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 $s_0, \ldots, s_n$ 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.

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 3.

Un *nombre de Stern* est un nombre premier qui n'est pas la somme d'un nombre premier et du double du carré d'un nombre entier non nul. Autrement dit, un nombre premier p est de Stern s'il ne se laisse pas écrire comme $p = q + 2k^2$ avec q premier et $k \in \mathbb{N}^*$.

1. Écrire une fonction `stern(p)` qui, pour un nombre premier p donné, renvoie 1 s'il est de Stern et 0 sinon. Vérifier que $2, 3, 17$ et 137 sont de Stern.

2. Écrire une fonction `Stern(n)` qui, pour un entier naturel n donné, renvoie la liste des nombres de Stern inférieurs à n.
Trouver les huit plus petits nombres de Stern.

Commandes utiles : `primep` *et* `next_prime`.

Remarque – *De nos jours on n'a pas trouvé d'autres nombres de Stern que ces huit là !*

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 million dollars pour une preuve. Il n'a jamais été réclamé.*

1. Solutions

Solution 1.

1. On trouve

$5,\ 8,\ 4,\ 2,\ 1,\ 2,\ 1,\ 2,\ \dots$
$6,\ 3,\ 5,\ 8,\ 4,\ 2,\ 1,\ 2,\ 1,\ 2,\ \dots$
$7,\ 11,\ 17,\ 26,\ 13,\ 20,\ 10,\ 5,\ 8,\ 4,\ 2,\ 1,\ 2,\ 1,\ 2,\ \dots$

Il semble que la suite finit par alterner entre 1 et 2, peu importe le choix de la valeur initiale.

2.
```
syr(s,n):=block(
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: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(83,60)]);` pour un bel envol !

Solution 2.

1.
```
sd(n):=block(
  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(
          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 3.

1. On fait une boucle sur k. La variable qui sortira t vaut 1 au début, puis elle devient 0 dès que $p-2k^2$ est premier.

```
stern(p):= block(  k:1, t:1,
 while (p-2*k^2>0 and t=1) do
 if primep(p-2*k^2) then t:t-1
 else k:k+1,
 return(t)    )
```

2. On fait une boucle sur p.

```
Stern(n):=block( p:2, L:[],
while p<=n do
(if stern(p)=1 then L:endcons(p,L),
p:next_prime(p)),
return(L)    );
```

La commande `Stern(2000)` donne les huit plus petits nombres de Stern 2, 3, 17, 137, 227, 977, 1187, 1493.

Solution 4.

```
goldbach(n):= block(
 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: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)
)   ;
```