes de 80 €, puis ajoute la TVA de 20 % : le client paie 72 €. Les coefficients se multiplient :

$$80 \times 0,75 \times 1,2 = 80 \times 0,9 = 72$$

Réponse. *prix* vaut 72 : prix TTC après une remise de 25 %.

Correction de l'exercice 2 - Échanger deux variables

a) Après $a = b$, la variable a vaut 9 : l'ancienne valeur 5 est **perdue**. Puis $b = a$ donne $b = 9$. À la fin, $a = 9$ et $b = 9$: l'échange a échoué.

b) On sauvegarde d'abord la valeur de a :

```
t = a
a = b
b = t
```

Avec $a = 5$ et $b = 9$: t reçoit 5, puis a reçoit 9, puis b reçoit $t = 5$. À la fin, $a = 9$ et $b = 5$: les valeurs sont bien échangées.

Une affectation **écrase** l'ancienne valeur de la variable : il faut la sauvegarder si l'on en a encore besoin.

Réponse. a) $a = 9$, $b = 9$ — b) $t = a$, $a = b$, $b = t$.

Correction de l'exercice 3 - Des frais de port conditionnels

a) La fonction compare le montant à 50 :

- $port(35)$: $35 \geq 50$ est faux, la fonction renvoie 4,9 ;
- $port(50)$: $50 \geq 50$ est vrai, elle renvoie 0 ;
- $port(72.5)$: $72,5 \geq 50$ est vrai, elle renvoie 0.

b) Avec $montant > 50$, un panier d'exactly 50 € paierait les frais de port, puisque $50 > 50$ est faux. Or l'offre dit « à partir de 50 € », ce qui inclut 50 : la condition correcte est bien

$$montant \geq 50$$

Le choix entre $>$ et $\geq$ traduit exactement l'inclusion ou non de la valeur frontière.

Réponse. a) 4,9 ; 0 ; 0 — b) un panier de 50 € paierait le port, à tort.

Correction de l'exercice 4 - Une fonction de prix TTC

Ajouter une TVA de taux $t\%$ revient à multiplier le prix hors taxes par le coefficient

$$1 + \frac{t}{100}$$

La fonction a deux **entrées** (ht et $taux$) et une **sortie** (le prix TTC) :

```
def ttc(ht, taux):
    return ht * (1 + taux / 100)
```

$ttc(250, 20)$ calcule $250 \times (1 + 0,2) = 250 \times 1,2 = 300$; Python renvoie 300.0.

La même fonction sert pour tous les taux : $ttc(250, 5.5)$ donnerait le prix avec une TVA à 5,5 %.

Réponse. $ttc(250, 20)$ renvoie 300.

Correction de l'exercice 5 - Entrées et sortie d'une fonction

a) Les **entrées** sont les trois paramètres : le prix de vente unitaire, le coût d'achat unitaire et la quantité vendue. La **sortie** est la valeur renvoyée par *return* : la marge totale.

b) La fonction calcule la marge unitaire, puis la multiplie par la quantité :

$$(25 - 18) \times 40 = 7 \times 40 = 280$$

marge(25, 18, 40) renvoie 280 : en vendant 40 articles achetés 18 € et revendus 25 €, le commerçant dégage 280 € de marge.

Les noms des paramètres rendent la fonction lisible : on comprend son rôle sans lire le calcul.

Réponse. a) entrées : prix de vente, coût d'achat, quantité ; sortie : la marge — b) 280.

Correction de l'exercice 6 - Une remise par paliers

a) Python teste les conditions **dans l'ordre** et s'arrête à la première vraie :

- à partir de 500 €, la remise vaut 10 % du montant ;
- entre 200 € (inclus) et 500 € (exclu), elle vaut 5 % ;
- en dessous de 200 €, il n'y a pas de remise.

b) *remise(150)* : aucune condition n'est vraie, la fonction renvoie 0.

remise(300) : $300 \geq 500$ est faux, $300 \geq 200$ est vrai, donc $0,05 \times 300 = 15$.

remise(600) : la première condition est vraie, donc

$$0,1 \times 600 = 60$$

L'ordre des tests compte : si l'on testait d'abord $m \geq 200$, une commande de 600 € n'aurait que 5 % de remise.

Réponse. b) 0 ; 15 ; 60.

Correction de l'exercice 7 - Afficher ou renvoyer ?

solde1 **affiche** le prix soldé mais ne **renvoie** rien : en Python, elle renvoie la valeur spéciale *None*. L'addition *None* + *None* est impossible, et le programme s'arrête sur une erreur (après avoir affiché 28.0 et 42.0).

solde2 renvoie la valeur calculée, qui peut être réutilisée :

```
total = solde2(40) + solde2(60)
```

donne $40 \times 0,7 + 60 \times 0,7 = 28 + 42 = 70$.

$$(40 + 60) \times 0,7 = 70$$

À retenir. Une fonction de calcul doit **renvoyer** son résultat avec *return* ; *print* sert seulement à montrer une valeur à l'écran.

Réponse. *solde1* ne renvoie rien ; avec *solde2*, le total vaut 70.

Correction de l'exercice 8 - Une fonction de taux d'évolution

Le taux d'évolution se calcule avec la formule

$$t = \frac{V_A - V_D}{V_D} \times 100$$

d'où la fonction :

```
def taux(vd, va):
    return (va - vd) / vd * 100
```

taux(80, 100) : $\frac{100 - 80}{80} \times 100 = 25$, soit une hausse de 25 % (Python renvoie 25.0).

taux(250, 200) : $\frac{200 - 250}{250} \times 100 = -20$, soit une baisse de 20 %.

On divise par la valeur de **départ** : écrire / *va* donnerait des taux faux.

Réponse. 25 et -20 .

Correction de l'exercice 9 - Coefficients et calcul en machine

a) $cm(t)$ renvoie le coefficient multiplicateur d'une évolution de $t\%$. Le programme calcule le coefficient global d'une hausse de 10% suivie d'une baisse de 20% :

$$1,1 \times 0,8 = 0,88$$

soit une baisse globale de 12% .

b) L'ordinateur représente les nombres décimaux en binaire, et des nombres comme $1,1$ n'y ont pas d'écriture exacte : de minuscules erreurs d'arrondi apparaissent. Le résultat affiché diffère de $0,88$ d'environ 10^{-16} , ce qui est sans importance.

Pour afficher un résultat propre, on arrondit : `round(cm(10) * cm(-20), 4)` donne 0.88 .

Réponse. a) le coefficient global $0,88$ — b) erreur d'arrondi machine, corrigée par `round`.

Correction de l'exercice 10 - Corriger une fonction

L'erreur. La ligne `montant * 0.9` calcule bien $50 \times 0,9 = 45$, mais le résultat n'est rangé **nulle part** : `montant` garde sa valeur 50 , qui est renvoyée.

La correction. Il faut affecter le résultat :

```
def a_payer(montant, carte):
    if carte:
        montant = montant * 0.9
    return montant
```

Avec `a_payer(50, True)`, la fonction renvoie

$$50 \times 0,9 = 45$$

et `a_payer(50, False)` renvoie 50 . On peut écrire `if carte:` au lieu de `if carte == True:`, puisque `carte` est déjà un booléen.

Réponse. Il manque l'affectation `montant = montant * 0.9`.

Correction de l'exercice 11 - Un programme découpé en fonctions

a) Chaque fonction a un rôle simple et peut être vérifiée ou modifiée seule : si le prix de vente change, on ne modifie que `recette`. C'est la **programmation modulaire**.

b) $benefice(q)$ calcule $90q - (q^2 + 10q + 1200)$.

— $q = 10$: $900 - (100 + 100 + 1200) = 900 - 1400 = -500$;

— $q = 20$: $1800 - (400 + 200 + 1200) = 0$;

— $q = 40$: $3600 - (1600 + 400 + 1200) = 3600 - 3200 = 400$.

$$B(q) = -q^2 + 80q - 1200$$

Avec 10 articles, l'entreprise perd 500 € ; avec 20, elle est à l'équilibre ; avec 40, elle gagne 400 €.

Réponse. b) -500 ; 0 ; 400 .

Correction de l'exercice 12 - Deux conditions à la fois

La fonction renvoie un **booléen** : `True` si les deux conditions sont vraies, `False` sinon.

a) $prime(25, 1)$: $25 \geq 18$ est vrai, mais $1 \geq 2$ est faux, donc `False`.

$prime(30, 3)$: les deux conditions sont vraies, donc `True`.

$\text{prime}(17, 2)$: $17 \geq 18$ est faux, donc *False*.

b) Avec *or*, il suffirait qu’une condition soit vraie : $\text{prime}(25, 1)$ et $\text{prime}(17, 2)$ renverraient *True*. Un apprenti de 17 ans avec 2 ans d’ancienneté toucherait la prime, ce qui ne respecte pas la règle.

and:les deux or:au moins une

Réponse. a) *False* ; *True* ; *False* — b) une seule condition suffirait.

Correction de l’exercice 13 - Quotient et reste

$n // 24$ est le **quotient entier** de la division de 250 par 24, et $n \% 24$ en est le **reste** :

$$250 = 24 \times 10 + 10$$

a) Le programme affiche *10 10* : on remplit 10 cartons complets, et il reste 10 articles.

b) Les 10 articles restants nécessitent un carton supplémentaire, à moitié vide : il faut 11 cartons. En Python, on peut l’écrire $n // 24 + 1$ quand le reste n’est pas nul.

Contrôle : $10 \times 24 = 240$ articles en cartons pleins, plus 10 : on retrouve 250.

Réponse. a) *10 10* : 10 cartons pleins, 10 articles restants — b) 11 cartons.

Correction de l’exercice 14 - Du langage naturel à Python

La flèche $\leftarrow$ se traduit par le signe $=$ de l’affectation, et le test par *if ... else*.

```
def cout_total(q):
    c = 1500
    if q > 20:
        c = c + 35 * q
    else:
        c = c + 40 * q
    return c
```

Pour $q = 30$, la condition $30 > 20$ est vraie, donc

$$C = 1500 + 35 \times 30 = 1500 + 1050 = 2550$$

Le coût fixe est de 1500 €, et le coût unitaire baisse à 35 € au-delà de 20 unités (tarif de gros).

Réponse. $\text{cout_total}(30)$ renvoie 2550.

Correction de l’exercice 15 - Synthèse : établir une facture

On calcule le montant hors taxes, on applique la remise si nécessaire, puis la TVA :

```
def facture(q, pu):
    montant = q * pu
    if q >= 100:
        montant = montant * 0.95
    return round(montant * 1.2, 2)
```

$\text{facture}(120, 2.5)$: $120 \times 2,5 = 300$ € HT ; $120 \geq 100$, donc remise : $300 \times 0,95 = 285$ € ; TVA :

$$285 \times 1,2 = 342$$

$\text{facture}(80, 2.5)$: 200 € HT, pas de remise, $200 \times 1,2 = 240$ € TTC.

La fonction renvoie 342.0 et 240.0.

Réponse. 342 € et 240 €.