l'algorithme de Kaprekar

par David et Jérôme (Tle C)

Lycée Bartholdi, 68000 Colmar

KAPREKAR,
vous avez dit
KAPREKAR ?
Mais qui est-ce donc,
KAPREKAR ??
… mystère !!!

Mathématicien hongrois
selon certains,
indien
selon d'autres,
KAPREKAR
a pourtant laissé son nom
à un des plus célèbres
et stupéfiants
algorithmes.

Se composant en tout
et pour tout
d'une opération de soustraction,
cet algorithme a longtemps intrigué et
intrigue toujours encore
nombre de
mathématiciens et
informaticiens.

Jugez-en vous-même.

1. — Prendre un nombre X de quatre chiffres, c'est-à-dire compris entre 0 et 9999 (NB : 74 peut s'écrire 0074).

2. — arranger par ordre décroissant les chiffres de X et le coder sous la forme *abcd* tel que $a \geq b \geq c \geq d$.

3. — arranger maintenant les chiffres de X par ordre croissant. On obtient bien entendu *dcba* tel que $d \leq c \leq b \leq a$.

4. — Calculer la différence Y de *abcd* - *dcba* (Y est un nombre de quatre chiffres).

5. — Refaire toutes les opérations à partir de 2. en remplaçant X par Y.

Tel est, en quelques mots, le principe on ne peut plus simple de l'algorithme de Kaprekar. Mais qu'y a-t-il donc de si surprenant dans tout cela, nous direz-vous ? Eh bien, quel que soit le nombre X choisi, Y finit toujours par tomber sur le nombre 6 174 et y rester. En effet, si l'on applique l'algorithme ci-dessus :

$$\begin{array}{r} 7\,641 \\ -\,1\,467 \\ \hline = 6\,174 \end{array}$$

De plus, ce résultat apparait au bout de huit tours maximum.

Remarque : Tous les nombres de quatre chiffres ne donnent pas 6 174 par l'algorithme de KAPREKAR. Ce sont tous les nombres de la forme aaaa (1111, 2222, …) et qui donnent un résultat nul. Notre étude se limite donc à 9990 nombres.

Quelques programmes.

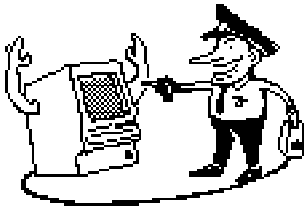
Malgré sa simplicité d'exécution, l'algorithme de Kaprekar devient vite lassant. Nous avons donc mis au point quelques programmes qui, décomposant, ordonnant puis soustrayant les nombres à notre place, nous rendaient le travail plus facile !

Les programmes Basic 1.1 et Casio (fx) utilisent le fait qu'il existe 24 façons différentes d'ordonner les quatre lettres A, B, C, D codant ici le nombre choisi.

Programme en Pascal

```
PROGRAM KAPREKAR;
USES
CRT;

VAR
J,CI,AI,BI,I,K : INTEGER;
C,A,B : STRING;
D,E : CHAR;
TEST : BOOLEAN;

PROCEDURE CLASSE;
BEGIN
REPEAT
  BEGIN
  FOR I:=1 TO 3 DO
    BEGIN
    IF ORD(A[I]) < ORD(A[I+1])
    THEN
      BEGIN
      D:=A[I];
      A[I]:=A[I+1]
      A[I+1]:=D;
      END;
    END;
  END;
UNTIL (ORD(A[I])>= ORD(A[2])) AND (ORD(A[2
])>= ORD(A[3])) AND (ORD(A[4])>= ORD(A[3]) );
END;

PROCEDURE INVERSE;
BEGIN
B:='';
FOR I:= 4 DOWNTO 1 DO
  BEGIN
  B:=B+A[1];
  END;
END;

BEGIN
CLRSCR;
WRITELN ('DONNEZ UN NOMBRE DE 4 CHIFFRES,N E
S''ECRIVANT PAS SOUS ');
WRITELN ('LA FORME "aaaa"');
TEST:=TRUE;
READLN (A);
J:=0;

REPEAT
BEGIN
  CLASSE;
  INVERSE;

  VAL(A,AI,K);
  VAL(B,BI,K);
  CI:=AI-BI;
  WRITELN (AI,' - ',BI,' = ',CI);
  STR(CI,A);
    IF (A[1]=A[2]) AND (A[2]=A[3]) AND (LE
NGTH (A) = 3) THEN TEST:=FALSE;
    IF (A[1]=A[2]) AND (A[2]=A[3]) AND (A[
3]=A[4]) THEN TEST:=FALSE;
  IF A = '6174' THEN TEST:=FALSE;
  J:=J+1
END;
UNTIL TEST:=FALSE
IF A = '6174' THEN
WRITELN ('NOMBRE D''ITERATIONS NECESSAIRES :
',J);
END.
```

test du programme : ne tourne pas … à cause de USES CRT ?



(pour les programmes Basic 1.1 et Casio : commander les photocopies aux élèves du lycée Bartholdi … il n'y a pas assez de place ici pour les transcrire.)

Au bout d'un moment, nous avons remarqué que durant le déroulement des programmes, certains nombres apparaissaient plus souvent que d'autres. C'est le cas par exemple de 8532, 7641 ou encore 6642. Nous nous sommes alors intéressés aux propriétés du nombre Y en général, c'est-à-dire à la différence *abcd - dcba*.

Propriétés du nombre Y.

Soit X un nombre à quatre chiffres abcd et Y la différence *abcd - dcba*, avec les conditions décrites plus haut. Effectuons alors :

$$\begin{array}{r} a\,b\,c\,d \\ -\,d\,c\,b\,a \\ \hline = A\,B\,C\,D \end{array}$$

☛ « d moins a » : ça ne va pas car a > d. Calculons donc (d + 10) - a = D et retenons 1.

☛ « c moins (b + 1) » : b > c ainsi b + 1 > c. Calculons alors c + 10 - (b + 1) = C, c'est-à-dire c - b + 9 = C et retenons 1.

☛ « b moins (c + 1) » : deux cas se présentent.

CAS 1 : b < (c + 1). Cela signifie que b = c car b > c, et (b + 10) - (c + 1) = B.

CAS 2 : b > (c + 1). On a b - c + 1 = B.

☛ « a moins d » : deux cas à nouveau.

CAS 1 : b < (c + 1). (b + 10) - (c + 1) = B, donc a - (d + 1) = A à cause de la retenue.

CAS 2 : b > (c + 1). Donc a - d = A tout simplement.

Etudions maintenant les systèmes obtenus :

$$\text{si } b > (c+1) : \begin{cases} d + 10 - a = D \\ a - d = A \\ c - b + 9 = C \\ b - c - 1 = B \end{cases} \Leftrightarrow \begin{cases} A + D = 10 \\ B + C = 8 \end{cases}$$

$$\text{si } b < (c+1) : \begin{cases} d + 10 - a = D \\ a - (d+1) = A \\ b + 10 - (c+1) = B \\ c - b + 9 = C \end{cases} \Leftrightarrow \begin{cases} A + D = 9 \\ B = C = 9 \\ (\text{car } b = c) \end{cases}$$

Conclusion :

Pour $b \neq c$, A + D = 10 et B + C = 8 ; pour b = c, A + D = 9 et B = C = 9.

Programme.

Nous venons donc de prouver qu'au bout d'un tour d'algorithme, le nombre choisi prend une forme ABCD telle que

A + D = 10 et B + C = 8,

ou

A + D = 9 et B = C = 9.

Grace à cette information, nous allons pouvoir prouver que tous les nombres retombent à la fin sur 6 174. Mais avant tout, faisons un programme nous permettant de retrouver les nombres répondant aux propriétés de Y obtenues.

Programme en Basic 1.0

```
10 FOR N=1000 TO 9999
20 A=FIX(N/1000)
30 B=FIX(N/100)-10*A
40 C=FIX(N/10)-100*A-10*B
50 D=N-1000*A-100*B-10*C
100 IF A+D=10 AND B+C=8 THEN PRINT N;
110 IF B=9 AND C=9 AND A+D=9 THEN PRINT N;
120 NEXT N
130 END
```

Grace au programme, nous obtenons une série de 90 nombres :

1089	1179	1269	1359	1449	1539
1629	1719	1809	1998	2088	2178
2268	2358	2448	2538	2628	2718
2808	2997	3087	3177	3267	3357
3447	3537	3627	3717	3807	3996
4086	4176	4266	4356	4446	4536
4626	4716	4806	4995	5085	5175
5265	5355	5445	5535	5626	5715
5805	5994	6084	6174	6264	6354
6444	6534	6624	6714	6804	6993
7083	7173	7263	7353	7443	7533
7623	7713	7803	7992	8082	8172
8262	8352	8442	8532	8622	8712
8802	8991	9081	9171	9261	9351
9441	9531	9621	9711	9801	9990

Cependant, il n'est pas nécessaire d'étudier tous les nombres, car deux ou trois nombres comportant les mêmes chiffres donnent le même résultat par l'algorithme (ex : 1 359 et 9 531). Ainsi, après élimination, il ne subsiste que 30 nombres :

9801	8820	7731	6642	5553
9711	8721	7632	6543	5544
9621	8622	7533	6444	
9531	8532	7443	6552	
9441	8442	7641		
9990	8730	7551		
9981	8640			
9972	8550			
9963				
9954				

L'arbre de Kaprekar.

Il ne nous reste maintenant plus qu'à effectuer l'algorithme pour chacun de ces nombres. On obtient alors un arbre où l'on remarque que chaque nombre vient "s'emboîter" dans un autre, pour donner à la fin, bien entendu, 6 174.

Conclusion.

Grace à l'arbre ci-dessous, nous avons la preuve que tous les nombres finissent par converger vers 6 174, et en huit tours au maximum — car les 30 nombres trouvés plus haut, d'après la propriété de Y, apparaissent dès le premier tour de l'algorithme. **Application pratique ?** Cette propriété de "convergence", d'après les spécialistes de la numération (informaticiens et mathématiciens) pourrait permettre l'élaboration de nouveaux langages informatiques, ainsi que de nouvelles méthodes de calcul, basées sur la proportionalité de nombres passant par les différentes branches de l'arbre, mais ça … c'est une autre histoire !