# HP Forum Archive 19

## Challenge: Keith numbers

*Message #1 Posted by* ***Don Shepherd*** *on 5 Sept 2009, 9:48 p.m.*

Number theory has all kinds of identifiers for mathematically unusual numbers. Take Keith numbers, for instance. I think it is easier to give an example of a Keith number than to explain mathematically what it is, so here goes.

Is 14 a Keith number? Yes. Why?

start a Fibonacci-like sequence with the individual digits of 14, and each subsequent number is the sum of the previous 2 (because 14 has 2 digits) numbers:

1,4,(1+4=)5, (4+5=)9, (5+9=)14. Since 14 is in the sequence that begins with 1,4, it is a Keith number.

Is 30 a Keith number? No. Why?

3,0, (3+0=)3, (0+3=)3, (3+3=)6, (3+6=)9, (6+9=)15, (9+15=)24, (15+24)=39. Since 30 is not in the sequence, it is not a Keith number.

What about 3 digit Keith numbers? Is 145 a Keith number?

1,4,5, (1+4+5=)10, (4+5+10=)19, (5+10+19=)34, (10+19+34=)63, (19+34+63=)116, (34+63+116=)213, so 145 is not a Keith number.

So, the challenge is to write a calculator program (RPN, RPL, BASIC, pick your favorite language and calculator) to identify as many Keith numbers as you can. You can verify your results by Googling Keith number. And post your code so we can all benefit from your brilliance!

I can't think of any practical reasons that Keith numbers are important, but there aren't that many of them and this problem does lend itself to solutions on the devices we have come to know and love.

## Re: Challenge: Keith numbers

*Message #2 Posted by* ***Valentin Albillo*** *on 5 Sept 2009, 11:36 p.m.,*
*in response to message #1 by Don Shepherd*

Hi, Don:

In the HP-71B, this 5-liner is as straightforward as it gets:

```
1  DESTROY ALL @ OPTION BASE 1 @ DIM D(12) @ FOR N=10 TO 10^12-1 @ N$=STR$(N)
2  L=LEN(N$) @ FOR I=1 TO L @ D(I)=VAL(N$[I,I]) @ NEXT I
3  S=0 @ FOR I=1 TO L @ S=S+D(I) @ NEXT I @ IF S=N THEN DISP N ELSE IF S>N THEN 5
4  FOR I=1 TO L-1 @ D(I)=D(I+1) @ NEXT I @ D(L)=S @ GOTO 3
5  NEXT N


>RUN


    14
    19
    28
    47
    61
    75
   197
   742
  1104
  1537
  2208
  2580
  3684
  4788
  7385
  7647
  7909
 31331
 34285
 34348
  ...
```

It can be optimized in a number of ways, such as using MATH ROM statements and avoiding loops by just adding one value and subtracting another to keep the running sum instead of computing it in a loop, etc, etc, but what the heck ? The above code took less than 2 minutes to write, enter, and run, and it delivers the correct results, so ...

Try to beat that time (from *nothing* to *correct results*) using RPL ... :-)

Best regards from V.

## Re: Challenge: Keith numbers

*Message #3 Posted by* ***Keith Midson*** *on 6 Sept 2009, 4:37 a.m.,*
*in response to message #2 by Valentin Albillo*

I like it for a whole lot of reasons ... Cheers, Keith

## Re: Challenge: Keith numbers

*Message #4 Posted by* ***Don Shepherd*** *on 6 Sept 2009, 11:52 a.m.,*
*in response to message #2 by Valentin Albillo*

Thanks, V, great solution. I'm working on an RPN solution to run on the super-fast 12c to see what it can do with this problem.

## Re: Challenge: Keith numbers on HP-28S

*Message #5 Posted by* ***C.Ret*** *on 7 Sept 2009, 5:08 a.m.,*
*in response to message #2 by Valentin Albillo*

> Quote:
>
> Try to beat that time (from *nothing* to *correct results*) using RPL ... :-)

Well, 2 mins is really a challange.

Except if you are using the ISKEITH? binary function on your HP-28S which returns True(1) for any keith number k given as argument or False(0) otherwise.

```
« DUP ->STR DUP SIZE
  ->  k T n                  @ k-number, T is string representation, n is length
     « 1 n FOR i
          T i i SUB STR->    @ Put any digit of the k-number in the stack
       NEXT
       DO                    @ main loop performing sum of the n-digits
          n ROLL
          n DUPN
          2 n START + NEXT   @ compute the n-digits sum (additing n-1 times)
          SWAP
          DROP               @ remove oldest digit from the stack


       UNTIL DUP k >= END    @ loop until sum egals or overpasses k
       n ROLLD
       n 1 - DROPN               @ remove remaining digts from stack, keeping last sum
       k ==                      @ perform test: last sum of n-digits egal k ?
      »
»
'ISKEITH?' STO
```

The ISKEITH? function/program may be used in any RPL calculator to list Keith's numbers; such as

```
« 10 9999 FOR i
     IF i ISKEITH? THEN i PR1 END
  NEXT
»


 14 19 28 47 61 75 197 742 1104 1537 2208 2580 3684 4788 7385 7647 7909 ...
```

But, **Don** is right, a lot of optimisation have to be carry out to beat the oberall 2 mins delay :-)

Especely with an HP-28S !

*Edited: 7 Sept 2009, 2:29 p.m. after one or more responses were posted*

## Re: Challenge: Keith numbers on HP-28S

*Message #6 Posted by* ***Don Shepherd*** *on 7 Sept 2009, 9:58 a.m.,*
*in response to message #5 by C.Ret*

C.Ret, thanks for your very elegant RPL solution to this problem. It "almost" makes me want to go out and buy a 50g and learn RPL. I had no idea that the 28s actually has an ISKEITH function, that is amazing.

Great work.

And thanks to Valentin for an equally elegant BASIC solution.

## Re: Challenge: Keith numbers on HP-28S

*Message #7 Posted by* ***Don Shepherd*** *on 7 Sept 2009, 6:26 p.m.,*
*in response to message #5 by C.Ret*

Oh, I think I see now, the 28s does NOT have a built-in ISKEITH function. You wrote it and called it ISKEITH, right? My understanding of RPL is not very advanced, I'm afraid.

Yeah, when I did this problem in RPN, I knew I wanted to avoid moving the digits around the registers and adding all the digits to create the next entry each time, and the key was to figure out how to replace the un-needed number (after saving it so you can subtract it from the new sum) with the new sum during each cycle. And that worked fine, even with the 12c's limited indirect addressing. And in lines 26-29 I double the current sum and subtract the un-needed digit from the sum to get the new sum. It took a lot of thought and sheets of paper for me to get this to finally work, but it was worth it.

I've thought about trying create a solution to this problem on the HP-65 or the 17bii+ solver, but the lack of indirect addressing killed that idea.

## Re: Challenge: Keith numbers

*Message #8 Posted by* ***C.Ret*** *on 7 Sept 2009, 5:27 p.m.,*
*in response to message #2 by Valentin Albillo*

I was trying to convert the 5-liner HP-71B program from **Valentin** to an oldest Pocket whith a rudimental BASIC (such as a SHARP PC-1211)

Quote:

```
1  DESTROY ALL @ OPTION BASE 1 @ DIM D(12) @ FOR N=10 TO 10^12-1 @ N$=STR$(N)
2  L=LEN(N$) @ FOR I=1 TO L @ D(I)=VAL(N$[I,I]) @ NEXT I
3  S=0 @ FOR I=1 TO L @ S=S+D(I) @ NEXT I @ IF S=N THEN DISP N ELSE IF S>N THEN 5
4  FOR I=1 TO L-1 @ D(I)=D(I+1) @ NEXT I @ D(L)=S @ GOTO 3
5  NEXT N
```

At first, string operation have to be avoid (not possible on a SHARP PC-1211). So mathematics have to be used in order to extract each digit from the N-number.

Second, to compute the sum of the l-digits, manipulation and shifting of D(i+1) to D(i) can easely be avoid. At each step, the new sum of the last l-digits, which can be express as :

```
1 4 5(=1+4) 9(=4+5)   14(=5+9)   23(=9+14)   ...
```

Each new term is egal to the double of the previous minus the droped digit :

```
1 4 5(=1+4) 9(=2*5-1) 14(=2*9-4) 23(=2*14-5) ...
```

This can spare the FOR/TO/NEXT loop computing the sum (see line 3) and D(i) transposition (line 4)!

So, this is a second version of **Valentin** HP-71B program, which stil can be optimize :

```
1 DESTROY ALL @ OPTION BASE 1 @ DIM D(99) @ FOR N=10 TO 99999
2 S=0 @ L=1+INT(LOG(N)) @ FOR I=1 TO L @ B=10^(L-I) @ D(I)=INT(N/B) MOD 10 @ S=S+D(I) @ NEXT I @ I=L
3 IF S<N THEN I=I+1 @ D(I)=S @ S=2*S-D(I-L) @ GOTO 3
4 IF s=N THEN DISP N
5 NEXT N
```

Or the really similar version adapted to SHARP PC-1211:

```
10: CLEAR :N=10
20: S=0:L=1+INT LOG N: FOR I=1 TO L:B=10^(I-L):X=INT NB-10*INT (NB/10):S=S+X:A(26+I)=X:NEXT I:I=26+L
30: IF S<N LET I=I+1, A(I)=S, S=2S-A(I-L):GOTO 30
40: IF s=N PRINT N
50: N=N+1:GOTO 20
```

And a second version of ISKEITH? :

```
« DUP LOG IP 1 +
  -> k n                            @ k tested number, n number of digits
   « 0                              @ initiate sum   S=0
     1 n FOR i
       k n i - ALOG / IP 10 MOD      @ put i-digit into stack
       SWAP OVER +                  @ add it to sum
     NEXT
     WHILE DUP k <                  @ loop while sum is less than k
       REPEAT
         n 1 + ROLL OVER 2 * SWAP -  @ new sum is 2x last sum minus least i-digit
       END
     n 1 + ROLLD n DROPN            @ save sum and remove remaining digits
     k ==                           @ test sum is k
    »
»
'ISKEITH?' STO
```

These three program are still relatively slow to list all Keith number since most of the running time is used to test any N sequnetially.

We are still waiting for a clever way to pick up keith-numbers !

## Re: Challenge: Keith numbers

*Message #9 Posted by* ***Egan Ford*** *on 7 Sept 2009, 5:49 p.m.,*
*in response to message #8 by C.Ret*

Quote:

> We are still waiting for a clever way to pick up keith-numbers !

You'll probably be waiting for sometime. I've looked into this on/off throughout the 3-day US holiday weekend and AFAIK the only known method is by *exhaustive search*. It'd be nice to find a way to eliminate a candidate sooner than later. Of course, I have not done an *exhaustive search* myself for a better solution. It's been a lazy weekend.

There are reverse Keith numbers as well, e.g. 341 following the same logic ends up as 143.

## Re: Challenge: Keith numbers

*Message #10 Posted by* ***Pal G.*** *on 8 Sept 2009, 10:41 a.m.,*
*in response to message #9 by Egan Ford*

Per Wolfram Mathworld:

```
"There is no known general technique for finding Keith numbers except by exhaustive search.
```

http://mathworld.wolfram.com/KeithNumber.html

## Re: Challenge: Keith numbers

*Message #11 Posted by* ***Valentin Albillo*** *on 8 Sept 2009, 9:17 a.m.,*
*in response to message #8 by C.Ret*

Hi, C. Ret:

Thanks for your interest in my instant-coffee HP-71B solution, but be aware that in your HP-71B version you've used **LOG** (base e-logarithm) when you actually want to use **LGT** (decimal logarithm) so your HP-71B version as written is faulty.

Also, in the HP-71B you can use **N DIV B** instead of **INT(N/B)** to perform integer division, etc. Not so in the PC-1211 BASIC dialect, which lacks **DIV**.

Best regards from V.

## Re: Challenge: Keith numbers

*Message #12 Posted by* ***Don Shepherd*** *on 7 Sept 2009, 2:12 a.m.,* *in response to message #1 by Don Shepherd*

Well, after considerable frustration with the 12c+ indirect addressing via the cfj command and a nasty bug that took hours to track down, here is a solution for the 12c. Enter the number you want to start with, like 10, and press R/S. It will run until you stop it, and it will display each Keith number and pause for 1 second. If you have an original 12c, it will take a rather long time. If you have the new 12c+ (Arm-based processor), it will go considerably faster. And if you have the 12c Windows-based emulator that HP is selling for $20, it will go faster still (although the original version would not handle the pse command correctly, but they sent me a fix for that).

Here is the code. I used all 5 TVM variables for purposes for which they were not intended.

```
01 clr fin
02 cf0
03 1
04 0
05 /
06 i
07 frac
08 1
09 0
10 x
11 cfj
12 rcl pv
13 +
14 pv
15 rcl i
16 intg
17 x=0
18 goto 20
19 goto 03
20 rcl n
21 fv
22 rcl cfj
23 pmt
24 rcl pv
25 cfj
26 2
27 x
28 rcl pmt
29 -
30 pv
31 rcl n
32 1
33 -
34 n
35 x=0
```

```
36 goto 38
37 goto 40
38 rcl fv
39 n
40 rcl pv
41 rcl 0
42 -
43 x=0
44 goto 46
45 goto 49
46 rcl 0
47 pse
48 goto 54
49 rcl pv
50 rcl 0
51 x<=y
52 goto 54
53 goto 22
54 rcl 0
55 1
56 +
57 goto 01
```

## Re: Challenge: Keith numbers

*Message #13 Posted by* ***Crawl*** *on 8 Sept 2009, 7:38 a.m.,*
*in response to message #1 by Don Shepherd*

Fun! I haven't done one of these in a while.

The gist of this program is that first the initial number's digits, then the running sums, are stored as entries in a row matrix.

You put the number D on the stack, and it exhaustively searches for all Keith numbers with D digits. So, on a real HP50g, if you put 2 on the stack and run the program, you get (after about a minute)

```
14
19
28
47
61
75
```

You might want to run it on an emulator for much larger values of D!

```
%%HP: T(3)A(R)F(.);
\<< 0 0 0 0 \-> D S N J M
  \<< 10 D 1 - ^ 1 + 10 D ^ 1 -
    FOR N N \->STR 'S' STO 1 D
      FOR J S J J SUB OBJ\->
      NEXT D ROW\-> 'M' STO
      DO 2 D
        FOR J M J GET
        NEXT M 1 GET 2 D
        FOR J M J GET +
        NEXT D ROW\-> 'M' STO
      UNTIL M D GET N \>=
      END M D GET N ==
      IF
      THEN N
      END
    NEXT
  \>>
\>>
```

## Re: Challenge: Keith numbers

*Message #14 Posted by* ***Don Shepherd*** *on 8 Sept 2009, 12:16 p.m.,*
*in response to message #13 by Crawl*

Thanks for another great RPL solution to this problem.

[ Return to Index | Top of Index ]

GTO Go back to the main exhibit hall