L'épreuve pratique de maths du Bac S sans tableur...

Guillaume CONNAN
http://gconnan.free.fr

18 juin 2008

I - Sujet 003

On va construire une procédure `toscane(n)` qui va réaliser `n` simulations de 10 000 lancers de trois dés en renvoyant :
- les fréquences de sortie de 9 et 10 ;
- les probabilités de gain de Bob et Alice ;
- les boîtes à moustaches correspondant à chaque événement.

1. Commandes utilisées

rand(n) : cette commande renvoie un entier n vérifiant $0 \leqslant n < n$. Par exemple

```
rand(2), rand(2), rand(2), rand(2), rand(2)
```

$$1,1,0,0,1$$

Pour simuler le lancer d'un dé cubique, nous utiliserons donc :

```
rand(6)+1
```

$$6$$

ranm(n,p,'tirage') : construit un tableau de n lignes et p colonnes, chaque cellule contenant l'issue d'un tirage. Par exemple :

```
T:=ranm(2,15,'rand(6)+1')
```

$$\begin{pmatrix} 5 & 6 & 4 & 6 & 1 & 2 & 6 & 5 & 4 & 1 & 3 & 4 & 2 & 4 & 3 \\ 1 & 2 & 4 & 5 & 2 & 4 & 4 & 5 & 5 & 4 & 6 & 3 & 6 & 1 & 4 \end{pmatrix}$$

donne 2×15 tirages d'un dé cubique.

count_eq(n,T) : compte le nombre d'occurrences de n dans le tableau T. Par exemple :

```
count_eq(4,T)
```

$$9$$

compte le nombre de sorties de 4 dans le tableau T.

moyenne(L) : calcule la moyenne des éléments d'une liste :

```
moyenne([16,15,12,10,19,7])
```

$$\frac{79}{6}$$

Attention ! T n'est pas une liste mais un tableau, c'est-à-dire une liste de listes. La n^e ligne étant obtenue en entrant `T[n-1]` car XCAS commence à compter à partir de 0.

```
moyenne(T[0])
```

$$\frac{56}{15}$$

Le résultat est sous forme exacte. Pour avoir une approximation, on utilise `evalf`

```
evalf(moyenne(T[0]))
```

3.733333

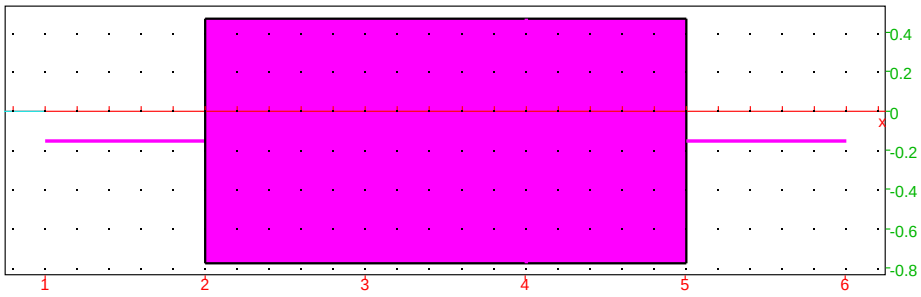
Si l'on ne veut que deux chiffres après la virgule, on peut par exemple utiliser `format` :

```
format(moyenne(T[0]),"f2")
```

3.73

moustache(L) : renvoie la « boîte à moustache » correspondant à la liste T :

```
moustache(T[0],couleur=magenta+rempli+epaisseur_ligne_3)
```



la partie `couleur=magenta+rempli+epaisseur_ligne_3` est bien sûr optionnelle mais permet de choisir l'aspect du graphique.

2. Observation du problème

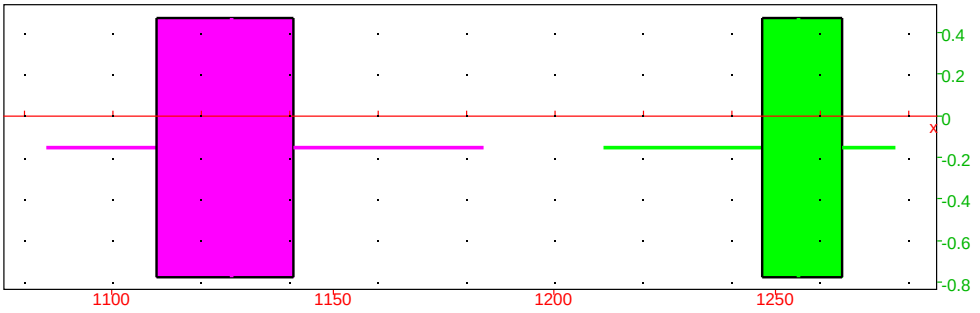
On va simuler n séries de 10 000 lancers de trois dés. On créera une liste `neuf` qui contiendra le nombre d'occurrences de 9 dans chacune des n séries et pareil pour 10. On en calculera les moyennes et les boîtes à moustache correspondant.

```
toscane(n):={
neuf:=NULL; //  une séquence vide au départ pour compter les 9
dix:=NULL; // idem pour 10
pour k de 1 jusque n faire // pour chacun des n simulations
T:=ranm(1,10000,'rand(6)+rand(6)+rand(6)+3') // on lance 10000 fois 3 dés et on
    fait la somme des numéros obtenus.
neuf:=neuf,count_eq(9,T); // on compte le nombre de 9 dans T et on le rajoute
    dans notre séquence
dix:=dix,count_eq(10,T); // idem pour 10
fpour;
N:=moyenne([neuf])/10000*100; // fréquence des sorties de 9
```

```
D:=moyenne([dix])/10000*100; // fréquence des sorties de 10
mousN:=moustache([neuf],couleur=magenta+rempli+epaisseur_ligne_3);
mousD:=moustache([dix],couleur=vert+rempli+epaisseur_ligne_3);
print("Sur "+10000*n+" essais, la fréquence de sortie de 9 est de "+format(N,"f2
    ")+"% et celle de 10 est de "+format(D,"f2")+"%");
mousN,mousD;
}:;
```

Par exemple, pour obtenir 100 simulations de 10 000 lancers on entre :

```
toscane(100)
```



"Sur 1000000 essais, la fréquence de sortie de 9 est de 11.26% et celle de 10 est de 12.53%"

3. À la recherche de l'origine du problème

On peut se demander si les six lancers correspondant à 9 sont équiprobables ; la question se pose évidemment en les mêmes termes pour 10. Pour se donner une idée, on va déterminer la fréquence de sortie de chacun de ces lancers en en simulant 10 000.
On utilise à bon escient la notion d'ensemble pour XCAS : il s'agit d'une collection d'éléments non ordonnée qu'on écrit entre %{ et %}.
On utilise également la fonction `apply(fonction,liste)` qui applique une fonction à chaque élément d'une liste.

```
neneuf():={
L:=[0$6];
neuf:=NULL:;
pour k de 1 jusque 10000 faire
a:=rand(6)+1:;b:=rand(6)+1:;c:=rand(6)+1:;
si a+b+c==9 alors neuf:=neuf,[a,b,c]; // on liste les tirages donnant 9
fsi:;
fpour:;
N:=[neuf]:;
n:=size(N)-1;
pour k de 0 jusque n faire
si %{op(N[k])%}==%{6,1,2%} alors L[0]:=L[0]+1;
sinon  si %{op(N[k])%}==%{5,2,2%} alors L[1]:=L[1]+1;
           sinon si %{op(N[k])%}==%{5,3,1%} alors L[2]:=L[2]+1;
                     sinon si %{op(N[k])%}==%{4,4,1%} alors L[3]:=L[3]+1;
                               sinon si %{op(N[k])%}==%{4,3,2%} alors L[4]:=L
                                  [4]+1;
                                         sinon si %{op(N[k])%}==%{3,3,3%} alors L
                                            [5]:=L[5]+1;
fsi;fsi;fsi;fsi;fsi;fsi;
fpour;
s:=sum(L);
```

```
T:=[["6,1,2","5,2,2","5,3,1" ,"4,4,1","4,3,2" ,"3,3,3"],apply(x->floor((x/s)*100)
    ,L)];
return(T);
}:;
```

Lançons 10 000 fois nos trois dés :

```
neneuf()
```

$$\begin{pmatrix} 6,1,2 & 5,2,2 & 5,3,1 & 4,4,1 & 4,3,2 & 3,3,3 \\ 23 & 11 & 24 & 12 & 23 & 3 \end{pmatrix}$$

On remarque que les tirages comportant un double sortent 2 fois moins que les tirages de faces toutes distinctes. La proportion est de 6 contre 1 pour le tirage "3,3,3".
On peut aisément adapter la procédure à 10 :

```
didix():={
L:=[0$6];
dix:=NULL:;
pour k de 1 jusque 10000 faire
a:=rand(6)+1:;b:=rand(6)+1:;c:=rand(6)+1:;
si a+b+c==10 alors dix:=dix,[a,b,c];
fsi:;
fpour:;
N:=[dix]:;
n:=size(N)-1;
pour k de 0 jusque n faire
si %{op(N[k])%}==%{6,3,1%} alors L[0]:=L[0]+1;
sinon  si %{op(N[k])%}==%{6,2,2%} alors L[1]:=L[1]+1;
           sinon si %{op(N[k])%}==%{5,4,1%} alors L[2]:=L[2]+1;
                     sinon si %{op(N[k])%}==%{5,3,2%} alors L[3]:=L[3]+1;
                               sinon si %{op(N[k])%}==%{4,4,2%} alors L[4]:=L
                                  [4]+1;
                                         sinon si %{op(N[k])%}==%{4,3,3%} alors L
                                            [5]:=L[5]+1;
fsi;fsi;fsi;fsi;fsi;fsi;
fpour;
s:=sum(L);
T:=[["6,3,1","6,2,2","5,4,1","5,3,2","4,4,2","4,3,3"],apply(x->floor((x/s)*100),L
    )];
return(T);
}:;
```

```
didix()
```

$$\begin{pmatrix} 6,3,1 & 6,2,2 & 5,4,1 & 5,3,2 & 4,4,2 & 4,3,3 \\ 21 & 10 & 23 & 22 & 11 & 10 \end{pmatrix}$$

Il ne reste plus qu'à prouver ce phénomène par des considérations probabilistes...

II - Sujet 007

On va obtenir directement a_n et b_n pour tout naturel n à l'aide du programme suivant qui calcule les listes $[a_n, b_n]$. Ainsi, $a_n = w(n)[0]$, le premier élément de la liste w(n) et $b_n = w(n)[1]$, le deuxième élément de la liste w(n) :

```
w(n):={
W:=[20,60];
si n=0 alors return(W);
sinon
 pour k de 1 jusque n faire
 W:=[(2*W[0]+W[1])/4,(W[0]+2*W[1])/4]
 // on calcule le nouveau W en fonction du précédent
fpour;
fsi;
}:;
```

Par exemple :

```
w(50)
```

$$[\frac{7178979876918525887702485}{3169126500570573503741758013 44},\frac{7178979876918525887702495}{3169126500570573503741758013 44}]$$

Pour en avoir une valeur approchée, on utilise `evalf`

```
evalf(w(50))
```

$$[0.000023, 0.000023]$$

Et si l'on veut tous les termes pair jusqu'à w_{50} :

```
[seq(evalf(w(2*j)),j=0..25)]
```

$$\begin{pmatrix} 20.000000 & 60.000000 \\ 21.250000 & 23.750000 \\ 12.578125 & 12.734375 \\ 7.114258 & 7.124023 \\ 4.004211 & 4.004822 \\ 2.252522 & 2.252560 \\ 1.267053 & 1.267055 \\ 0.712718 & 0.712718 \\ 0.400904 & 0.400904 \\ 0.225508 & 0.225508 \\ 0.126848 & 0.126848 \\ 0.071352 & 0.071352 \\ 0.040136 & 0.040136 \\ 0.022576 & 0.022576 \\ 0.012699 & 0.012699 \\ 0.007143 & 0.007143 \\ 0.004018 & 0.004018 \\ 0.002260 & 0.002260 \\ 0.001271 & 0.001271 \\ 0.000715 & 0.000715 \\ 0.000402 & 0.000402 \\ 0.000226 & 0.000226 \\ 0.000127 & 0.000127 \\ 0.000072 & 0.000072 \\ 0.000040 & 0.000040 \\ 0.000023 & 0.000023 \end{pmatrix}$$

On obtient encore plus facilement les termes des suites (u_n) et (v_n) :

```
u(n):=w(n)[0]+w(n)[1]:;
```

```
v(n):=w(n)[1]-w(n)[0]:;
```

Visualisons les premiers termes grâce à la commande `seq` :

```
seq(v(j),j=0..10)
```

$$40, 10, \frac{5}{2}, \frac{5}{8}, \frac{5}{32}, \frac{5}{128}, \frac{5}{512}, \frac{5}{2048}, \frac{5}{8192}, \frac{5}{32768}, \frac{5}{131072}$$

```
seq(u(j),j=0..10)
```

$$80, 60, 45, \frac{135}{4}, \frac{405}{16}, \frac{1215}{64}, \frac{3645}{256}, \frac{10935}{1024}, \frac{32805}{4096}, \frac{98415}{16384}, \frac{295245}{65536}$$

Pour enfoncer le clou, étudions les rapports entre termes consécutifs :

```
seq(v(j+1)/v(j),j=0..10)
```

$$\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}$$

```
seq(u(j+1)/u(j),j=0..10)
```

$$\frac{3}{4}, \frac{3}{4}, \frac{3}{4}, \frac{3}{4}, \frac{3}{4}, \frac{3}{4}, \frac{3}{4}, \frac{3}{4}, \frac{3}{4}, \frac{3}{4}, \frac{3}{4}$$

III - Sujet 030

Encore une suite définie par une relation de récurrence ! Nous commençons à savoir comment faire ! Petite variante ici : le premier terme varie et nous voulons une représentation graphique.

```
u(n):={
U:=uo;
si n==1 alors U;
sinon
  pour k de 2 jusque n faire
  U:=U/(k-1)+1.0;
  fpour;
fsi;
}:;
```

Par exemple, fixons uo à -10 et observons les 15 premiers termes :

```
uo:=-10:;
```

```
seq(u(k),k=1..15)
```

$$-10, -9.000000, -3.500000, -0.166667, 0.958333, 1.191667, 1.198611, 1.171230, 1.146404, 1.127378, 1.112738, 1.10$$

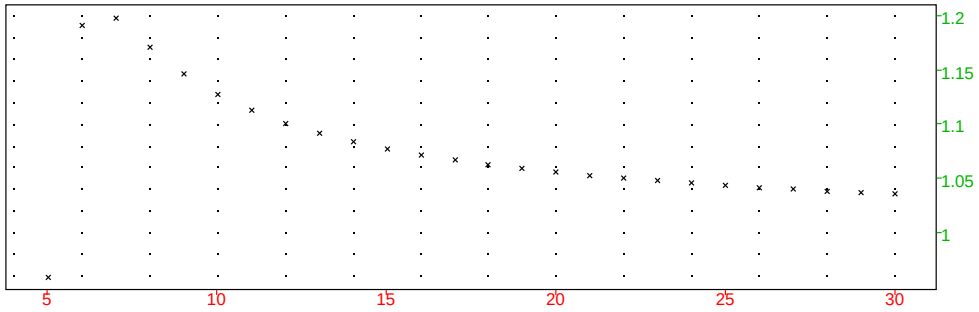
ou, de manière plus lisible :

```
 [[seq("u("+k+")",k=1..15)],[seq(format(u(k),"f2"),k=1..15)]]
```

u(1)	u(2)	u(3)	u(4)	u(5)	u(6)	u(7)	u(8)	u(9)	u(10)	u(11)	u(12)	u(13)	u(14)	u(15)
-10.00	-9.00	-3.50	-0.17	0.96	1.19	1.20	1.17	1.15	1.13	1.11	1.10	1.09	1.08	1.08

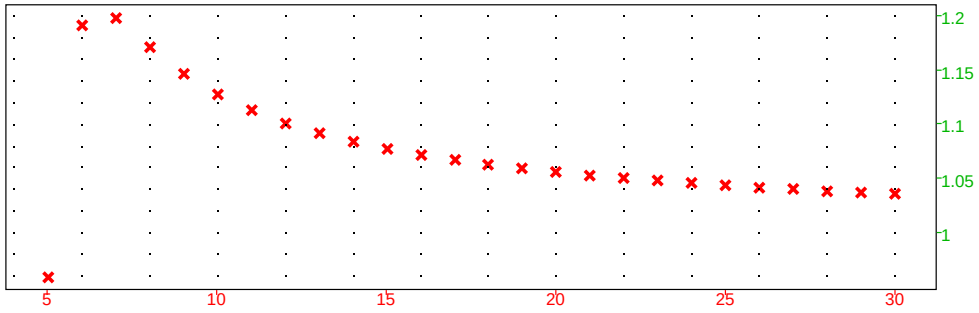
Ensuite, regardons ce que cela donne graphiquement :

```
seq(point(k,u(k)),k=5..30)
```



ou, de manière plus visible :

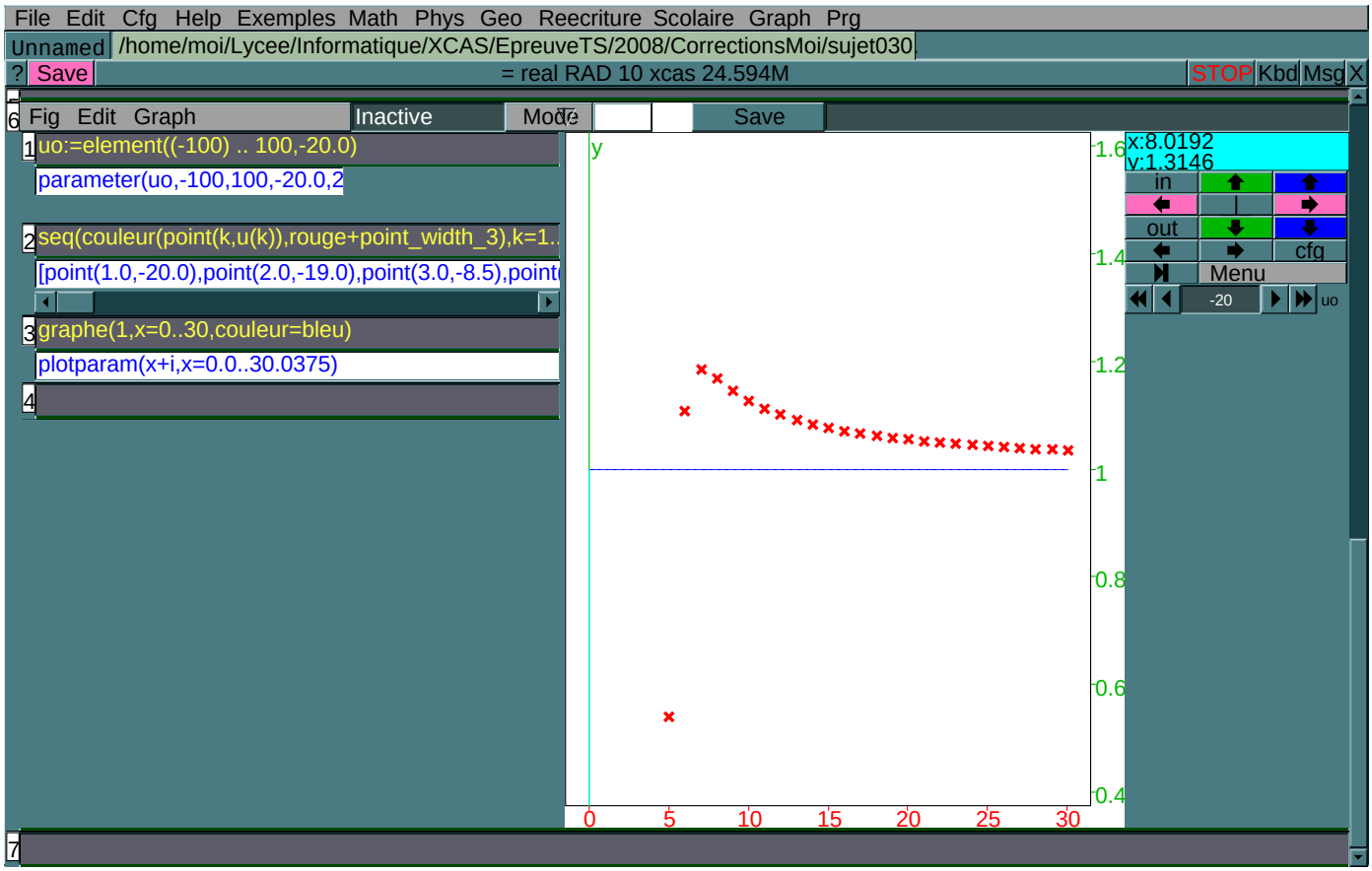
```
seq(couleur(point(k,u(k)),rouge+point_width_3),k=5..30)
```



Est-ce que le premier terme joue un rôle important ? Créons un curseur le faisant varier en ouvrant une fenêtre de géométrie (Alt + G) :

```
uo:=element((-100) .. 100) // uo varie entre -100 et 100
seq(couleur(point(k,u(k)),rouge+point_width_3),k=1..30)
graphe(1,x=0..30,couleur=bleu) // pour visualiser la limite
```

Il ne reste plus qu'à faire varier le curseur :



IV - Sujet 044

Nous allons ré-utiliser n donc vidons les mémoires d'affectation :

```
restart(NULL)
```

$$[D, L, N, T, U, W, a, b, c, didix, dix, mousD, mousN, n, neneuf, neuf, s, P, u, uo, v, w]$$

Le problème ici, c'est que XCAS peut donner directement la réponse !

```
factoriser((6/n)*somme(k^2,k=1..n))
```

$$(n+1)(2n+1)$$

... donc faisons comme si nous ne le savions pas...
Nous allons donc construire une somme avec une boucle *pour* :

```
u(n):={
local s,k;
s:=0;
pour k de 1 jusque n faire
  s:=s+k^2; // on rajoute k^2 à chaque fois
fpour;
return(6*s/n) // on n'oublie pas de multiplier par 6/n
}:;
```

Nous obtenons alors :

```
seq(u(j),j=1..10)
```

$$6,15,28,45,66,91,120,153,190,231$$

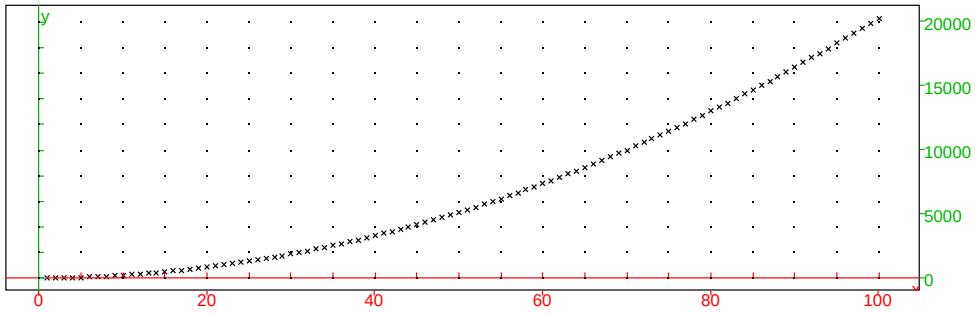
ou, sous forme de tableau :

```
[[seq("u("+k+")",k=1..10)],[seq(u(k),k=1..10)]]
```

$$\begin{pmatrix} u(1) & u(2) & u(3) & u(4) & u(5) & u(6) & u(7) & u(8) & u(9) & u(10) \\ 6 & 15 & 28 & 45 & 66 & 91 & 120 & 153 & 190 & 231 \end{pmatrix}$$

Si ces nombres ne nous parlent pas, observons graphiquement :

```
seq(point(j,u(j)),j=1..100)
```



Puisqu'il s'agit d'un sujet de Bac, il ne faut pas chercher trop loin :-)
Ça ressemble à un segment de parabole donc on va chercher $f(n)$ sous la forme d'une expression du type an^2+bn+c. On peut calculer facilement u_1, u_2 et u_3 donc on peut par exemple résoudre le système :

$$\begin{cases} a+b+c=u_1 \\ 4a+2b+c=u_2 \\ 9a+3b+c=u_3 \end{cases}$$

ce qui donne avec XCAS :

```
f(n):=a*n^2+b*n+c // on définit f
```

$$(n)\rightarrow a*n\hat{}2+b*n+c$$

```
linsolve([f(1)=u(1),f(2)=u(2),f(3)=u(3)],[a,b,c]) // on résout le système
```

$$[2,3,1]$$

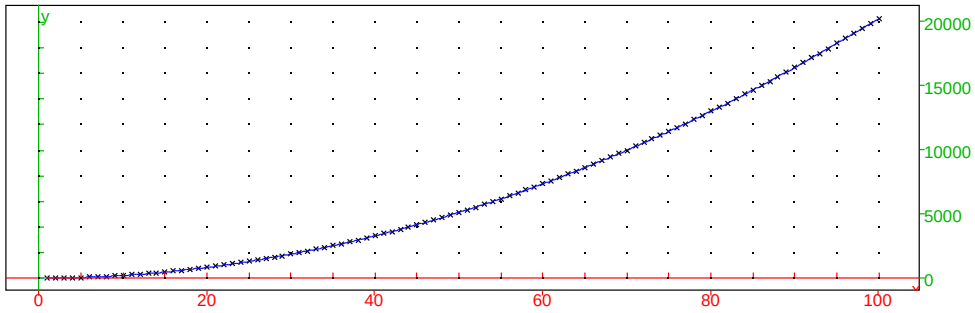
On obtient donc que $f(n)=2n^2+3n+1$:

```
[a,b,c]:=linsolve([f(1)=u(1),f(2)=u(2),f(3)=u(3)],[a,b,c]):; f(n)
```

$$\text{Done}, 2n^2+3n+1$$

Observons graphiquement :

```
seq(point(j,u(j)),j=1..100),graphe(f(n),n=1..100,couleur=bleu)
```



Cela semble correspondre graphiquement. Du point de vue numérique :

```
seq(f(k)-u(k),k=1..30)
```

$$0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0$$

V - Sujet 063

Tiens, encore une suite ! Bon, cette fois-ci, c'est en arithmétique mais c'est malgré tout plus simple informatiquement puisque nous avons la formule explicite donnant u_n en fonction de n.

```
u(n):=12*n+5:;
```

```
seq(u(k) mod 20, k=0..19)
```

$$5\%20,-3\%20,9\%20,1\%20,-7\%20,5\%20,-3\%20,9\%20,1\%20,-7\%20,5\%20,-3\%20,9\%20,1\%20,-7\%20,5\%20,-$$

Le % 20 indique que nous travaillons modulo 20. XCAS donne par défaut le reste symétrique. Si nous voulons avoir des entiers, il faut donc travailler « modulo zéro » :

```
seq((u(k) mod 20)%0, k=0..19)
```

$$5,-3,9,1,-7,5,-3,9,1,-7,5,-3,9,1,-7,5,-3,9,1,-7$$

On nous demande ensuite de faire varier a, b et p. Nous allons donc construire une fonction dépendant de ces trois variables :

```
f(a,b,p):=seq((a*k+b mod p)%0, k=0..19):;
```

Observons les cas particuliers demandés :

```
f(5,-3,20)
```

$$-3,2,7,-8,-3,2,7,-8,-3,2,7,-8,-3,2,7,-8,-3,2,7,-8$$

```
f(5,-3,7)
```

$$-3,2,0,-2,3,1,-1,-3,2,0,-2,3,1,-1,-3,2,0,-2,3,1$$

On peut multiplier les exemples.
On peut observer les influences de chaque paramètre. Construisons d'abord une procédure donnant la période :

```
periode(a,b,p):={
K:=1;
tantque (a*K+b) mod p != b mod p faire K:=K+1;
ftantque;
K;
}:;
```

puis observons avec plusieurs valeurs de p :

```
seq(periode(5,-3,p),p=1..20)
```

$$1,2,3,4,1,6,7,8,9,2,11,12,13,14,3,16,17,18,19,4$$

Est-ce que b semble influer ?

```
[seq([seq(periode(5,b,p),p=1..20)],b=-3..3)]
```

$$\begin{pmatrix}
1 & 2 & 3 & 4 & 1 & 6 & 7 & 8 & 9 & 2 & 11 & 12 & 13 & 14 & 3 & 16 & 17 & 18 & 19 & 4 \\
1 & 2 & 3 & 4 & 1 & 6 & 7 & 8 & 9 & 2 & 11 & 12 & 13 & 14 & 3 & 16 & 17 & 18 & 19 & 4 \\
1 & 2 & 3 & 4 & 1 & 6 & 7 & 8 & 9 & 2 & 11 & 12 & 13 & 14 & 3 & 16 & 17 & 18 & 19 & 4 \\
1 & 2 & 3 & 4 & 1 & 6 & 7 & 8 & 9 & 2 & 11 & 12 & 13 & 14 & 3 & 16 & 17 & 18 & 19 & 4 \\
1 & 2 & 3 & 4 & 1 & 6 & 7 & 8 & 9 & 2 & 11 & 12 & 13 & 14 & 3 & 16 & 17 & 18 & 19 & 4 \\
1 & 2 & 3 & 4 & 1 & 6 & 7 & 8 & 9 & 2 & 11 & 12 & 13 & 14 & 3 & 16 & 17 & 18 & 19 & 4 \\
1 & 2 & 3 & 4 & 1 & 6 & 7 & 8 & 9 & 2 & 11 & 12 & 13 & 14 & 3 & 16 & 17 & 18 & 19 & 4
\end{pmatrix}$$

et a ?

```
[seq([seq(periode(a,-3,p),p=1..20)],a=1..10)]
```

$$\begin{pmatrix}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 \\
1 & 1 & 3 & 2 & 5 & 3 & 7 & 4 & 9 & 5 & 11 & 6 & 13 & 7 & 15 & 8 & 17 & 9 & 19 & 10 \\
1 & 2 & 1 & 4 & 5 & 2 & 7 & 8 & 3 & 10 & 11 & 4 & 13 & 14 & 5 & 16 & 17 & 6 & 19 & 20 \\
1 & 1 & 3 & 1 & 5 & 3 & 7 & 2 & 9 & 5 & 11 & 3 & 13 & 7 & 15 & 4 & 17 & 9 & 19 & 5 \\
1 & 2 & 3 & 4 & 1 & 6 & 7 & 8 & 9 & 2 & 11 & 12 & 13 & 14 & 3 & 16 & 17 & 18 & 19 & 4 \\
1 & 1 & 1 & 2 & 5 & 1 & 7 & 4 & 3 & 5 & 11 & 2 & 13 & 7 & 5 & 8 & 17 & 3 & 19 & 10 \\
1 & 2 & 3 & 4 & 5 & 6 & 1 & 8 & 9 & 10 & 11 & 12 & 13 & 2 & 15 & 16 & 17 & 18 & 19 & 20 \\
1 & 1 & 3 & 1 & 5 & 3 & 7 & 1 & 9 & 5 & 11 & 3 & 13 & 7 & 15 & 2 & 17 & 9 & 19 & 5 \\
1 & 2 & 1 & 4 & 5 & 2 & 7 & 8 & 1 & 10 & 11 & 4 & 13 & 14 & 5 & 16 & 17 & 2 & 19 & 20 \\
1 & 1 & 3 & 2 & 1 & 3 & 7 & 4 & 9 & 1 & 11 & 6 & 13 & 7 & 3 & 8 & 17 & 9 & 19 & 2
\end{pmatrix}$$

Ça change plus... Il semble y avoir des liens entre la période et le PGCD de a et p... mais ceci est une autre histoire.

VI - Sujet 066

Une suite ! Bon, elle est aléatoire cette fois nous dit le sujet...
Construisons-la :

```
s(n):={
local S,k;
S:=0;
pour k de 1 jusque n faire
  si rand(2)==0 alors S:=S+1;
  sinon S:=S-1;
  fsi;
fpour;
}:;
```

Observons :

```
seq(s(k),k=1..10)
```

$$1,2,-1,-4,-1,4,-1,-2,5,2$$

simulons à plus grande échelle :

```
[seq([seq(s(k),k=1..10)],j=1..20)]
```

$$\begin{pmatrix}
-1 & 2 & -1 & -2 & -1 & 0 & -5 & -4 & 3 & 0 \\
-1 & -2 & -3 & -2 & 1 & -2 & -5 & 4 & 1 & -4 \\
1 & 0 & 3 & 0 & -1 & -2 & 5 & 2 & 3 & -4 \\
1 & 0 & -1 & 0 & 3 & 4 & -1 & -2 & 1 & 0 \\
1 & 2 & 1 & -2 & 1 & 0 & 3 & 2 & -1 & -4 \\
-1 & 0 & -1 & 2 & -1 & 2 & 3 & -6 & -1 & 2 \\
1 & 0 & -1 & 0 & -3 & 0 & 1 & 4 & 5 & 0 \\
-1 & 2 & -1 & -2 & -1 & -2 & -1 & 0 & -1 & 2 \\
-1 & -2 & 3 & -4 & -1 & 0 & -1 & 4 & -3 & -2 \\
-1 & 2 & -1 & 4 & -1 & 0 & 1 & 0 & 3 & 2 \\
1 & 0 & 1 & 2 & 1 & -2 & -1 & -4 & 3 & 2 \\
1 & 0 & 1 & 0 & -1 & -2 & 1 & 2 & -1 & 0 \\
-1 & -2 & 1 & 4 & -3 & 0 & -1 & -2 & 5 & -2 \\
-1 & 0 & -1 & -2 & -3 & 2 & -1 & -2 & -3 & -4 \\
-1 & -2 & -3 & 0 & 1 & -4 & -7 & 2 & -3 & -4 \\
-1 & 0 & -1 & 0 & -3 & -2 & 1 & 0 & -3 & -2 \\
-1 & 0 & -1 & 2 & 3 & 4 & 1 & -4 & 3 & 0 \\
-1 & 2 & -1 & 0 & 1 & 4 & -1 & 6 & -5 & 4 \\
1 & 0 & 1 & 0 & -1 & 0 & -1 & 0 & -1 & 0 \\
-1 & 0 & -1 & -2 & -3 & 4 & -1 & -4 & -1 & 2
\end{pmatrix}$$

À chaque fois que nous tapons sur Enter, nous obtenons 20 nouvelles simulations : ça fait gagner un peu de place...
Voyons un peu plus loin et effectuons 10 000 simulations :

```
A:=[seq([seq(s(k),k=1..10)],j=1..10000)]:;
```

cela peut prendre jusqu'à 15 secondes avec un ordinateur peu performant.
Occupons-nous du calcul des fréquences. Le problème revient à comptabiliser, pour une colonne donnée, la fréquence d'apparition de 0. Notre tableau a 10 colonnes et 10 000 lignes. Pour extraire la j^e colonne on entre `col(A,j-1)` car XCAS commence à compter à O ! On n'oublie pas d'utiliser la commande `count_eq` rencontrée au paragraphe I - page 1.

```
seq(count_eq(0,col(A,j))/100,j=0..9)
```

$$0,\frac{4987}{100},0,\frac{3753}{100},0,\frac{771}{25},0,\frac{2663}{100},0,\frac{602}{25}$$

Ou, plus joliment :

```
[[seq("A"+j,j=0..9)],[seq(evalf(count_eq(0,col(A,j))/100)+"%",j=0..9)]]
```

A0	A1	A2	A3	A4	A5	A6	A7	A8	A9
0.0%	49.87%	0.0%	37.53%	0.0%	30.84%	0.0%	26.63%	0.0%	24.08%