

CORRECTION

Devoir à la maison n°2

Introduction : la conjecture de Syracuse

Le mathématicien allemand Lothar Collatz (1910-1990) inventa la construction d'une suite de nombres de la manière suivante :

- on part d'un nombre entier plus grand que zéro ;
 - s'il est pair, on le divise par 2 ;
 - s'il est impair, on le multiplie par 3 et on ajoute 1.

En répétant l'opération, on obtient une suite d'entiers positifs dont chacun ne dépend que de son prédécesseur. Par exemple, à partir du nombre 5, on construit la suite de Syracuse du nombre 5 :

5, 16, 8, 4, 2, 1, 4, 2, 1, 4, 2, 1...

Lothar Collatz énonça une conjecture en 1937 : une suite de Syracuse partant de n'importe quel entier strictement positif atteint toujours 1. A ce jour, aucun mathématicien n'a réussi à démontrer cette conjecture.

Exercice 1 : construction d'une suite de Syracuse à l'aide d'un algorithme

Un algorithme est une méthode générale pour résoudre un type de problèmes.

Quelques règles fondamentales :

- Un algorithme doit toujours se terminer après un nombre fini d'étapes.
- Chaque étape de l'algorithme doit être définie précisément pour chaque cas.
- Des valeurs sont données en entrée, dans un format précisé.
- Des valeurs sont calculées en sortie, dépendantes des valeurs données en entrée.

Cet algorithme consiste à écrire une suite de Syracuse et de ne s'arrêter que si le nombre 1 est atteint.

Etape 1 : Choisir un nombre entier N , différent de 1
Ecrire le nombre N

Etape 2 : Tant que $N \neq 1$ faire les opérations suivantes :
Si N est pair alors N prend la valeur $N \div 2$
Sinon N prend la valeur $3N + 1$
Ecrire le nombre N

Applique cet algorithme en partant du jour de ta naissance plus 32 (par exemple, si tu es né(e) le 8 alors tu démarres avec $N = 40$).

Exercice 2 : apprendre à programmer un algorithme de calcul

Dans l'exercice précédent il fallait construire à la main une suite de Syracuse.
Le programme **ALGOBOX** permet d'automatiser ce type de calculs répétitifs.

A - Prise en main d'AlgoBox : définir des variables, affecter des valeurs, afficher un résultat

Travail à faire : programmer et faire fonctionner l'algorithme suivant

Entrée : Saisir un nombre N
Traitement : a prend la valeur $N + 2$
 b prend la valeur $a \times 2$
 N prend la valeur $N + b$
Sortie : Afficher a
Afficher b
Afficher N

Pour $N = 1$
 a prend la valeur $1 + 2 = 3$
 b prend la valeur $3 \times 2 = 6$
 N prend la valeur $1 + 6 = 7$

Code de l'algorithme

```
1  VARIABLES
2  N EST_DU_TYPE NOMBRE
3  a EST_DU_TYPE NOMBRE
4  b EST_DU_TYPE NOMBRE
5  DEBUT_ALGORITHME
6  LIRE N
7  a PREND_LA_VALEUR N+2
8  b PREND_LA_VALEUR a*2
9  N PREND_LA_VALEUR N+b
10 AFFICHER "a ="
11 AFFICHER a
12 AFFICHER "b="
13 AFFICHER b
14 AFFICHER "N="
15 AFFICHER N
16 FIN_ALGORITHME
```

```
***Algorithme lancé***
Entrer N : 1
a =3
b=6
N=7
***Algorithme terminé***
```

B – Tester si un nombre est pair

Travail à faire : programmer et faire fonctionner l'algorithme suivant

Entrée : Saisir un nombre N
Traitement : a prend la valeur de la partie entière de $N \div 2$
Si $N = a \times 2$ **alors** :
Afficher « N est un nombre pair »
Sinon :
Afficher « N est un nombre impair »

Code de l'algorithme

```
1  VARIABLES
2  N EST_DU_TYPE NOMBRE
3  a EST_DU_TYPE NOMBRE
4  DEBUT_ALGORITHME
5  LIRE N
6  a PREND_LA_VALEUR floor(N/2)
7  SI (N==a*2) ALORS
8  DEBUT_SI
9  AFFICHER N
10 AFFICHER " est un nombre pair"
11 FIN_SI
12 SINON
13 DEBUT_SINON
14 AFFICHER N
15 AFFICHER " est un nombre impair"
16 FIN_SINON
17 FIN_ALGORITHME
```

C – Programmer un compteur

Algorithme à programmer

Entrée : Saisir un nombre positif N
Saisir un nombre positif t

Traitement : **compteur** prend la valeur 0
Tant que N est inférieur à 100 000 **faire** :
 $N = N \times t$
 Afficher N
 compteur prend la valeur **compteur + 1**
Afficher compteur

Code de l'algorithme

```

1  VARIABLES
2  N EST_DU_TYPE NOMBRE
3  compteur EST_DU_TYPE NOMBRE
4  t EST_DU_TYPE NOMBRE
5  DEBUT_ALGORITHME
6  LIRE N
7  LIRE t
8  compteur PREND_LA_VALEUR 0
9  TANT_QUE (N<100000) FAIRE
10 DEBUT_TANT_QUE
11 N PREND_LA_VALEUR N*t
12 AFFICHER N
13 compteur PREND_LA_VALEUR compteur+1
14 FIN_TANT_QUE
15 AFFICHER "Nombre de boucles = "
16 AFFICHER compteur
17 FIN_ALGORITHME
    
```

1) Que fait cet algorithme ?

Cet algorithme calcule le plus petit nombre entier a tel que $N \times t^a > 100\,000$

2) Que se passe-t-il si le nombre t est inférieur ou égal à 1 ?

Si a est inférieur ou égal à 1 le programme ne s'arrête jamais (ce n'est pas un algorithme)

D – Compléter l'algorithme de Syracuse ci-dessous et tester.

Travail à faire : **compléter**, **programmer** et **faire fonctionner** l'algorithme suivant.

Le fichier de l'algorithme doit être envoyé par email, en pièce jointe à l'adresse 2d3.mathxy@gmail.com afin d'être testé.

Qu'affiche cet algorithme en sortie ? **Cet algorithme affiche la suite de Syracuse du nombre entré et calcule la longueur de cette suite.**

Code de l'algorithme

```

1  VARIABLES
2  N EST_DU_TYPE NOMBRE
3  M EST_DU_TYPE NOMBRE
4  Compteur EST_DU_TYPE NOMBRE
5  DEBUT_ALGORITHME
6  ..... PREND_LA_VALEUR 0
7  LIRE .....
8  TANT_QUE ..... FAIRE
9  DEBUT_TANT_QUE
10 M PREND_LA_VALEUR floor(N/2)
11 SI (N==M*2) ALORS
12 DEBUT_SI
13 N PREND_LA_VALEUR .....
14 FIN_SI
15 SINON
16 DEBUT_SINON
17 N PREND_LA_VALEUR .....
18 FIN_SINON
19 AFFICHER N
20 Compteur PREND_LA_VALEUR .....
21 FIN_TANT_QUE
22 AFFICHER "NOMBRE de BOUCLES : "
23 AFFICHER .....
24 FIN_ALGORITHME
    
```

```

***Algorithme lancé***
Entrer N : 6
3
10
5
16
8
4
2
1
NOMBRE de BOUCLES : 8
***Algorithme terminé***
    
```