Maple TD4

1 Correction des exercices précédents

1.1 Manipulations de séquences

```
n:=20; s1 := seq(2*i,i=0..n-1);

max:=proc(s)
local m,i; m:=s[1];
for i from 1 to nops(s) do if s[i] > m then m := s[i] end if; od; return m; end proc;

somme:=proc(s)
local resultat,i; resultat:=0; for i from 1 to nops(s) do resultat := resultat + s[i] od;
return resultat; end proc;

sommepair := proc(n) return somme(seq(2*i,i=0..n-1)); end proc;
```

1.2 Dessin élémentaire

```
myplot:=proc(f,a,b,delta)
l:=[a,f(a)]:
for x from a to b by delta do l:= l,[x,f(x)]: end do:
plot([l]);
end proc;
myplot(x->x^2,0,2,0.1);
```

1.3 Triangle de Sierpiński

1. `s:=[[0,0],[1,0],[0.5,sqrt(3)/2],[0,0]];`
2. ```
   translation:=proc(p,vx,vy)
     return [p[1]+vx,p[2]+vy]
   end proc;
   ```
3. ```
   translationSeq:=proc(s,vx,vy)
     return [seq(translation(s[i],vx,vy),i=1..nops(s))]
   end proc;
   ```
4. Écrire une procédure `suivant(s,n)` prenant une séquence s décrivant le n-ième triangle de sierpinski et renvoyant la séquence décrivant le triangle suivant.

   ```
   suivant:=proc(s,n)
   local s1,s2;
   s1 := translationSeq(s,2^(n-1),0);
   s2 := translationSeq(s,2^(n-2),sqrt(3)*2^(n-2));
   return [op(s),op(s2),op(s1),[0,0]];
   end proc;
   ```
5. Écrire une procédure `sierpinski(n)` utilisant les fonctions précédentes et renvoyant les points du n-ieme triangle de Sierpiński et l'afficher.

   ```
   sierpinski:=proc(n)
   local s,i;
   s:=[[0,0],[1,0],[0.5,sqrt(3)/2],[0,0]];
   for i from 2 to n do
   s:=suivant(s,i);
   ```

```
od;
plot(s);
end proc;
```

2 Exercices

2.1 Suites

1. Ecrire la fonction *babylone* tel que $babylone(x) = \frac{x+\frac{a}{x}}{2}$
2. Par définition, la suite de Syracuse est :

$$\begin{cases} u_0 \text{un entier strictement positif} \\ u_{n+1} = 3u_n + 1 \text{ si } u_n \text{ est impair} \\ u_{n+1} = \frac{u_n}{2} \text{ si } u_n \text{ est pair} \end{cases}$$

   Ecrire une fonction *syracuse* prenant en argument u_n et renvoyant u_{n+1}
3. Ecrire une procédure prenant en paramètre une fonction f, un réél r et effectuant le tracé de la ligne joignant les points (k, v_k) où $v_0 = r$ et $v_{i+1} = f(v_i)$. Afficher la suite de syracuse et celle définit par la fonction babylone.
4. Ecrire une procédure prenant les même paramètres et effectuant le tracé de la ligne joignant les points (v_0, v_0), (v_0, v_1), (v_1, v_1), (v_1, v_2). Vous pourrez utiliser la fonction map. Ajoutez au dessin les droites $y = x$, $y = \frac{x}{2}$ et $y = 3x + 1$. Afficher la suite de syracuse et celle définit par la fonction babylone.

2.2 Trajectoires

Soit un boulet de canon de masse m tiré avec une vitesse initial $v = (vx, vy)$.

1. Considérons qu'il n'y a pas de frottements, écrire les équations différentielles régissant la vitesse.
2. Les résoudre avec Maple. (utilisez dsolve)
3. Tracer $v_y(t)$.
4. Ecrire les équations différentielles régissant la position du boulet.
5. Les résoudre avec Maple.
6. Tracer la trajectoire.
7. On suppose maintenant que le vent ajoute une force F_{vent} dans tout l'espace. Reprendre les questions précédentes en en tenant compte.
8. Ecrire une procédure ayant comme paramètres m, vx, vy et F_{vent} et affichant la trajectoire du boulet. Jouez avec les différents paramètres.
9. Idem en rajoutant une force de frottement F_v de valeur une fonction f de la vitesse.

2.3 Pendule

On va maintenant utiliser l'option de dsolve type=numeric. L'équation régissant un pendule peut parfois s'écrire : $y'' = -sin(y)$.

1. Essayer de la résoudre avec Maple.
2. Essayer avec l'option type=numeric. Sous quelle forme est donnée la solution ? Si Sol est le résultat, que vaut S(1) ?
3. Tracer la courbe $y(t)$ avec odeplot.
4. Tracer la courbe $y(t)$ sans utiliser odeplot (vous aurez besoin de subs et plot).
5. Trouver la période avec fsolve.