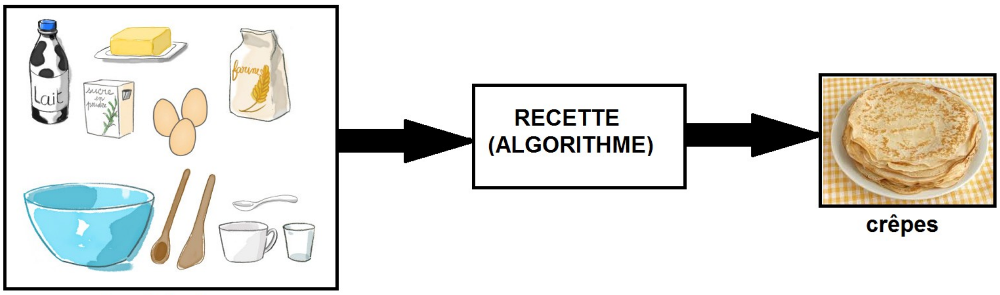
Chapitre 3

Quelques algorithmes

3.1. Définitions

Al Khwarizmi
(~780 – ~850)

Un algorithme est un énoncé d'une suite d'opérations permettant de donner la réponse à un problème.



Le mot « algorithme » vient du nom du mathématicien perse **Al Khwarizmi**, qui, au 9ème siècle écrivit le premier ouvrage systématique sur la solution des équations linéaires et quadratiques.

La notion d'algorithme est donc historiquement liée aux manipulations numériques, mais elle s'est progressivement développée pour porter sur des objets de plus en plus complexes : des textes, des images, des formules logiques, des objets physiques, etc.

3.2. La méthode de Héron

Héron
d'Alexandrie
(10-75)

En mathématiques, la méthode de Héron ou méthode de Babylone est une méthode efficace d'extraction de racine carrée.

Pour déterminer la racine carrée du nombre a, on choisit un nombre x_0 « assez proche » de $\sqrt{a}$, en général la partie entière de $\sqrt{a}$, puis on construit une suite définie par :

$$x_{n+1} = \frac{x_n + \frac{a}{x_n}}{2}$$

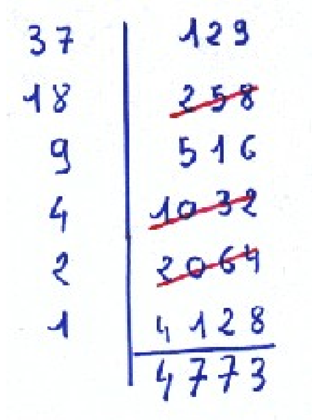
La suite ainsi obtenue est une suite décroissante à partir du second terme, convergeant vers $\sqrt{a}$.

La convergence est quadratique : l'écart entre chaque terme et la limite $\sqrt{a}$ évolue comme le carré de l'écart précédent, c'est-à-dire que le nombre de décimales exactes double à chaque itération.

Exercice 3.1

Programmez cet algorithme en Python et testez-le.

3.3. Multiplication à la russe



D'autres méthodes de multiplication en vidéo :

http://ow.ly/4tqcY

Il existe une méthode pour multiplier deux nombres où il ne faut que savoir multiplier ou diviser par deux, et additionner. On appelle cette méthode « multiplication à la russe ».

1. Dans la colonne de gauche, on divise par deux en prenant la partie entière et on s'arrête à 1.
2. Dans la colonne de droite, on double successivement chaque nombre.
3. On raye à droite tous les chiffres en face d'un nombre pair.
4. On fait la somme des nombres de droite restants.

Justification

Remplacer dans la colonne de gauche chaque nombre impair par 1 et chaque nombre pair par 0 revient à exprimer le nombre de gauche en base 2, si on lit de haut en bas. Les opérations effectuées sur la colonne de droite correspondent alors à une multiplication dans la base 2.

Exercice 3.2

Programmez cet algorithme en Python et estimez le nombre d'opérations moyen pour aboutir au résultat de la multiplication.

3.4. La date de Pâques

Algorithme de Thomas O'Beirne (validité : 1901-2099)

Soit M l'année du calcul (prenons 2005 comme exemple) :

« Pâques est le premier dimanche qui tombe immédiatement après le 14e jour de la (nouvelle) Lune qui tombe le 21 mars ou immédiatement après ».

La date de Pâques se situe entre le 22 mars et le 25 avril.

- On pose $n = M - 1900$ (on retranche 1900 à l'année, donc $n = 105$ pour notre exemple) ;
- On prend a, le reste de n dans la division par 19 (105 / 19 = 5 reste 10 ; $a = 10$) ;
- On calcule $a \times 7 + 1$ (ce qui donne pour l'exemple $7 \times 10 + 1 = 71$) ;
- On en prend b, le résultat entier de la division par 19 (71 / 19 = 3 donc $b = 3$) ;
- On calcule $(11 \times a) - b + 4$ (soit $11 \times 10 - 3 + 4 = 111$) ;
- On en prend c le reste de la division par 29 (111 / 29 = 3 reste 24, donc $c = 24$) ;
- On calcule d la partie entière de $n / 4$ (105 / 4 = 26) ;
- On calcule $n - c + d + 31$ (soit $105 - 24 + 26 + 31 = 138$) ;
- On en prend e le reste de la division par 7 (138 / 7 = 19 reste 5, donc $e = 5$) ;
- On calcule $P = 25 - c - e$ (dans l'exemple : $P = 25 - 24 - 5 = -4$) ;
- La date de Pâques tombe P jours après le 31 mars (ou avant si P est négatif). Ce qui signifie que :
  - $P = 1$ correspond au 1er avril. Autrement dit, P positif correspond au jour du mois d'avril ;
  - pour $P = 0$, le jour de Pâques est le 31 mars ;
  - pour $P = -1$, Pâques tombe le 30 mars. Autrement dit, P négatif doit être ajouté à 31 pour obtenir le jour du mois de mars.

Pour l'année 2005, on trouve $P = -4$, ce qui veut dire que Pâques tombait le $31 - 4 = 27$ mars.

En refaisant ce calcul pour l'année 2006, on trouve $P = 16$: Pâques 2006 tombait le 16 avril.

Exercice 3.3

Programmez cet algorithme en Python.

3.5. Calcul du jour de la semaine

Pour une date de la forme jour/mois/année où « jour » prend une valeur de 1 à 31, « mois » de 1 à 12 et « année » de 1583 à 9999, utiliser la formule :

c = (14 – mois)/12 (en fait, c = 1 pour janvier et février, c = 0 pour les autres mois)
a = année – c
m = mois + 12c – 2
j = (jour + a + a/4 – a/100 + a/400 + (31m)/12) mod 7

La réponse obtenue pour j correspond alors au jour de la semaine : 0 = dimanche, 1 = lundi, etc.

Deux remarques

1. Dans toutes les divisions "/", on ne garde que la partie entière du résultat. Par exemple, 35/4= 8.
2. « mod 7 » signifie que le résultat est le reste de la division par 7 (p. ex. 23 mod 7 = 2).

Exercice 3.4

Programmez cet algorithme en Python.
Entrée : le jour, le mois et l'année (avec année > 1582).
Sortie : le jour j de la semaine (0 à 6).
Exemple : jour_semaine(27,11,2014) → 4

Nouvelle notion : les listes

Comme son nom l'indique, une liste est une suite ordonnée de valeurs, qui peuvent être de types différents. Les éléments sont listés entre deux crochets et séparés par des virgules.
On peut créer directement une liste avec des éléments (ici des chaînes de caractères) :

```
mois = ["janvier", "février", "mars"]
```

On peut accéder à un élément en donnant son indice. Attention, on numérote à partir de 0. Par exemple `mois[1]` donnera comme résultat `février`.

Exercice 3.5

Dans un article publié dans le *Journal of Recreational Mathematics*, Vol. 22, No. 4, 1990, p. 280, Mike **Keith** (www.cadaeic.net) propose un autre algorithme pour déterminer le jour de la semaine correspondant à une date donnée :

Entrée
d : jour (d = 1, ..., 31)
m : mois (m = 1, ..., 12; 1 = Janvier, 2 = Février, 3 = Mars, ..., 12 = Décembre)
y : année (> 1582)
Sortie
j : jour de la semaine (j = 0 à 6; 0 = Dimanche, 1 = Lundi, ..., 6 = Samedi)

On pose $z = y - 1$, si $m < 3$ et $z = y$ sinon.
$k = [(23 \cdot m)/9] + d + 4 + y + [z/4] - [z/100] + [z/400]$
$j = k$ mod 7, si $m < 3$ et $j = (k-2)$ mod 7 sinon

[x] est le nombre entier résultant de la division euclidienne, le reste étant ignoré.
k mod 7 : reste de la division par 7

a. Programmez cet algorithme dans une fonction en Python.

Exemple : jour_semaine(20,11,2014) → 4

b. Utilisez une liste pour que la fonction retourne un nom de jour plutôt qu'un chiffre.
Exemple : jour_semaine(20,11,2014) → Jeudi

c. Modifiez votre programme pour afficher tous les vendredis 13 du 21ème siècle.

Exercice 3.6

Écrivez en Python un programme qui affiche les jours fériés d'une année donnée : Nouvel-An, 1er Août, la Toussaint et Noël, ainsi que les fêtes mobiles. Rappelons que les dates des fêtes mobiles sont fixées d'après la date de Pâques :

- Mardi-Gras (47 jours avant Pâques),
- Vendredi-Saint (2 jours avant Pâques),
- Pâques,
- Ascension (39 jours après Pâques),
- Pentecôte (49 jours après Pâques),
- Fête-Dieu (60 jours après Pâques).

Réutilisez les fonctions définies aux exercices 3.4 et 3.5 pour calculer la date de Pâques et le jour de la semaine.

Pour les fêtes mobiles, vous aurez besoin de deux nouvelles fonctions : la première convertira une date en un numéro du jour de l'année (par exemple, le 10 avril 2015 est le 100ème jour de l'année 2015), la seconde fera l'inverse (par exemple, le 101ème jour de l'année 2016 est le 10 avril). Vous pourrez grâce à ces deux fonctions calculer de nouvelles dates à partir de la date de Pâques.

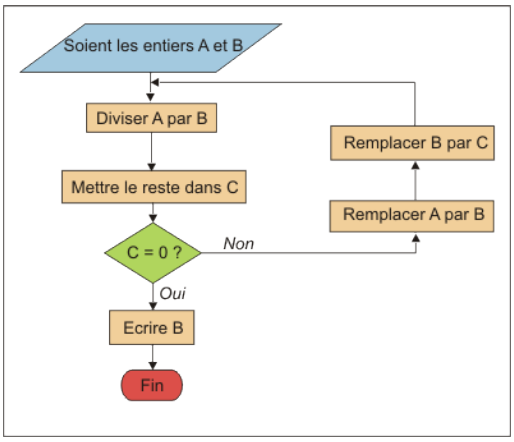
Voici le résultat que vous devriez obtenir pour 2016 :

Nouvel-An : Vendredi 1er janvier
Mardi Gras : Mardi 9 février
Vendredi Saint : Vendredi 25 mars
Pâques : Dimanche 27 mars
Ascension : Jeudi 5 mai
Pentecôte : Dimanche 15 mai
Fête-Dieu : Jeudi 26 mai
Fête nationale : Lundi 1er août
Toussaint : Mardi 1er novembre
Noël : Dimanche 25 décembre

Euclide
(325-265 av. J.-C.)

3.7. Algorithme d'Euclide

Donné ci-après sous forme d'organigramme (aussi appelé parfois ordinogramme), l'algorithme d'Euclide permet de trouver le plus grand diviseur commun de deux nombres A et B (avec A > B).



Exercice 3.7

Programmez l'algorithme d'Euclide en Python.

3.8. Fractions continues

En mathématiques, une fraction continue est une expression de la forme :

$$a_0+\cfrac{1}{a_1+\cfrac{1}{a_2+\cfrac{1}{a_3+\ldots}}}$$

comportant un nombre fini ou infini d'étages.

Pour transformer une fraction quelconque en un fractione continue, on va utiliser l'algorithme d'Euclide. Examinons le déroulement de l'algorithme, avec les deux nombres entiers 135 et 42. On procède à une suite de divisions euclidiennes avec reste :

$$\begin{aligned} 135 &= 3 \times 42 + 9 \\ 42 &= 4 \times 9 + 6 \\ 9 &= 1 \times 6 + 3 \\ 6 &= 2 \times 3 + 0 \end{aligned}$$

Une autre manière d'interpréter cet algorithme consiste à approcher par étapes le quotient 135 / 42. La partie entière de ce quotient est 3, ce qui permet d'écrire, selon la première division euclidienne effectuée :

$$135 = 3 \times 42 + 9 \Rightarrow \frac{135}{42}=3+\frac{9}{42}$$

Que peut-on dire de la fraction 9 / 42, à part qu'elle est plus petite que 1 ? Son inverse, 42 / 9, possède comme partie entière : 4 ; et plus précisément, si l'on utilise les résultats de la deuxième division euclidienne :

$$42 = 4 \times 9 + 6 \Rightarrow \frac{9}{42}=\cfrac{1}{4+\cfrac{6}{9}}$$

Ainsi, de proche en proche :

$$\frac{135}{42}=3+\frac{9}{42}=3+\cfrac{1}{4+\cfrac{6}{9}}=3+\cfrac{1}{4+\cfrac{1}{1+\cfrac{1}{2}}}$$

On utilise souvent la notation suivante, plus commode :

$$\frac{135}{42}=[3,4,1,2]$$

Les listes : l'instruction append

Pour définir une liste vide, l'instruction est :

```
ma_liste = []
```

Pour ajouter un élément en fin de liste, on a la méthode `append()` :

```
ma_liste.append(13)
```

donnera la liste `[13]`,

```
ma_liste.append(5)
```

donnera la liste `[13,5]`, etc.

Note : Il existe de nombreuses autres choses à savoir sur les listes, mais vous n'aurez besoin de rien de plus pour faire l'exercice 3.7.

Exercice 3.8

Écrivez en Python un algorithme qui transforme une fraction en une fraction continue. Le résultat sera donné sous la forme d'une liste.