Exercice 1.

La suite de Syracuse de premier terme $s_0 \in \mathbb{N}$ est définie par la formule de récurrence

$$s_{n+1} = \begin{cases} \dfrac{s_n}{2} & \text{si } s_n \text{ est pair,} \\ \dfrac{3s_n+1}{2} & \text{sinon.} \end{cases}$$

1. (Sur papier.) Calculer la suite de Syracuse de premier terme 5, puis de premier terme 6 et enfin de premier terme 7. Que remarquez-vous ?

2. Ecrire une fonction `syr(s,n)` qui renvoie la liste des termes $s_0, \dots, s_n$ de la suite de Syracuse ayant s pour premier terme.

3. Représenter graphiquement la suite pour différents choix du premier terme.

4. Chercher sur Wikipedia des informations sur la « conjecture de Collatz ».

Exercice 2.

L'indien Srinivâsa Aiyangâr Râmânujan (1887–1920) était un mathématicien d'exception. Son collègue anglais Hardy rapporte l'anecdote suivante :

> Je me souviens que j'allais voir Râmânujan une fois, alors qu'il était malade. J'avais pris un taxi portant le numéro 1729 et je remarquais que ce nombre me semblait peu intéressant, ajoutant que j'espérais que ce ne fût pas mauvais signe.
> – Non, me répondit-il, c'est un nombre très intéressant : c'est le plus petit nombre admettant deux décompositions différentes en somme de deux cubes strictement positifs. En effet, $9^3 + 10^3 = 1^3 + 12^3 = 1729$.

Vérifiez la réponse de Râmânujan et trouvez le nombre le deuxième plus petit possédant cette propriété.

Indication : Pour $n \in \mathbb{N}^*$ fixé on pourra considérer la fonction $(a, b) \mapsto a^3 + b^3$ définie pour $a, b \in \mathbb{N}^*$ tels que $a \leqslant b$ et $a^3 + b^3 \leqslant n$. Il s'agit alors de trouver les couples distincts ayant même image par cette fonction.

Exercice 3.

Un entier naturel est *parfait* s'il est somme de ses diviseurs propres. Par exemple 6 et 28 sont parfaits car

$$6 = 1 + 2 + 3, \qquad 28 = 1 + 2 + 4 + 7 + 14.$$

1. Ecrire une fonction `sd(n)` qui renvoie la somme des diviseurs propres de n. On pourra utiliser un test de divisibilité avec la commande `mod`.

2. Déterminer les nombres parfaits inférieurs à 1000. Le programme est-il assez rapide pour trouver les nombres parfaits inférieurs à 10000 ?

3. Que donne la commande `divisors(n)` ? Utiliser `listify` pour transformer le résultat en liste. En utilisant ces deux commandes reprogrammer une fonction `sd(n)` plus rapide. Puis déterminer les quatre plus petits nombres parfaits.

Remarque – *Jusqu'à nos jours on ne sait pas s'il existe un nombre parfait impair.*

Exercice 4.

En 1742 le mathématicien allemand Christian Goldbach (1690-1764) écrivit une lettre au mathématicien suisse Leonhard Euler dans laquelle il proposait la conjecture suivante :

> *Tout nombre entier pair strictement supérieur à 2 peut être écrit comme la somme de deux nombres premiers (éventuellement deux fois le même).*

1. Écrire une fonction `goldbach(n)` qui renvoie, s'il en existe, un couple (k, m) de nombres premiers tel que $k + m = n$ et qui renvoie le couple vide, dans le cas contraire.

2. Programmer une fonction `Goldbach(n)` qui teste la conjecture de Goldbach jusqu'à n.
L'essayer pour $n = 1000$.

Remarque –*A ce jour la conjecture reste ouverte... En 2008 elle a été vérifiée par ordinateur pour tous les nombres pairs jusqu'à $1,1 \times 10^{18}$. En 2000, afin de faire de la publicité pour le livre* Uncle Petros and Goldbach's Conjecture *de Apostolos Doxiadis, un éditeur britannique offrit un prix de un miollion dollars pour une preuve. Il n'a jamais été réclamé.*

1. Solutions

Solution 1.

1. On trouve

$5,\ 8,\ 4,\ 2,\ 1,\ 2,\ 1,\ 2,\ \ldots$

$6,\ 3,\ 5,\ 8,\ 4,\ 2,\ 1,\ 2,\ 1,\ 2,\ \ldots$

$7,\ 11,\ 17,\ 26,\ 13,\ 20,\ 10,\ 5,\ 8,\ 4,\ 2,\ 1,\ 2,\ 1,\ 2,\ \ldots$

La suite semble toujours finir par alterner entre 1 et 2.

2.
```
syr(s,n):=block
([y,k,Y],
y:s,Y:[s],
      for k:1 thru n do
           (  if mod(y,2)=0 then y:y/2
              else y:(3*y+1)/2,
              Y:endcons(y,Y)  ),
      return(Y));
```

On essaie `syr(35,10);` et on obtient

$$[35, 53, 80, 40, 20, 10, 5, 8, 4, 2, 1]$$

3. On reprend le même programme en ajoutant les abscisses :

```
> Syr(s,n):=block([y,k,Y],
           y:s,Y:[[0,y]],
           for k:1 thru n do
            (  if mod(y,2)=0 then y:y/2
               else y:(3*y+1)/2,
               Y:endcons([k,y],Y)  ),
           return(Y));
```

On fait par exemple `plot2d([discrete,Syr(35,15)]);`

Solution 2.

On commence par une procédure `TaxiTriplets` qui à tout n donne la liste de tous les triplets de la forme $[a, b, k]$ avec $k \leqslant n$, $a \leqslant b$ et $a^3 + b^3 = k$.

```
TaxiTriplets(n):= block(  [a,b,L],
   L:[],
    for b:1 thru (n-1)^(1/3) do
        (a:1, while (a<=b and a^3+b^3<=n) do
          (L:endcons([a,b,a^3+b^3],L), a:a+1)),
          return (L));
```

On voit que dans `TaxiTriplets(1729)` le triplet contenant 1729 apparait effectivement deux fois. Mais pour voir plus clair il est mieux d'ordonner ces triplets.

```
TaxiOrdonne(n):=sort(
  TaxiTriplets(n), lambda([x, y], x[3] <= y[3])
);
```

Maintenant cherchons les doublons dans cette liste.

```
TaxiDoublons(n):= block(  [L,k,T],
  L:TaxiOrdonne(n), T:[],
  for k:2 thru length(L) do
  if (L[k-1][3]=L[k][3])
then T:endcons(L[k],endcons(L[k-1],T)),
return(T)
   );
```

Avec `TaxiDoublons(1729)` on voit que 1729 est le plus petit nombre de Râmânujan, et avec `TaxiDoublons(5000)` on trouve le suivant, à savoir 4104, avec ses décompositions

$$4104 = 2^3 + 16^3 = 9^3 + 15^3.$$

En une fraction de seconde la commande `Doublon(50000)` donne les nombres de Râmânujan plus petits que 50000 :

$$1729, 4104, 13832, 20683, 32832, 39312, 40033, 46683,$$

ainsi que leurs décompositions.

Solution 3.

1.
```
sd(n):=block(
  [S,k],
  S:0,
  for k:1 thru n/2
  do if mod(n,k)=0 then S:S+k,
  return(S) );
```

2. Avec la boucle

```
L:[]$
for n:2 thru 10000 do
if n=sd(n) then L:endcons(n,L)$
L;
```

on trouve que les nombres parfaits inférieurs à 1000 sont 6, 28 et 496. Pour aller à 10000 le programme semble ramer…

3. La commande `divisors(12)` donne l'ensemble (pas la liste) des diviseurs de n.

```
sd(n):=block(    [S,L,k],
          L:listify(divisors(n)),S:0,
          for k:1 thru length(L)-1
          do S:S+L[k],
          return(S) );
```

Alternativement on peut utiliser la commande `sum` de Maxima :

```
SD(n):=sum(listify(divisors(n))[k],k,1,length(listify(divisors(n)))-1);
```

On trouve que les quatre nombres parfaits les plus petits sont $6, 28, 496, 8128$.

Solution 4.

```
goldbach(n):= block(
 [k,couple],
 k:2, couple:[],
 while (couple=[] and k<=n/2) do
    (     if primep(n-k) then couple:[k,n-k]
          else k:next_prime(k)
    ),
      return(couple));
```

On peut tester `goldbach(4)` ou `goldbach(100)`. Avec `goldbach(11)` on voit que la condition que n est impair est essentielle.

```
Goldbach(n):= block
(       [k,c],
   k:4, c:[qqchose],
   while (k<=n and not(c=[])) do
     (c:goldbach(k),k:k+2),
  if c=[] then print("Conjecture fausse")
  else print ("Conjecture vérifiée jusqu'à ",n)
)   ;
```