Pages 25 à 29

EXERCICE 45

Comprendre un algorithme

Dans l'algorithme ci-contre, la variable m contient un nombre entier.
Selon la question posée, elle prendra différentes valeurs en début d'algorithme.

```
m ← .....
Tant que m > 9
    m ← m − 9
Fin Tant que
```

1. a. Si, en début d'algorithme, la variable m reçoit 5, quelle valeur contient-elle en fin d'algorithme ?

Elle contient la valeur 5.

b. Si en début d'algorithme, la variable m reçoit 15, quelle valeur contient-elle en fin d'algorithme ?

Elle contient la valeur 6.

2. Dans cette question, la valeur que reçoit la variable m en début d'algorithme est 30.
Compléter le tableau suivant et en déduire la valeur de m après l'exécution de cet algorithme.

Étapes	m	Condition vérifiée ?
Avant la boucle	30	oui
1er passage dans la boucle	21	oui
2e passage dans la boucle	12	oui
3e passage dans la boucle	3	non

En fin d'algorithme, la valeur de m est : 3

3. La variable m peut-elle contenir 21 en fin d'algorithme ?

Non, car tant que la valeur de m est strictement supérieure à 9, on reste dans la boucle « Tant que ».

Ainsi, la valeur de m en fin d'algorithme est inférieure ou égale à 9.

4. Quelles valeurs peut recevoir la variable m en début d'algorithme pour que la boucle « Tant que » soit parcourue ?

La valeur de m en début d'algorithme doit être strictement supérieure à 9.

EXERCICE 46

Comprendre un algorithme

En 2014, Carole verse sur son livret d'épargne 3 000 €.
Chaque année, la somme disponible sur le livret est multipliée par 1,03.
L'algorithme ci-contre permet de calculer l'année à partir de laquelle Carole disposera pour la première fois d'au moins 3 500 €.

```
S ← 3000
A ← 2014
Tant que S < 3500
    S ← S × 1,03
    A ← A + 1
Fin Tant que
```

1. Quelle variable contient les valeurs successives de l'épargne disponible ?

C'est la variable S qui contient les valeurs successives de l'épargne disponible.

2. Quel est le rôle de la variable A ?

La variable A contient les valeurs successives des années, depuis 2014 jusqu'à l'année solution.

3. Compléter le tableau suivant et en déduire la valeur que contient la variable A après l'exécution de cet algorithme.

Étapes	S	A	Condition vérifiée ?
Avant la boucle	3000	2014	oui
1er passage dans la boucle	3090	2015	oui
2e passage dans la boucle	3182,7	2016	oui
3e passage dans la boucle	3278,18	2017	oui
4e passage dans la boucle	3376,53	2018	oui
5e passage dans la boucle	3477,82	2019	oui
6e passage dans la boucle	3582,16	2020	non

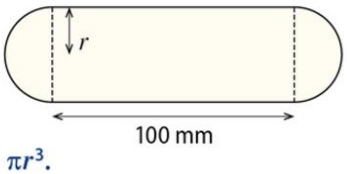
En fin d'algorithme, la valeur de A est : 2020

EXERCICE 47

Géométrie dans l'espace

Analyser une situation • Compléter et programmer un algorithme

Une ampoule destinée à recevoir du sérum est constituée d'un corps cylindrique de hauteur 100 mm et de deux demi-sphères de rayon r millimètres. On veut déterminer à partir de quelle valeur **entière** du rayon, **exprimé en millimètres**, le volume de l'ampoule dépasse 20 centilitres.



1. Montrer que le volume de l'ampoule, exprimé en mm³, est égal à $100\pi r^2 + \frac{4}{3}\pi r^3$.

Le volume du cylindre en mm³ est : $100 \times \pi r^2$.

Le volume de chaque demi-sphère en mm³ est $\frac{1}{2} \times \frac{4}{3}\pi r^3$, d'où le résultat annoncé.

2. Compléter l'algorithme ci-dessous afin qu'en fin d'algorithme, la variable r ait la valeur cherchée.

3. Programmer cet algorithme et répondre à la question posée.

```
def ampoule():
    r=0
    V=0
    while V<=200000:
        r=r+1
        V=100*pi*r**2+pi*r**3*4/3
    return(r)
```

$r \longleftarrow 0$
$V \longleftarrow 0$
Tant que $V \leqslant 200000$
  $r \longleftarrow r + 1$
  $V \longleftarrow 100 \times \pi \times r^2 + \pi \times r^3 \times 4/3$
Fin Tant que

La fonction ampoule retourne 23, donc le volume de l'ampoule dépasse 20 centilitres pour un rayon de 23 mm.

EXERCICE 48

Comprendre un algorithme

Le 1er janvier 2015, David a reçu 90 € d'étrennes, puis chaque année, celles-ci augmentent de 5 €. David décide de ne pas dépenser cet argent avant de disposer de 700 €.

1. Déterminer le montant E des étrennes que David reçoit le 1er janvier 2016.

David reçoit 95 €.

2. En déduire la somme totale T dont il dispose en 2016.

$T = 90 + 95 = 185$. En 2016, il dispose de 185 €.

3. L'algorithme ci-contre permet de calculer l'année à partir de laquelle David disposera d'au moins 700 €.

$E \longleftarrow 90$
$T \longleftarrow 90$
$A \longleftarrow 2015$
Tant que $T < 700$
  $E \longleftarrow E + 5$
  $T \longleftarrow T + E$
  $A \longleftarrow A + 1$
Fin Tant que

a. Faire fonctionner cet algorithme en complétant le tableau ci-dessous :

Étapes	E	T	A	Condition vérifiée ?
Avant la boucle	90	90	2015	oui
1er passage dans la boucle	95	185	2016	oui
2e passage dans la boucle	100	285	2017	oui
3e passage dans la boucle	105	390	2018	oui
4e passage dans la boucle	110	500	2019	oui
5e passage dans la boucle	115	615	2020	oui
6e passage dans la boucle	120	735	2021	non

b. En déduire à partir de quelle année, David pourra dépenser son argent.

L'année cherchée est 2021.

EXERCICE 49

Analyser une situation, compléter un programme et écrire un algorithme

Une entreprise de forage creuse des puits dans le désert afin d'atteindre la nappe d'eau phréatique.
Cette entreprise facture le premier mètre creusé 100 €, le second mètre 140 € et ainsi de suite en augmentant le prix de chaque nouveau mètre creusé de 40 €.

1. Calculer le prix M du troisième mètre creusé, puis le prix total S d'un puits de trois mètres de profondeur.

Prix du troisième mètre : $M = 180$ €. On a $S = 100 + 140 + 180 = 420$ €.

2. Compléter le programme de la fonction **puits** ci-contre, d'argument la profondeur H d'un puits (en mètres), afin qu'elle retourne le prix (en euros) de ce puits. Utiliser ce programme pour déterminer le prix d'un puits de 8 mètres de profondeur, puis celui d'un puits de 12 mètres de profondeur.

```
def puits(H):
    M=100
    S=100
    N=1
    while N<H:
        M=M+40
        S=S+M
        N=N+1
    return(S)
```

Un puits de 8 mètres de profondeur coûte 1 920 €.

Un puits de 12 mètres de profondeur coûte 3 840 €.

3. Une organisation humanitaire dispose d'un budget de 4 000 €.

a. En utilisant le programme de la question **2**, déterminer la profondeur maximale d'un puits que peut financer l'organisation.

Pour $H = 12$, le prix est de 3 840 €.

Pour $H = 13$, le prix est de 4 420 €.

La profondeur maximale est de 12 mètres.

b. Compléter l'algorithme ci-contre afin que la variable N contienne, en fin d'algorithme, cette profondeur maximale.

```
M ← 100
S ← 100
N ← 0
Tant que S ≤ 4000
    M ← M + 40
    S ← S + M
    N ← N + 1
Fin Tant que
```

EXERCICE 50

Compléter un programme

En 2016, les rejets polluants d'un groupe industriel sont évalués à 5 000 tonnes.
Le groupe est contraint de réduire ses rejets polluants de 8 % chaque année jusqu'à ce que ceux-ci ne dépassent pas 2 000 tonnes annuelles. On suppose que le groupe respecte ce plan de réduction.

1. Par quelle valeur est multipliée chaque année la quantité de rejets polluants ?

La quantité de rejets polluants est multipliée chaque année par 0,92.

2. La fonction **polluants**, programmée ci-contre en langage Python, a pour arguments la quantité annuelle r de polluants rejetés (en tonnes) et l'année n correspondant à ces rejets polluants. Compléter ce programme afin que la fonction **polluants** retourne en quelle année le groupe industriel atteindra pour la première fois son objectif.

```
def polluants(r,n):
    while r>2000:
        r=r*0.92
        n=n+1
    return(n)
```

3. Utiliser ce programme pour déterminer en quelle année l'objectif sera atteint.

polluants(5000,2016) renvoie 2027.

L'objectif sera donc atteint en 2027.

EXERCICE 51

Comprendre un programme

On donne ci-dessous deux fonctions mot1 et mot2 programmées en langage Python.
Les variables A et B contiennent des chaînes de caractères.

```
def mot1(A,B):
    L=A+B
    while len(L)<=8:
        L=L+A
        L=L+B
    return(L)
```

```
def mot2(A,B):
    L=A+B
    while len(L)<=8:
        L=L+A
    L=L+B
    return(L)
```

1. Que retourne mot1("pa", "pi") ? mot1("pa", "pi") retourne : 'papipapipapi'.

2. Que retourne mot2("pa", "pi") ? mot2("pa", "pi") retourne : 'papipapapapi'.

EXERCICE 52

Comprendre un programme

On donne ci-dessous trois fonctions calc1, calc2 et calc3 programmées en langage Python.
Les variables a, b et n contiennent des nombres entiers.

```
def calc1(a,b):
    n=0
    while n<20:
        n=a+b
        a=2*a
        b=3*b
        n=b*n
    return(n)
```

```
def calc2(a,b):
    n=0
    while n<20:
        n=a+b
        a=2*a
        b=3*b
    n=b*n
    return(n)
```

```
def calc3(a,b):
    n=0
    while n<20:
        n=a+b
        a=2*a
    b=3*b
    n=b*n
    return(n)
```

1. Compléter les calculs ci-dessous afin de déterminer ce que retourne calc1(5,1).

Avant le passage dans la boucle : n reçoit 0, a reçoit 5, b reçoit 1.

Lors du 1er passage dans la boucle : n reçoit 6, a reçoit 10, b reçoit 3 et n reçoit : $3 \times 6 = 18$.

Lors du 2e passage dans la boucle : n reçoit 13, a reçoit 20, b reçoit 9 et n reçoit : $9 \times 13 = 117$.

La condition $n < 20$ n'étant plus vérifiée, on sort de la boucle : calc1(5,1) retourne 117.

2. Que retourne calc2(5,1) ?

Avant le passage dans la boucle : n reçoit 0, a reçoit 5, b reçoit 1.

Lors du 1er passage dans la boucle : n reçoit 6, a reçoit 10, b reçoit 3.

Lors du 2e passage dans la boucle : n reçoit 13, a reçoit 20, b reçoit 9.

Lors du 3e passage dans la boucle : n reçoit 29, a reçoit 40, b reçoit 27.

La condition $n < 20$ n'étant plus vérifiée, on sort de la boucle.

n reçoit alors : $27 \times 29 = 783$. Donc calc2(5,1) retourne 783.

3. Que retourne calc3(5,1) ?

Avant le passage dans la boucle : n reçoit 0, a reçoit 5, b reçoit 1.

Lors du 1er passage dans la boucle : n reçoit 6, a reçoit 10.

Lors du 2e passage dans la boucle : n reçoit 11, a reçoit 20.

Lors du 3e passage dans la boucle : n reçoit 21, a reçoit 40.

La condition $n < 20$ n'étant plus vérifiée, on sort de la boucle.

b reçoit alors 3 et n reçoit : $3 \times 21 = 63$. Donc calc3(5,1) retourne 63.

EXERCICE 53

Comprendre un programme

Une petite ville organise chaque année un loto. En 2016, le nombre de participants à ce loto était de 200 et chaque participant devait payer 5 €. On fait l'hypothèse que d'une année sur l'autre, il y a 15 nouveaux participants et que la participation augmente de 0,50 €.

On donne ci-contre un programme écrit en langage Python.

```
def loto(a,j,p):
    s=j*p
    while s<=1500:
        j=j+15
        p=p+0.5
        s=j*p
        a=a+1
    return(a)
```

1. a. Que retourne loto(2016,200,5) ?

loto(2016,200,5) retourne 2019.

b. À quoi correspond cette valeur ?

Il s'agit de la première année où la recette dépassera 1 500 €.

2. On ajoute l'instruction « *a=a-1* » après la boucle while, comme sur la vue d'écran ci-contre.

```
    s=j*p
    a=a+1
a=a-1
return(a)
```

Que retourne désormais loto(2016,200,5) ? À quoi correspond cette valeur ?

loto(2016,200,5) retourne 2018. Cela correspond à la dernière année où la recette sera inférieure ou égale à 1 500 €.

EXERCICE 54

Comprendre et compléter un programme

On donne ci-contre une fonction mult1 programmée en langage Python.
L'argument n de cette fonction contient un entier naturel non nul inférieur à 100.

```
def mult1(n):
    k=n
    c=1
    while k<100:
        c=c+1
        k=c*n
    return(k)
```

1. Déterminer ce que retourne mult1(21) en complétant le tableau ci-dessous.

mult1(21) retourne 105

2. Quel est le rôle de cette fonction ?

Cette fonction retourne le plus petit multiple de n supérieur ou égal à 100.

Étapes	c	k	Condition vérifiée ?
Avant la boucle	1	21	oui
1er passage dans la boucle	2	42	oui
2e passage dans la boucle	3	63	oui
3e passage dans la boucle	4	84	oui
4e passage dans la boucle	5	105	non

3. a. Compléter le programme de la fonction mult2 ci-contre afin que cette fonction retourne la même valeur que la fonction mult1.

```
def mult2(n):
    k=n
    while k<100:
        k=k+n
    return(k)
```

b. Dans la console, on a exécuté la fonction mult1 pour différentes valeurs de n. Vérifier que la fonction mult2 renvoie les mêmes résultats.

```
>>> mult1(1)
100
>>> mult1(2)
100
>>> mult1(3)
102
>>> mult1(7)
105
>>> mult1(23)
115
```

```
>>> mult2(1)
100
>>> mult2(2)
100
>>> mult2(3)
102
>>> mult2(7)
105
>>> mult2(23)
115
```