Le plus petit chiffre

On utilise ici des sous-programmes : ils rendent accessibles des situations relativement complexes par le découpage des tâches qu'ils permettent. La même situation devient beaucoup trop lourde à mettre en place avec un logiciel n'autorisant pas les sous-routines (comme c'est le cas pour la version actuelle d'Algobox).

Dans les programmes

Réalisation d'une simulation, probabilité sur un ensemble fini, calculs sur des nombres entiers, boucle, instruction conditionnelle. Découverte d'une croissance exponentielle ("explosion" du temps de calcul).

1 Plus petit chiffre

Écrire la partie traitement de l'algorithme suivant :

Entrée	un entier naturel n
Traitement	
Sortie	le plus petit chiffre de l'écriture décimale de n

Le traduire en machine.

2 Simulation

Soit $j \geqslant 2$ un entier. On tire au hasard un entier s'écrivant avec j chiffres (écriture décimale usuelle). Si le plus petit chiffre est 0, je gagne et si le plus petit chiffre est 2, vous gagnez.

Écrire un programme qui simule n parties et affiche les fréquences de sortie de chacun des chiffres de 0 à 9.

Acceptez vous de jouer à ce jeu ?

3 Dénombrement

Écrire un programme qui affiche les probabilités de sortie de chacun des chiffres de 0 à 9 (qui dénombrera donc le nombre d'apparitions de 0 comme plus petit chiffre d'un entier à j chiffres, le nombre d'apparitions de 1 comme plus petit chiffre d'un entier à j chiffres ...)

4 Mystère

On donne le programme ci-dessous :

Entrée	un entier $j \geqslant 2$

Traitement	initialisation de liste à [0,0,0,0,0,0,0,0,0,0]
	b prend la valeur $10^j - 10^{j-1}$
	Pour k de 0 jusque 9 faire
	  liste[k] prend la valeur $b-(10-(k+1))^j$
	  b prend la valeur $b-$liste[k]
	Fin_ Pour

Sortie	Affichage de la liste

Faire le lien avec les questions précédentes. Expliquez le fonctionnement.

5 Somme des plus petits chiffres des entiers à j chiffres

Où l'on contaste qu'un peu de mathématiques économise des décennies de calcul.

1. Écrire la partie traitement puis traduire sur machine :

Entrée	un entier naturel $j \geqslant 2$
Traitement	
Sortie	la somme des plus petits chiffres de tous les entiers à j chiffres

   On se servira d'une boucle parcourant l'ensemble des nombres à j chiffres.
2. Évaluer expérimentalement le temps de calcul pour l'exécution avec l'entrée $j = 4$. En déduire une évaluation du temps de calcul nécessaire à l'exécution avec l'entrée $j = 16$.
3. Proposer une formule permettant le calcul de cette somme beaucoup plus rapidement.

Éléments de réponses – Ti nspire

1 Plus petit chiffre

```
Define pptchiffre(n)=
Func
Local ppc,unite
If n = 0 Then
Return 0
EndIf
ppc := 9
While n ≠ 0
unite := mod(n,10)
If unite < ppc Then
ppc := unite
EndIf
n := (n−unite)/10
EndWhile
Return ppc
EndFunc
```

Une version récursive :

```
pptchif(n):=when(n<10,n,min({mod(n,10),pptchif(intDiv(n,10))}))
```

Dans la version suivante, on construit d'abord la liste des chiffres de l'écriture décimale de l'entier $n \neq 0$ puis on utilise la fonction min du logiciel qui renvoie le plus petit élément d'une liste :

```
Define pptchiffre(n)=
Func
Local liste
liste :={ }
While n ≠ 0
liste:=augment({mod(n,10)},liste)
n:=intDiv(n,10)
EndWhile
Return min(liste)
EndFunc
```

2 Simulations

```
Define simulation(j, n)=
Func
Local k,liste,tirage,ppc
liste := { 0,0,0,0,0,0,0,0,0,0 }
For k,1,n
tirage :=randInt(10^(j-1),10^j - 1)
ppc :=pptchiffre(tirage)
liste[ppc+1] :=liste[ppc+1]+1
EndFor
liste :=approx(liste/n)
Return liste
EndFunc
```

3 Dénombrement exact par boucles

Define repartition(j)=	
Func	
Local liste,p,i	
liste:= {0,0,0,0,0,0,0,0,0,0}	liste contiendra les effectifs
For i,10^{j-1},$10^j - 1$	boucle testant tous les entiers à j chiffres
p :=pptchiffre(i) + 1	p devient le plus petit chiffre de l'entier i, le décalage de 1 sert à remplir la liste dont les éléments sont numérotés de 1 à 10 (et non de 0 à 9)
liste[p] :=liste[p] + 1	liste[1], premier élément de la liste, compte les 0, liste[2] compte les 1 ...
EndFor	fin de la boucle
Return liste	
EndFunc	fin de la définition de la fonction

Quelques résultats :

1. Avec $j = 3$:
   liste :[171,217,169,127,91,61,37,19,7,1]
   liste :[0.19,0.241,0.188,0.141,0.101,0.068,0.041,0.021,0.008,0.001]
2. Avec $j = 4$:
   liste :[2439,2465,1695,1105,671,369,175,65,15,1]
   liste :[0.271,0.274,0.188,0.123,0.075,0.041,0.019,0.007,0.002,0.0]

4 Programme mystère

```
Define repart(j)=
Func
Local liste,b,i
liste:= {0,0,0,0,0,0,0,0,0,0}
b := 10^j − 10^(j−1)
For i,1,10
liste[i] :=b − (10 − i)^j
b := b − liste[i]
EndFor
Return liste
EndFunc
```

Le programme mystère a le même rôle que le programme précédent de décompte exact.
Pour les entiers à 3 chiffres, on peut dénombrer de la façon suivante (et c'est le principe du programme mystère) :

1. Il existe $10^3 - 10^2 = 9 \times 10 \times 10 = 900$ entiers à trois chiffres.
2. Il en existe $9 \times 9 \times 9$ qui ne contiennent que des chiffres $\geqslant 1$. Le nombre d'entiers à trois chiffres dont le plus petit chiffre est 0 est donc de $900 - 9^3 = 171$.
3. Il existe 8^3 entiers à trois chiffres de plus petit chiffre $\geqslant 2$. Il y a donc $900 - 171 - 8^3 = 217$ entiers de trois chiffres de ppc 1.
4. Il existe 7^3 entiers à trois chiffres de plus petit chiffre $\geqslant 3$. Il y a donc $900 - 171 - 217 - 7^3 = 169$ entiers de trois chiffres de ppc 2.
5. etc

Pour les entiers entre 10^{j-1} et $10^j - 1$, c'est à dire les entiers à j chiffres :

Entiers à j chiffres	Effectif
à j chiffres	$10^j - 10^{j-1} = 10^{j-1} \times 9$
tous les chiffres $\geqslant 1$	9^j
tous les chiffres $\geqslant 2$	8^j
tous les chiffres $\geqslant 3$	7^j
tous les chiffres $\geqslant 4$	6^j
tous les chiffres $\geqslant 5$	5^j
tous les chiffres $\geqslant 6$	4^j
tous les chiffres $\geqslant 7$	3^j
tous les chiffres $\geqslant 8$	2^j
tous les chiffres $\geqslant 9$	1

Plus petit chiffre	Effectif
ppc=0	$9\times10^{j-1}-9^j$
ppc=1	$9\times10^{j-1}-\left(9\times10^{j-1}-9^j\right)-8^j=9^j-8^j$
ppc=2	$9\times10^{j-1}-\left(9\times10^{j-1}-9^j\right)-\left(9^j-8^j\right)-7^j=8^j-7^j$
ppc=3	7^j-6^j
ppc=4	6^j-5^j
ppc=5	5^j-4^j
ppc=6	4^j-3^j
ppc=7	3^j-2^j
ppc=8	2^j-1^j
ppc=9	1

5 Somme des plus petits chiffres des entiers à j chiffres

1. Exemple de programme déterminant la somme :

```
Define sommedesppc(j)
Func
Local som,k
som :=0
For k,10^(j-1),10^j-1
som :=som+pptchiffre(k)
EndFor
Return som
EndFunc
```

   Ou en utilisant la fonction somme du langage :

$$s(j):=\sum_{k=10^{j-1}}^{10^j-1}\text{pluspetitchiffre}(k)$$

2. On part sur un temps de une seconde pour l'exécution avec l'entrée $j=4$. Ce temps correspond à peu près à l'exécution de la boucle qui comprend à peu près $10^4-10^3=9\times10^3$ tours.
   Dans l'exécution avec l'entrée $j=16$, la boucle comprend à peu près 9×10^{15} tours, soit 10^{12} fois plus. En considérant que le temps est multiplié à peu près dans les mêmes proportions, on trouve plus de 31 700 ans.
   Avec $j=20$, on multiplie par $\approx10^{16}$ et on obtient un temps de plus de 300 millions d'années.
3. D'après le tableau d'une question précédente, la somme :

$$S=\sum_{n\text{ entier à }j\text{ chiffres}}\text{pluspetitchiffre}(n)$$

   est donnée par :

$$\begin{aligned}S&=0\times\left(9\times10^{j-1}-9^j\right)+1\times\left(9^j-8^j\right)+2\times\left(8^j-7^j\right)+3\times\left(7^j-6^j\right)+\cdots+8\times\left(2^j-1\right)+9\\&=9^j+8^j+\cdots+2^j+1\end{aligned}$$

d'où un programme simple de calcul :

$$t(j) := \sum_{k=1}^{9} k^j$$