# La boucle non bornée while

## Plan du T.P.



**Retour au sommaire**

## 1 La boucle while

L'une des tâches que les ordinateurs font le mieux est la répétition de tâches identiques. Il existe plusieurs méthodes pour programmer ces tâches répétitives, notamment la boucle Tant que.

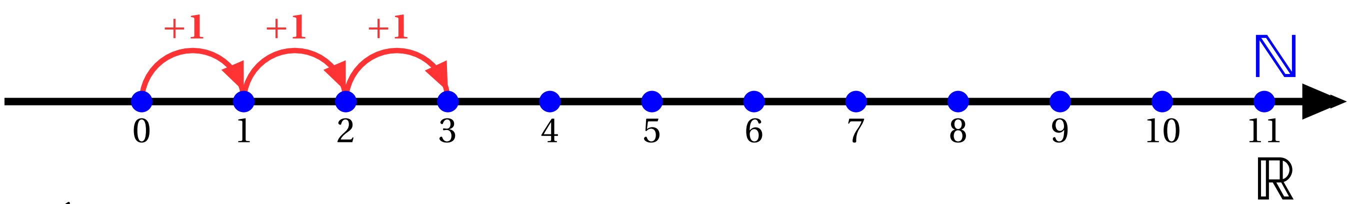
### 1.1 Manipuler

**Exercice 1 : Promenade sur les entiers**

La boucle "Tant que" commence à l'aide de l'instruction `while` :

```
nombre = 0
while nombre <= 5 :
    print(nombre)
    nombre = nombre + 1
```

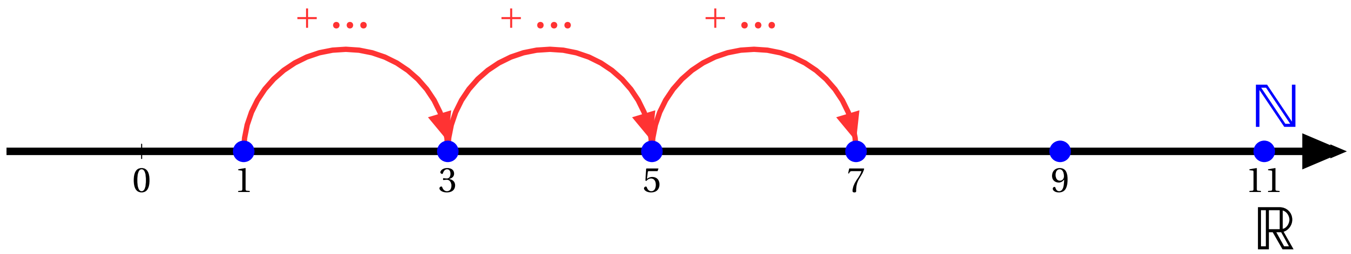
**a.** Compléter le schéma suivant :




**b.** Écrire un programme qui affichera successivement les nombres : 0, 1, 2, 3, . . . , 9, 10.

**Exercice 2 : Promenade impaire**

**a.** Nous souhaiterions maintenant parcourir les nombres impairs de 1 à 41.
Compléter le schéma suivant :



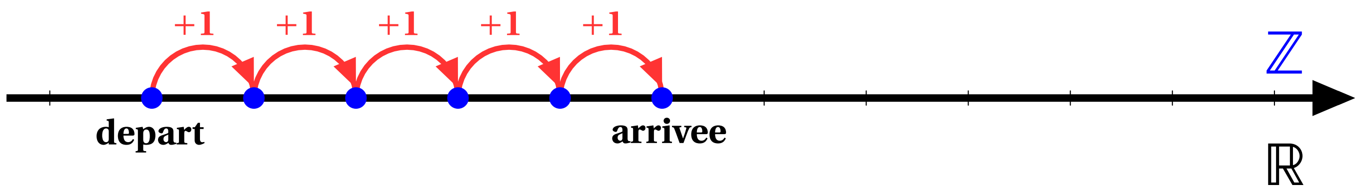

**b.** Écrire un programme qui affichera successivement les nombres impairs de 1 à 41 : 1, 3, 5, 7, 9, ... , 41.

**Exercice 3 : Fonction de promenade entre deux entiers**

Afin d'éviter de ré-écrire le même programme à chaque nouvelle suite de nombres, nous pouvons penser à une fonction.

**a.** Écrire une fonction, `promenade(depart, arrivee)` :

- prenant en argument deux nombres entiers : `depart` et `arrivee`
- et affichant tous les entiers de `depart` à `arrivee`



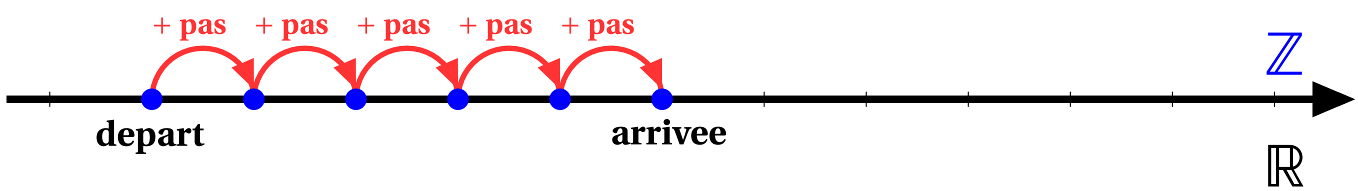

**b.** Tester votre fonction avec différentes valeurs.

**Exercice 4 : Avec des pas de longueur variable**

Nous souhaiterions maintenant pouvoir faire des pas de longueur variable (par exemple de longueur 2 pour parcourir des suites de nombres pairs ou impairs).

**a.** Écrire une fonction, `promenade2(depart, arrivee, pas)` :

- prenant en argument trois nombres entiers : `depart`, `arrivee` et `pas`,
- et affichant les entiers de `depart` à `arrivee`, en faisant des pas de longueur `pas`.




**b.** Tester votre fonction avec différentes valeurs.

## 1.2 Verbaliser

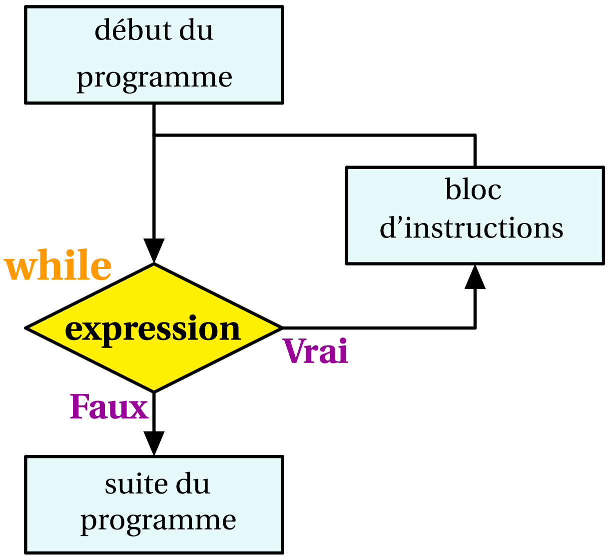
```
1 nombre = 0
2 while nombre <= 5 :
3     print(nombre)
4     nombre = nombre + 1
```

L'instruction `while`, à la ligne 2, indique à Python qu'il lui faut répéter continuellement le bloc d'instructions qui suit (lignes 3 et 4), tant que la condition "`nombre < 5`" est vraie.

Comme l'instruction `if` abordée au chapitre précédent, l'instruction `while` amorce une instruction composée. Le double point à la fin de la ligne 2 introduit **le bloc d'instructions à répéter**, lequel doit obligatoirement être décalé de 4 espaces : être **indenté**.

Lorsque la condition n'est plus vraie, le bloc indenté des lignes 3 et 4 est ignoré et le programme passe aux lignes suivantes, ou s'arrête s'il n'y a pas de suite.

## 1.3 Abstraire




### Syntaxe d'une boucle Tant que :

```
while condition_logique :
    bloc d'instructions
```

La fonction `print` que nous venons d'utiliser pour manipuler la boucle `while` offre juste un affichage, mais ne permet pas de renvoyer un résultat que nous pourrons ré-exploiter.

Nous utiliserons pour cela des **fonctions**, qui renvoient un résultat au moyen de **l'instruction** `return` :

### Syntaxe d'une fonction :

```
def nom_de_la_fonction(argument1, argument2, ...) :
    instructions
    return resultat
```

## 2 Problèmes de seuil

### Exercice 5 : Menuiserie

Une menuiserie produit et installe des escaliers en bois. Nous estimerons que le bénéfice mensuel en euros, réalisé lors de la fabrication de $x$ escaliers, est donné par la fonction :

$$f \text{ définie sur } \mathbb{R} \text{ par : } f(x) = 4000x + 2000$$

On donne ci-dessous le graphique de la fonction $f$ sur $[0\,;\,10]$.

**a.** Résoudre graphiquement l'inéquation :

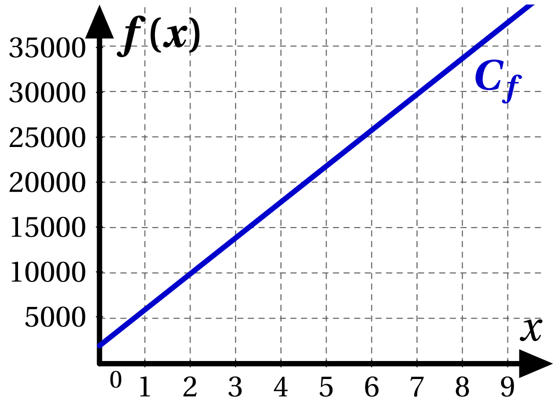
$$f(x) \geqslant 30\,000$$

..........................................

**b.** Quelle conclusion en tirer pour la menuiserie ?

..........................................

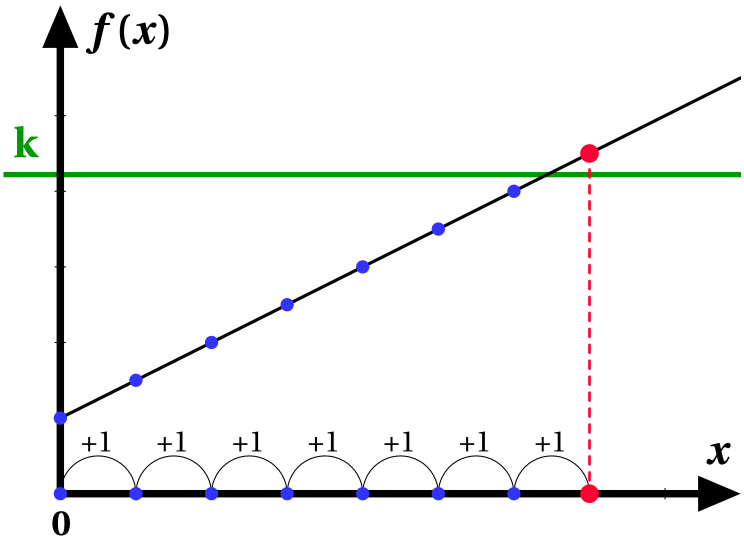
..........................................




**c.** Écrire en Python une fonction `f(x)` :
- prennant en argument un nombre $x$,
- et renvoyant l'image de $x$ par la fonction $f$.

**d.** Quel est le rôle de cette fonction ?

```
Fonction seuil(k) :
    x = 0
    Tant que f(x) < k :
        x ← x+1
    Retourner x
```




..........................................................................................

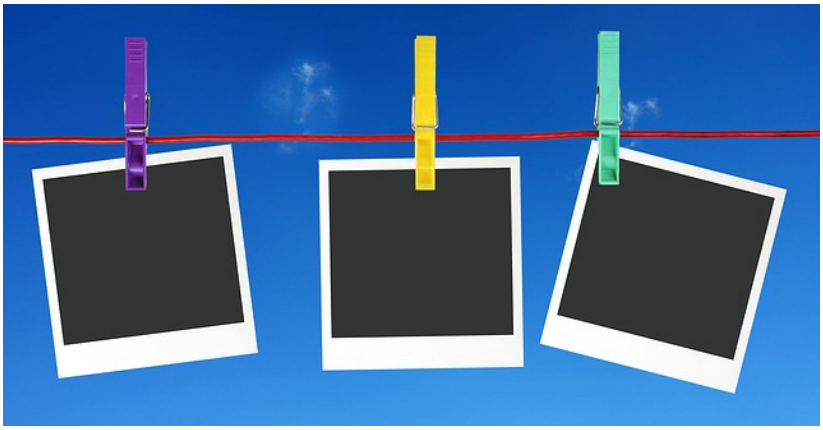
..........................................................................................

**e.** Écrire cette fonction en Python.

**f.** Combien d'escaliers la menuiserie doit-elle produire et installer dans le mois pour voir son bénéfice mensuel dépasser 100 000 euros ?

..........................................................................................

## Exercice 6 : un problème d'impression de photos



Un site d'impression de photos en ligne propose deux types de tarifs :

- tarif 1 : 0,5 € par photo,
- tarif 2 : un abonnement de 20 €, puis 0,35 € par photo.

Déterminer à partir de combien de photos il est intéressant de prendre l'abonnement.

### Traitement informatique

**1.** **a.** Écrire une fonction `tarif_1(x)`, prenant en argument le nombre $x$ de photos à imprimer et renvoyant le prix à payer par la première formule.

**b.** Utiliser votre fonction pour déterminer le prix d'impression de 65 photos par la formule 1.

**2.** **a.** Écrire une fonction `tarif_2(x)`, prenant en argument le nombre $x$ de photos à imprimer et renvoyant le prix à payer par la seconde formule.

**b.** Utiliser votre fonction pour déterminer le prix d'impression 65 photos par la formule 2.

**3.** Écrire une fonction `seuil()`, faisant appel aux fonctions des questions **1.** et **2.**, et renvoyant le nombre de photos à partir duquel il est intéressant de prendre l'abonnement.

### Traitement mathématique

Résoudre maintenant le problème à l'aide d'une méthode **mathématique**.

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

..................................................................................................

# 3 $\sqrt{2}$

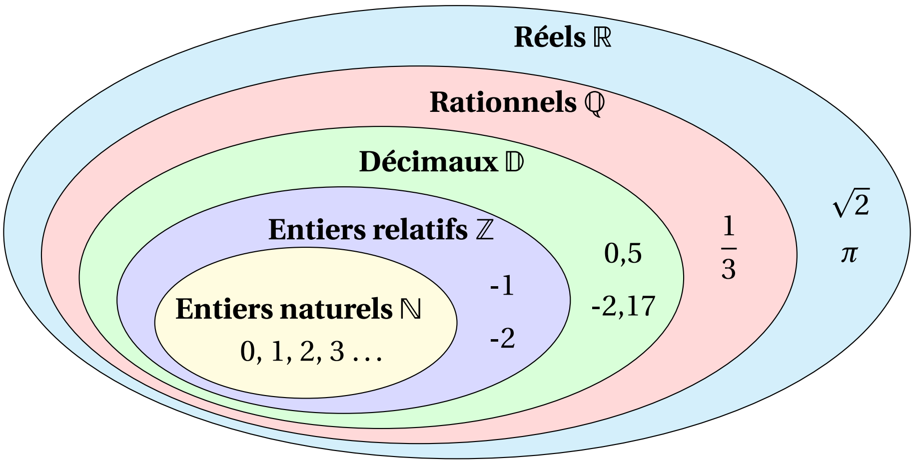
## 3.1 Les Pythagoriciens

**Pythagore, $\sim -580$ à $-495$, Grèce Antique**

L'École pythagoricienne [(1)], fondée par Pythagore [(2)] en Grande-Grèce, constitue une confrérie à la fois scientifique et religieuse. Le pythagorisme repose en effet sur une initiation et propose à ses adeptes un mode de vie éthique et alimentaire, ainsi que des recherches scientifiques sur le Cosmos.

Cette communauté politique et philosophique s'étale sur neuf à dix générations et a joui d'une très grande notoriété aussi bien dans l'Antiquité Grecque que Romaine. Ses membres explorèrent la science des nombres (arithmétique), les bases de l'acoustique et la théorie musicale, la géométrie, le mouvement des étoiles et la cosmologie.

## 3.2 Rappels sur les irrationnels

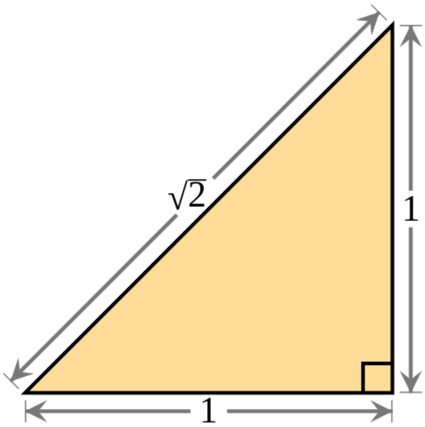



(1). https://fr.wikipedia.org/wiki//École_pythagoricienne
(2). https://fr.wikipedia.org/wiki/Pythagore

**Rappels**

- Un **nombre rationnel** est un nombre pouvant s'écrire sous la forme $\boldsymbol{\frac{a}{b}}$, où $a$ est un entier relatif et $b$ un entier naturel non nul.
- Un **nombre irrationnel** est un nombre qui n'est pas rationnel.
- $\boldsymbol{\sqrt{2}}$ est un nombre **irrationnel** (et $\sqrt{2} \simeq 1,414\ \ldots$).

## 3.3 La découverte des irrationnels

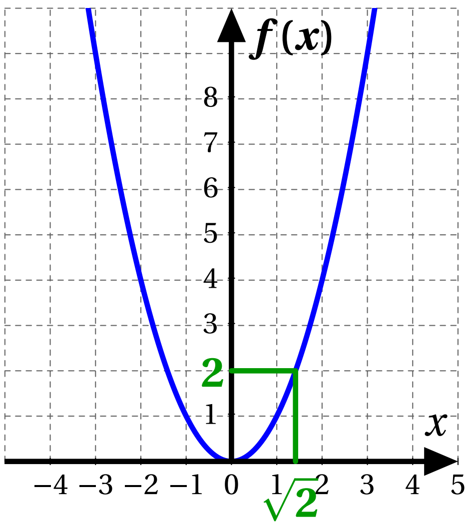



Pythagore découvrit des lois établissant un rapport entre la longueur de la corde d'une lyre, et la hauteur de la note émise (plus la corde est longue, plus le son est grave). De même, il tenta de développer une conception du Cosmos et de la matière dans laquelle, chaque mystère résiderait dans des rapports entre nombres : ce que nous appelons aujourd'hui des nombres rationnels.

La découverte du nombre $\sqrt{2}$, et d'autres nombres du même type, est à l'origine d'une crise profonde chez les mathématiciens grecs. En effet, $\sqrt{2}$ ne peut s'écrire comme un rapport $\frac{a}{b}$, remettant ainsi en cause la conception du Cosmos développée par les Pythagoriciens. L'émergence de ces nombres aboutira, des siècles plus tard, à la nécessité de créer de nouveaux nombres : les irrationnels.

## 3.4 Déterminer un encadrement de racine de 2 par balayage

### La fonction carré

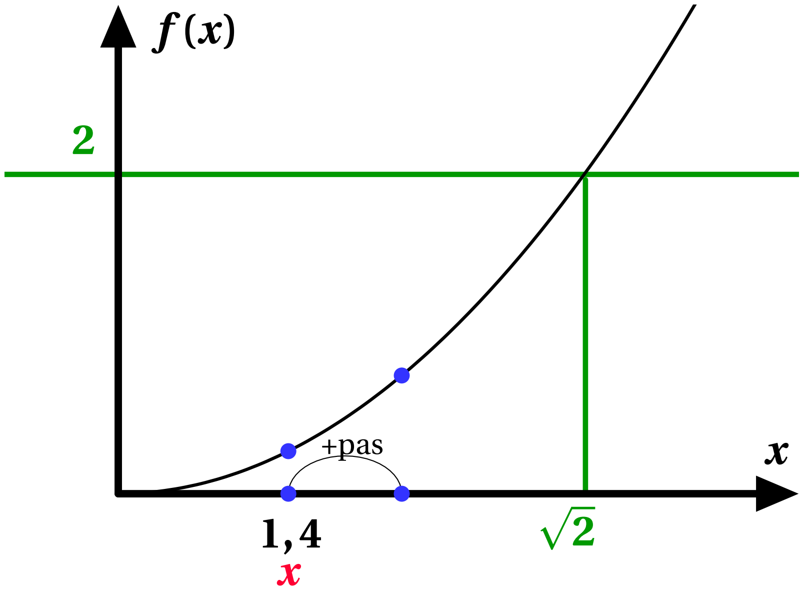



- La fonction carré est définie sur $\mathbb{R}$ par : $f(x) = x^2$ :
- Quelle est l'image par la fonction carré de $\sqrt{2}$ ? ............
- $(\sqrt{2})^2 = $ .........
- $\sqrt{2} \simeq$ .........

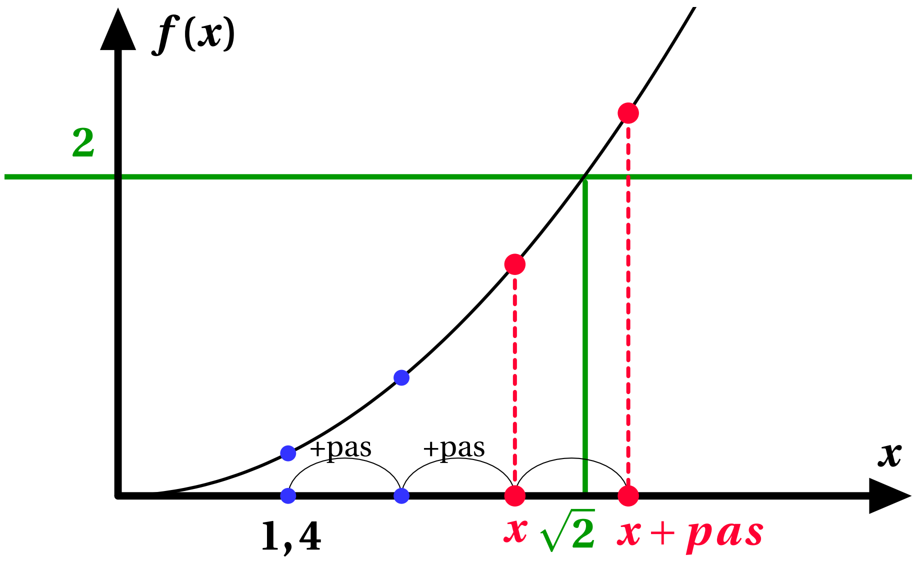
Afin d'obtenir une approximation de $\sqrt{2}$, nous allons maintenant "zoomer" sur la partie de la courbe se trouvant entre les abscisses $1,4$ et $\sqrt{2}$.

**Représentation imagée de l'algorithme**

- $x$ commence son voyage sur l'axe des réels en $1,4$,
- son image $x^2$ se déplace sur la courbe de la fonction carré :




- À force de faire des pas de longueur pas, il arrive un moment où :




**$(x + pas)^2$ dépasse 2, pour la première fois.**

## Approximation de $\sqrt{2}$ de précision donnée

**1.** La fonction ci-dessous permet d'obtenir une approximation de $\sqrt{2}$, avec une erreur inférieure à pas. La compléter.

```
• Fonction approximation(pas) :
      x = ...
      Tant que (x + pas) ** 2 < ... :
            ...............
      Retourner x
```

**2.** Écrire en Python une fonction `approximation(pas)` :

- renvoyant une valeur approchée de $\sqrt{2}$,
- avec une erreur inférieure à la variable `pas`.

**3.** Proposer une approximation de $\sqrt{2}$ à $10^{-6}$ près.

.............................................................................................

## Encadrement de $\sqrt{2}$

Pour renvoyer maintenant un encadrement, nous allons avoir besoin d'écrire une fonction renvoyant plusieurs valeurs :

- la borne inférieure de l'intervalle : $x$,
- et la borne supérieure de l'intervalle : $x + \text{pas}$.

Nous utiliserons une liste, qui se note en Python entre crochets, en séparant les valeurs par des virgule :

```
def encadrement(pas) :
    ...
    return [x,x+pas]
```

**4.** Écrire une fonction `encadrement(pas)` :

- prenant en argument un pas,
- et renvoyant les deux bornes d'un intervalle de longueur inférieure à pas, contenant $\sqrt{2}$.

**5.** Tester votre fonction avec des pas de valeurs : 0,1 puis 0,01 ...

## Arrondis

Notre fonction renvoie trop de chiffres après la virgule pour être facile à lire.

- avec un pas de longueur 0,1, nous aimerions qu'elle renvoie des nombres à une décimale,
- avec un pas de longueur 0,01, nous aimerions qu'elle renvoie des nombres à deux décimales ...

**6.** Pour obtenir des résultats avec un nombre adapté de chiffres après la virgule :

- Copier-coller la fonction suivante en début de programme :

```
from math import log
def arrondi(x, pas) :
    """ arrondir x au même nombre de décimales que pas """
    decimales = round(-log(pas)/log(10))
    return(round(x, decimales))
```

- Puis, remplacer la ligne commençant par `return` dans la fonction `encadrement` par :

```
return [arrondi(x, pas), arrondi(x+pas, pas)]
```

**7.** Proposer un encadrement de $\sqrt{2}$ à $10^{-6}$ près.

.............................................................................................

# 4 Modélisation d'une propagation épidémique

## Exercice 7 : modélisation d'une épidémie

Considérons une population humaine ou animale.

- au jour 0, trois individus sont contaminés par le virus d'une épidémie,
- chaque jour, le nombre d'individus contaminés triple.

**a.** Quelles sont les deux variables qu'il semble pertinent d'introduire ?

..........................................................................................

**b.** Écrire une fonction Python :

- prenant en argument l'effectif de la population,
- et renvoyant le premier jour où toute la population sera contaminée.

**Indication :** avant de commencer la boucle while, penser à créer les deux variables de la question **a**.

**c.** Utiliser la fonction pour déterminer au bout de combien de jours une population d'un million d'individus sera entièrement contaminée.

..........................................................................................

**d.** Notre modèle (→ triple tous les jours), est très simplifié. Mais il a déjà une caractéristique très nette : ce modèle vous semble-t-il correspondre à une épidémie se propageant rapidement ou lentement ?

..........................................................................................

**e.** Proposer en langage naturel un algorithme correspondant à une épidémie où le nombre d'individus contaminés augmente de 5% tous les jours.

..........................................................................................

..........................................................................................

..........................................................................................

..........................................................................................

..........................................................................................

..........................................................................................

..........................................................................................

**Ronald Ross : paludisme et modélisation**

Ronald Ross né en Inde en 1857 et meurt à Londres en 1932. Il est britannique, médecin bactériologiste et entomologiste de l'Armée des Indes britanniques. Il a reçu le prix Nobel de physiologie ou médecine pour ses travaux sur le paludisme.

Ses travaux établissent que la transmission du paludisme des oiseaux se fait par un moustique. Il estime par ailleurs, par la modélisation, le seuil de moustiques à éradiquer pour éradiquer le paludisme.

## Exercice 8 : propagation d'une rumeur

Imaginons un modèle très simplifié de propagation d'une rumeur au sein d'une population :

- au jour 1, un individu était au courant de cette rumeur,
- chaque jour, 3 nouveaux individus apprennent cette rumeur.

**a.** Quelles sont les deux variables qu'il semble pertinent d'introduire ?

.............................................................................................................

**b.** Écrire une fonction `rumeur(n)` :

- prenant en argument l'effectif total n de la population,
- renvoyant, à l'aide de l'instruction `return`, le premier jour où toute la population connaîtra cette rumeur.

**Indication :** avant de commencer la boucle while, penser à créer les deux variables de la question **a**.

**c.** Au bout de combien de jours une population de 16 000 individus sera intégralement atteinte par la rumeur ?

.............................................................................................................

## Synthèse

**Henri Poincaré :**

**« La mathématique est l'art de donner le même nom à des choses différentes. »**

Henri Poincaré né en 1854 à Nancy et meurt en 1912 à Paris. Il est mathématicien, physicien, philosophe et ingénieur français. Il est l'un des derniers mathématiciens à comprendre l'ensemble des savoir mathématiques de son époque. Il est aussi philosophe, et est souvent considéré comme un des derniers grands savants universels.

Les deux exercices que nous venons de traiter, modélisation d'une propagation épidémique et propagation d'une rumeur, donnent une illustration de l'une de ses affirmations restée célèbre : « La mathématique est l'art de donner le même nom à des choses différentes ».

En considérant une épidémie puis une rumeur, nous avons imaginé des modèles et des programmes informatiques presque identiques. Nous avons ainsi abstrait l'épidémie et la rumeur de leurs contextes respectifs. Nous avons finalement, en quelques sortes, donné le même nom à des choses différentes.