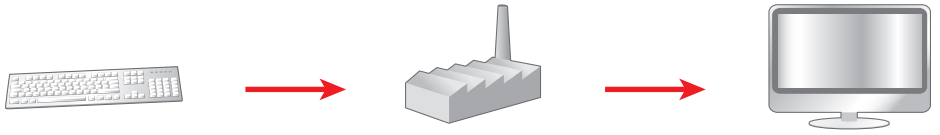
Vidéo
Contrôler le type et traiter une variable
hatier-clic.fr/ma2360

A.1 Variables : initialisation et traitement

Une **variable** est une « boîte » possédant un **nom** et dans laquelle est **stockée une valeur**.
Elle possède généralement un **type** : entier, flottant (nombre décimal), chaîne de caractères, etc.
Un **algorithme** manipule les variables, le plus souvent en trois phases :



INITIALISATION
Des valeurs sont **affectées aux variables** en prenant garde aux types choisis.

TRAITEMENT
Des **instructions** (calculs, test, etc.) sont effectuées sur les variables.

SORTIE
Les **résultats** contenus dans les variables sont affichés.

Savoir-faire Contrôler le type et traiter une variable

Le programme de calcul ci-contre peut être assimilé à un algorithme.

- Coder ce programme en Python.

- Choisir un nombre pair.
- Calculer le carré de sa moitié.
- Ajouter au résultat le nombre entier qui suit le nombre choisi.
- Calculer et afficher la racine carrée du résultat.

Solution

Deux variables peuvent suffire :

- *un entier N pour le nombre choisi par l'utilisateur ;*
- *un flottant R pour stocker le résultat obtenu à chaque étape.*

Langage naturel

1 $R \leftarrow \frac{N}{2} \times \frac{N}{2}$
2 $R \leftarrow R + (N + 1)$
3 $R \leftarrow \sqrt{R}$
4 Afficher R

python

```
1 import math
2 N=int(input("Entrez N pair :"))
3 R=(N/2)*(N/2)
4 R=R+(N+1)
5 R=math.sqrt(R)
6 print("Le résultat est :",R)
```

Cette première ligne permet d'importer des instructions mathématiques comme la racine carrée, et d'utiliser des constantes prédéfinies comme Pi.

`int` impose à N d'être un entier.

L'entier suivant N est $N+1$. Cette ligne se lit : « La variable R devient R augmentée de $N+1$. »

`print()` permet d'afficher plusieurs éléments de différents types.

À mon tour

Corrigés p. 383

1 Layla cherche la valeur à choisir pour que le résultat du programme de calcul ci-contre soit nul.

a. Coder ce programme en Python.

b. Déterminer par essais successifs la valeur cherchée.

- Choisir un nombre quelconque.
- Soustraire 4 à ce nombre.
- Multiplier le résultat par –5.
- Ajouter le nombre choisi au départ.

Exercice 6 p. 362

Vidéo
Programmer une instruction conditionnelle
hatier-clic.fr/ma2361

A.2 Instructions conditionnelles

Dans un algorithme, une **instruction conditionnelle** permet de différencier les tâches à effectuer en fonction d'une condition à valeur booléenne, c'est-à-dire vrai ou faux (▶ Rabat IV, Logique).
Cette **condition**, si elle est vraie, entraîne un **traitement**. Dans le cas contraire :

- soit l'algorithme se poursuit sans que le traitement ne soit exécuté :

```
Si la condition est vraie
    alors traitement
Fin Si
```

- soit un autre traitement est engagé :

```
Si la condition est vraie
    alors traitement n° 1
    sinon traitement n° 2
Fin Si
```

Savoir-faire Programmer une instruction conditionnelle

Une grossiste vend ses tomates par kilogramme entier, à 1,40 €/kg pour toute commande inférieure à 100 kg. Chaque kilogramme supplémentaire est facturé 1,10 €. Au-delà de 300 kg, elle offre une réduction de 10 % aux clients possédant la carte de fidélité.

a. Quel prix paiera un client pour une commande de 350 kg avec sa carte de fidélité ?

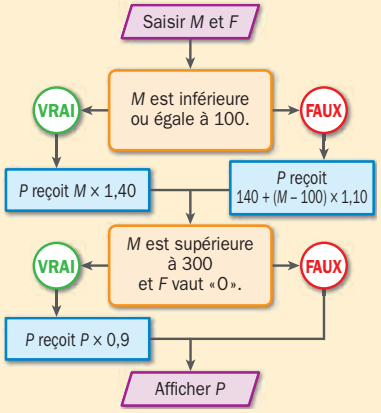
b. Construire un algorigramme affichant le prix d'une commande en fonction des informations entrées par le client, puis le coder en Python.

Solution

a. $100 \times 1{,}40 + (350 - 100) \times 1{,}10 = 415$ et $415 \times 0{,}9 = 373{,}50$.
Avec la réduction de 10 %, le client paiera 373,50 €.

b. L'algorithme nécessite trois variables : un entier M pour la masse de tomates, un flottant P pour le prix à payer et une chaîne F valant « O » ou « N » pour indiquer si le client possède la carte de fidélité.

Algorigramme



python

```
1  M=int(input("Masse ?"))
2  F=str(input("Carte ?(O/N)"))
3
4  if M<=100:
5      P=M*1.4
6  else:
7      P=140+(M-100)*1.1
8
9  if M>300 and F=="O":
10     P=P*0.9
11
12 print("Prix :",P,"€")
```

Réduire une quantité de 10 % revient à la multiplier par $\left(1 - \frac{10}{100}\right) = 0{,}9$.

Ne pas oublier de donner un type aux variables d'entrée (ici, entier et chaîne).

Le symbole <= teste l'inégalité « inférieure ou égale ». En Python, cette expression booléenne vaut « True » (vrai) ou « False » (faux).

Le symbole == teste l'égalité.

LOGIQUE
L'opérateur logique and permet de tester plusieurs conditions simultanément.
▶ Rabat IV, Logique

À mon tour

Corrigés p. 383

2 Les tarifs d'un peintre dépendent de la surface à traiter : 30 €/m² jusqu'à 100 m², puis 25 €/m² au-delà de 100 m². À cela s'ajoute la TVA : 10 % si l'habitation a plus de 2 ans, et 20 % sinon.

a. Quel est le prix à payer pour repeindre 150 m² dans une maison construite il y a 3 ans ?

b. Construire un algorigramme affichant le prix des travaux en fonction des informations entrées par le client, puis le coder en Python.

Exercice 7 p. 362

Vidéo
Programmer une boucle bornée
hatier-clic.fr/ma2366

B.1 Boucle bornée

Une **boucle bornée** permet la **répétition** d'une séquence d'instructions un **nombre fini de fois**.
Chaque **itération** (répétition de la séquence) est comptabilisée grâce à une variable appelée **indice**.
Cet indice varie entre **deux bornes**, un **pas** lui étant ajouté à chaque itération.

Pour indice allant de **borne 1** à **borne 2** avec un **pas** donné
    effectuer la séquence d'instructions
Fin Pour

Savoir-faire Programmer une boucle bornée

Chaque année, une banque augmente de 2 % les capitaux placés sur ses comptes. Après l'augmentation du capital, elle prélève sur chaque compte 20 € annuels de frais.
a. Quel capital aura-t-on au bout de deux ans en plaçant 3 000 € ?
b. Écrire un programme en Python affichant le capital final (arrondi au centime) en fonction de la somme placée et du nombre d'années écoulées, saisis par l'utilisateur.

Solution

a. *Au bout d'un an, le capital sera :*

$$3\,000 \text{ €} \times 1{,}02 - 20 \text{ €} = 3\,040 \text{ €}$$

Au bout de deux ans, le capital sera :

$$3\,040 \text{ €} \times 1{,}02 - 20 \text{ €} = 3\,080{,}80 \text{ €}$$

b. *Il faut deux variables : S (flottant) pour la somme placée et D (entier) pour la durée du placement. La variable S subit le même traitement chaque année : on utilise une boucle bornée dont l'indice A représente les années écoulées.*

python

```
1  S=float(input("Somme initiale :"))
2  D=int(input("Durée :"))
3  for A in range(1,D+1,1):
4      S=S*1.02-20
5  S=round(S,2)
6  print("Somme finale :",S)
```

L'augmentation du capital de 2 % s'obtient en le multipliant par 1,02.

En Python, la deuxième borne est exclue. Il faut donc écrire D+1 pour obtenir D itérations. Il était aussi possible d'écrire, plus simplement, `range(D)`.

L'indentation indique les instructions qui font partie de la boucle. Ici, l'arrondi au centième du capital final se calcule une fois les itérations de la boucle terminée.

À mon tour

Corrigés p. 383

1 Le 1er juillet, un Phyllostachys (bambou) a pour hauteur 100 cm. Le premier jour, il grandit de 80 cm. Puis sa croissance est chaque jour inférieure de 25 % par rapport à celle du jour précédent.
a. Quelle est la taille de ce bambou au bout de 3 jours ?
b. Écrire un programme en Python affichant la taille (en cm) de ce bambou en fonction du nombre de jours écoulés, entré par l'utilisateur.
c. Ce bambou dépassera-t-il 5 m au 1er septembre ?

Exercice 4 p. 368

Vidéo
Programmer une boucle non bornée
hatier-clic.fr/ma2367

B.2 Boucle non bornée

Une **boucle non bornée** est la **répétition** d'une séquence d'instructions, soumise à une **condition**. Tant que cette condition est vérifiée, la séquence est répétée.

Tant que condition **est vraie**
    effectuer la séquence d'instructions
Fin Tant que

Savoir-faire Programmer une boucle non bornée

Lors d'un rallye, le véhicule d'une concurrente subit un problème mécanique.
Le réservoir de son 4×4 a une fuite qui s'aggrave au fil des kilomètres : de 5 mL au premier kilomètre, celle-ci double tous les 4 km parcourus. Le véhicule contenait 60 L de carburant au moment de l'accident et il consomme en moyenne 10 L pour 100 km parcourus.
• Quelle distance approximative (au km près) la concurrente peut-elle parcourir avant de tomber en panne sèche ?
Résoudre cet exercice à l'aide d'un algorithme que l'on codera en Python.

Solution

60 – 0,1 – 0,005 = 59,895 ; après 1 km, il reste 59,895 L dans le réservoir.
59,895 – 0,1 – 0,005 = 59,79 ; après 2 km, il reste 59,79 L dans le réservoir.
59,79 – 0,1 – 0,005 = 59,685 ; après 3 km, il reste 59,685 L dans le réservoir.
59,685 – 0,1 – 0,005 = 59,58 ; après 4 km, il reste 59,58 L dans le réservoir.
La fuite s'aggrave et double : elle est à présent de 0,010 L par kilomètre.
59,58 – 0,1 – 0,010 = 59,47 ; après 5 km, il reste 59,47 L dans le réservoir.
Ce raisonnement se répète tant qu'il reste du carburant dans le réservoir.

5 mL = 0,005 L

Une consommation de 10 L pour 100 km donne, par proportionnalité, une consommation de 0,1 L pour 1 km.

Langage naturel

```
1  Carburant ← 60
2  Fuite ← 0,005
3  Distance ← 0
4  Tant que Carburant est supérieur à 0
5      Distance ← Distance + 1
6      Carburant ← Carburant – 0,1
7      Carburant ← Carburant – Fuite
8      Si Distance est divisible par 4
9          Fuite ← Fuite × 2
10     Fin Si
11 Fin Tant que
12 Afficher la Distance et le Carburant
```

python

```
1  Carburant=60
2  Fuite=0.005
3  Distance=0
4  while Carburant>0:
5      Distance=Distance+1
6      Carburant=Carburant-0.1
7      Carburant=Carburant-Fuite
8      if Distance%4==0:
9          Fuite=Fuite*2
10 print(Distance)
11 print(Carburant)
```

`Distance%4` donne le reste de la division euclidienne de `Distance` par 4. Si ce reste est nul, alors `Distance` est divisible par 4.

Le test sur la distance doit avoir lieu après la modification des variables : la fuite n'est doublée qu'une fois les 4 km parcourus.

Ces instructions sont extérieures à la boucle.

Le programme affiche 46 km pour `Distance` *et environ –6,02 pour* `Carburant`.
La panne sèche a donc lieu entre le 45e et le 46e km.

À mon tour

Corrigés p. 383

2 Un bambou de 1,80 m grandit quotidiennement de 10 %.
Chaque jour, un panda roux passe et en mange 20 cm.
• Combien de jours ce panda roux mettra-t-il pour manger entièrement ce bambou ?
Répondre à la question à l'aide d'un algorithme que l'on codera en Python.

Exercice 6 p. 368

Vidéo
Utiliser une fonction
hatier-clic.fr/ma2372

C Les fonctions

Une **fonction** est une séquence d'instructions, isolée du programme principal, qui n'est exécutée que quand elle est appelée. Elle peut admettre en entrée un ou plusieurs **paramètres**.
Après **traitement**, elle peut renvoyer un **résultat**.

Savoir-faire C.1 Utiliser une fonction

Le plan est muni d'un repère orthonormé.
a. Écrire en Python une fonction `DIST` dont les paramètres sont les coordonnées de deux points et qui renvoie la distance entre ces deux points.
b. Quel est le périmètre d'un triangle ayant pour sommets les points A(3 ; 4), B(−5 ; 8) et C(−3 ; −7) ? Répondre à l'aide d'une fonction `PERIMETRE` dont les paramètres sont les coordonnées des sommets d'un triangle et qui renvoie le périmètre de ce triangle, arrondi à 0,1 unité près.

Solution

a. La fonction `DIST` prend en paramètres les coordonnées $(X_1 ; Y_1)$ et $(X_2 ; Y_2)$ des points entre lesquels on souhaite calculer la distance.

python

```
1  import math
2
3  def DIST(X1,Y1,X2,Y2):
4      return math.sqrt((X2-X1)**2+(Y2-Y1)**2)
```

La distance entre deux points de coordonnées $(X_1 ; Y_1)$ et $(X_2 ; Y_2)$ est donnée par la formule $\sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$.
Chapitre 4

L'indentation marque le début et la fin de la fonction.

b. La fonction `PERIMETRE` prend en paramètres les coordonnées des trois sommets du triangle : $(X ; Y)$, $(U ; V)$ et $(Z ; T)$.

```
5  def PERIMETRE(X,Y,U,V,Z,T):
6      P=DIST(X,Y,U,V)+DIST(X,Y,Z,T)+DIST(Z,T,U,V)
7      return round(P,1)
```

La fonction `DIST` est appelée trois fois pour calculer la longueur de chacun des trois côtés du triangle.

On donne des valeurs aux paramètres et on affiche le résultat :

```
8  print(PERIMETRE(3,4,-5,8,-3,-7))
```

(ou dans la console :

```
>>> PERIMETRE(3,4,-5,8,-3,-7)
```

)

Le périmètre du triangle ABC vaut 36,6.

Il est possible d'appeler une fonction directement dans la console : dans ce cas, l'instruction `print` est superflue.

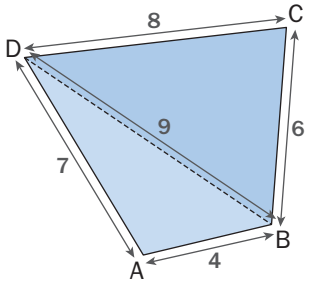
À mon tour

Corrigés p. 383

1 La formule de Héron permet de calculer l'aire d'un triangle :

$$\mathcal{A} = \sqrt{p(p-a)(p-b)(p-c)}$$

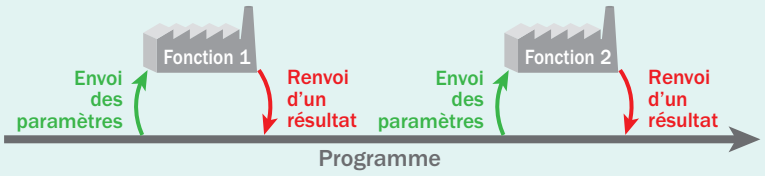
où a, b, et c sont les longueurs des trois côtés du triangle et p son demi-périmètre.
a. Écrire une fonction en Python qui calcule et renvoie l'aire d'un triangle à partir des longueurs de ses trois côtés.
b. Utiliser cette fonction pour calculer l'aire du quadrilatère ci-contre.



Exercice 3 p. 374

Vidéo
Décomposer un problème en sous-problèmes
hatier-clic.fr/ma2373

Une fonction permet de structurer un programme et de le décomposer en **sous-programmes** réutilisables.



Savoir-faire C.2 Décomposer un problème en sous-problèmes

• Y a-t-il plus de nombres premiers compris entre 2 et 5 000 qu'entre 5 001 et 10 000 ? Répondre à l'aide d'un programme en Python s'appuyant sur des fonctions.

Aide Chapitre 2, p. 29

• Un nombre entier est **premier** s'il a exactement deux diviseurs : 1 et lui-même.
Exemples • 5 est premier, mais 8 ne l'est pas : il est divisible par 2.
• 1 n'est pas premier, car il ne possède qu'un seul diviseur : lui-même.
• Pour déterminer si un nombre n est premier, il suffit de tester s'il possède un diviseur strictement supérieur à 1 et inférieur ou égal à $\sqrt{n}$. S'il n'en a pas, alors il est premier.

Solution

• Il faut d'abord écrire une fonction `EST_IL_PREMIER(nombre)` *qui renvoie le booléen « True » si le paramètre* `nombre` *est premier et « False » dans le cas contraire.*

```
1  import math
2  def EST_IL_PREMIER(nombre):
3      diviseur=2
4      while diviseur<=math.sqrt(nombre):
5          if nombre%diviseur==0:
6              return False
7          diviseur=diviseur+1
8      return True
```

`nombre%diviseur` renvoie le reste de la division euclidienne de `nombre` par `diviseur`. S'il est nul, alors `nombre` n'est pas premier : la fonction se termine immédiatement en renvoyant le booléen « False ».

Si la boucle se termine sans renvoyer « False », c'est qu'il n'y a pas de diviseurs : `nombre` est donc premier et la fonction renvoie le booléen « True ».

• Il faut ensuite créer une fonction `LISTE_PREMIER(debut,fin)` *qui renvoie la liste des nombres premiers compris entre* `debut` *et* `fin` *(inclus).*

```
9  def LISTE_PREMIER(debut,fin):
10     liste=[]
11     for nombre in range(debut,fin+1,1):
12         if EST_IL_PREMIER(nombre):
13             liste.append(nombre)
14     return liste
```

La fonction `EST_IL_PREMIER` renvoyant un booléen, il n'est pas nécessaire d'utiliser d'opérateur de comparaison (==, <, >).

• Enfin, il faut appeler `LISTE_PREMIER(2,5000)` *et* `LISTE_PREMIER(5001,10000)`*, puis comparer la taille des listes renvoyées.*

```
15 print(len(LISTE_PREMIER(2,5000)))
16 print(len(LISTE_PREMIER(5001,10000)))
```

L'instruction `len()` renvoie la longueur d'une liste, c'est-à-dire le nombre d'éléments de cette liste.

Ici, le programme affiche 669 et 560 ce qui permet d'affirmer qu'il y a plus de nombres premiers entre 2 et 5 000 qu'entre 5 001 et 10 000.

À mon tour

Corrigés p. 383

2 Un nombre entier est dit « abondant » s'il est strictement inférieur à la somme de ses diviseurs (hormis lui-même). Par exemple, 12 est abondant car 12 < 1 + 2 + 3 + 4 + 6.

a. Montrer que 40 est abondant et que 32 ne l'est pas.

b. Dresser, à l'aide de fonctions en Python, la liste de tous les nombres abondants inférieurs à 1 000.

Exercice 8 p. 374