HP Forum Archive 15

Short & Sweet Math Challenge #10: Counting beans

*Message #1 Posted by **Valentin Albillo** on 2 June 2005, 12:37 p.m.*

Hi all,

This new **S&SMC #10** completes the "trilogy" of *find-the-number* style challenges, and once again, we're looking for a 10-digit integer number (what else ?) and the answer is unique. So grab your favorite HP (preferably) calculator/handheld and do write a program for it to try and solve

The challenge

Find a 10-digit integer number such that its 1st digit counts the number of 0's in the whole number itself, its 2nd digit counts the number of 1's, and so on till its 10th digit, which counts the number of 9's present. For example, this would be a solution:

```
    3541500926
```

were it not for the fact that there are two "0", not three; there's a single "1", not five, there's a single "2", not four; etc, etc.

The solution is **unique**. Numbers beginning by one or more "0"'s are *not* 10-digit numbers. As usual, you're expected to produce a program for your chosen calculator(s) that upon running will find the one and only solution. There's no need to let it run till it makes sure there are no others, but if you want to, so much the better.

Midway next week I'll post my solution (plus comments), which is a simple 5-line program for the HP-71B which finds the only solution in an average time of 15 seconds (nearly instantaneous under Emu71). It uses very, very little programmer's insight (see Recommended Guidelines, below), just the bare minimum and obvious heuristic to speed up the search. I may also provide an HP42S and/or HP-41C and/or HP-15C solution if available time permits. You'll have the whole weekend and then some to try your hand at it.

Recommended Guidelines

:

- Absolutely refrain from using a PC, laptop, or PDA (unless running some HP calc emulator/simulator) to solve the challenge. A Mathematica, Visual Basic, C++, or Java solution is *useless* to the intended purposes of this challenge, in fact actually *defeats* them and is to be considered unpolite on purpose.

- Refrain from using your own smarts to solve the challenge, then perpetrate a program which incorporates so much of your previous study of the problem that it has pretty little to do. In an extreme case, you find the solution entirely by hand, then your program simply prints it out !.

  On a more typical example, you let the puzzle in so solved an state that the program finds it only too easy to deal with the pitiful remains. That's *not* it. The idea is that **the machine** does most of the work for you, **not** the other way around.

So keep your heuristics *simple* and show instead your programming muscle (if any), not your own puzzle-solving abilities. Try to consider yourself a *puzzle-solver dummy* but an HP-calc *programmer genius*, and see what comes out. Or else.

Best regards from V.

Edited to correct a major typo, doesn't affect any attempts to solve it

Edited: 3 June 2005, 8:54 a.m. after one or more responses were posted

Re: Short & Sweet Math Challenge #10: Counting beans

Message #2 Posted by . on 3 June 2005, 2:44 a.m.,
in response to message #1 by Valentin Albillo

Hi,

This is an interesting challenge. How do you keep coming up with them?

> Quote:
>
> It uses very, very little programmer's insight (see Recommended Guidelines, below), just the bare minimum and obvious heuristic to speed up the search

I've written a program that gives the correct answer, but it takes far too long. It simply starts at zero and checks every possible number. I'm probably missing the obvious but I can't think of any way to speed up the result right now *g*

.

Re: Short & Sweet Math Challenge #10: Counting beans

Message #3 Posted by ***Arnaud Amiel*** *on 3 June 2005, 3:30 a.m.,*
in response to message #1 by Valentin Albillo

This one looks easy and I could solve it in my head without paper in about 5mn so I though I will give it a try on my favourite the hp55. Unfortunately the lack of indirect register adressing and the few available steps make it quite hard. I will continue trying though.

My algorithm: Assume there are lots of zeros so start with 9000000000 And correct your number one step at a time:

```
9000000001
9100000001
8100000010
8200000010
...
```

You quickly get to the solution (3 or four more steps)

Arnaud

Edited: 3 June 2005, 7:08 a.m. after one or more responses were posted

Re: Short & Sweet Math Challenge #10: Counting beans

Message #4 Posted by ***Chris Dean*** *on 3 June 2005, 5:17 a.m.,*
in response to message #3 by Arnaud Amiel

Numbers of this type are called autobiographical numbers.

Re: Short & Sweet Math Challenge #10: Counting beans

Message #5 Posted by ***J-F Garnier*** *on 3 June 2005, 9:14 a.m.,*
in response to message #3 by Arnaud Amiel

I found a solution based on the same principle. This is an iterative algorithm that converge to the solution very quickly. It just counts the occurrence of each digit, and continuously update the digits until a stable state is found. But of course, it doesn't prove that the solution is unique.

I'm not sure that this is the kind of solution that Valentin is asking for, but I find it interesting.

J-F

```
  10 ! S&SMC #10 JFG June 2005
  20 OPTION BASE 0
```

```
 30 DESTROY D
 40 DIM D(9) ! the 10 digits
 50 K=0 ! iteration counter
 60 REPEAT
 70   F=0
 80   FOR I=0 TO 9 ! for each position
 90     N=0
100     FOR J=0 TO 9 ! scan the present values
110       IF D(J)=I THEN N=N+1
120     NEXT J
130     IF N>=10 THEN N=9 ! in case all digits are the same
140     IF N<>D(I) THEN D(I)=N @ F=1 ! and update the position if needed
150   NEXT I
160   K=K+1
170   DISP K;':'; @ FOR I=0 TO 9 @ DISP STR$(D(I)); @ NEXT I @ DISP
180 UNTIL F=0  ! until no more change
```

(Replace the REPEAT/UNTIL loop with a GOTO if you don't use the JPC Rom)

Edited: 3 June 2005, 9:16 a.m.

Re: Short & Sweet Math Challenge #10: Counting beans

*Message #6 Posted by **Thibaut** on 3 June 2005, 10:53 a.m.,*
in response to message #3 by Arnaud Amiel

In order to check my algorithm, i did in excel first.

BUT : it doesn't work with all numbers : if the sequence 7110000100 is reached, it generates 6300000100 wich loops back to 7110000100. (it does work with 6548456111 for example)

Arnaud, I looked at your iteration and I'm under the impression that you apply "too fast" the calculation. My sequence starting with 9000000000 (I had the same idea as yours) is :

9000000000 9000000001 8100000001 7200000010 7110000100 6300000100 7101001000 6300000100

I've posted the excel file here

Your feedback is much appreciated !

Re: Short & Sweet Math Challenge #10: Counting beans

Message #7 Posted by ***Arnaud Amiel*** *on 3 June 2005, 2:14 p.m.,*
in response to message #6 by Thibaut

I was worried that there could be some numbers that wouldn't simplify but hadn't looked into it enough. I thought any input would converge towards the solution but it is clearly not the case. As my plans were to use a hp55, I would have actually started with 0000000000 which would have converged.

As to the simplification algorithm, you are also right, what I wrote does not make sense. I am still working on it but I hope to have something slightly faster (number of iterations) than what you wrote.

I also believe that I will give up on the 55 and go to the 49 (I don't have a 65 and my 97 is currently in pieces)

Thanks for pointing out that the algorithm is not always converging, my intuition was wrong so the input has to be chosen carefully.

Arnaud

Re: Short & Sweet Math Challenge #10: Counting beans

Message #8 Posted by ***Arnaud Amiel*** *on 3 June 2005, 6:21 p.m.,*
in response to message #7 by Arnaud Amiel

Still no answer for the hp55 but I tried my algorithm on the 49g+ and I get the right number in 5.56 seconds. The program will run on a 48G and with very little changes on an S(X) or a 28 (The REVLIST command is used to nicely display the final result)

Here is the RPL code:

```
<<
[ 0 0 0 0 0 0 0 0 0 0 ] -> N   @we work on 10 digits array
@Thibaut noted above that the starting value may be important. His example does not work here though
<<
DO
  N @to compare before and after processing
  0 9 FOR I
          N I DUP2
          0 -> V C  @The Value we check, the Counter of digits
          <<  OBJ-> OBJ-> DROP @We get all the digits on the stack
          1 SWAP FOR A
                     IF V == THEN 'C' INCR DROP END @We count for the Value
          NEXT
          C @Number of digits with the Value
          >>
          SWAP 1 + SWAP PUT 'N' STO @We update the table
  NEXT
```

```
UNTIL N SAME END @We have to the solution
N
>>
@Next we transform the array into a real
OBJ-> OBJ-> DROP ->LIST REVLIST OBJ-> DROP
1 9 FOR I 10 * + NEXT
>>
```

I know some people around here think RPL does not look nice but it does the job...

Arnaud

Edited: 4 June 2005, 2:27 a.m.

Re: Short & Sweet Math Challenge #10: Counting beans

Message #9 Posted by ***chris dean*** *on 3 June 2005, 2:39 p.m.,*
in response to message #6 by Thibaut

Thibaut I think you answer is incorrect. It represents a number with 6 zeros, 3 ones and 1 seven which it is not.

Re: Short & Sweet Math Challenge #10: Counting beans

Message #10 Posted by ***Thibaut*** *on 4 June 2005, 3:21 p.m.,*
in response to message #9 by chris dean

That's not my answer. Some "root" numbers generate the correct number. Simply open my excel file and you'll het the right answer. Others don't, therefore I wonder where my algorithm is wrong or to say this differently, whith what kind of numbers it works and what kind of numbers it doesn't.

Valentin, your support is much appreciated !

Re: Short & Sweet Math Challenge #10: Counting beans

Message #11 Posted by ***Chris Dean*** *on 5 June 2005, 4:36 a.m.,*
in response to message #10 by Thibaut

```
    Dim A(9) As Integer
    Dim B(9) As Integer
    Dim iRow As Integer
    Dim strOut As String
    Dim iCount As Integer
    Dim jCount As Integer
    Dim iFlag As Integer


    A(0) = 9


    iFlag = 1
    While iFlag = 1
        iFlag = 0
        For iCount = 0 To 9
            B(iCount) = 0
            For jCount = 0 To 9
                If A(jCount) = iCount Then
                    B(iCount) = B(iCount) + 1
                End If
            Next
            If A(iCount) <> B(iCount) Then
                A(iCount) = B(iCount)
                iFlag = 1
            End If
        Next
        strOut = ""
        For iCount = 0 To 9
            strOut = strOut + Trim(Str(A(iCount)))
        Next
        Print strOut
    Wend
```

The algorithm above does not oscillate and reaches the solution in about 4 or 5 prints. The reason it does not oscillate is the line where A(iCount) is set to B(iCount). This can be applied to your algorithm and hopefully solves your problem.

I found programming this on my HP49G+ and Hp17BII+ too stressful and after developing the algorithm in Excel modified it to run on an old Casio FX-700P which took ages to run! I really need an HP71!

Regards

Chris 123

Re: Short & Sweet Math Challenge #10: Counting beans

Message #12 Posted by ***Bram*** *on 3 June 2005, 8:17 a.m.,* *in response to message #1 by Valentin Albillo*

Hi Valentin,

I found the answer in less than a couple of minutes by hand, but, as you stated, the challenge is in the programming, so I took my HP-32SII and voilá.
A very brute force program that will find the answer after some ages of running. It tests a number in about 3 seconds, so.....
I cannot yet possibly think of a way to have it run faster. (but *that* wasn't the challenge ;-)

```
999999999
STO X
LBL A
1
STO+ X
10
STO i
0
LBL Z   ; Zero reg A .. reg J
STO(i)
DSE i
GTO Z
1.010   ; do ten times
STO T
RCL X
PSE
1E9
/
STO Z
LBL S   ; Split number into digits and count them
RCL Z
FP
STO Z
LASTx
IP      ; one single digit
STO i   ; save it as an index
ISG i   ; 0..9 becomes 1..10 (regs A to J)
ABS     ; dummy instruction
ISG(i)  ; increment occurance of this digit
ABS     ; dummy instruction
1E10
/
STO+ Z
10
STO* Z  ; digit has been added to the tail of the number
ISG T
GTO S
1.010   ; do ten times
```

```
STO i
LBL C   ; Compare each digit with occurance
RCL Z
FP
STO Z
LASTx
IP      ; one single digit
STO T   ; save it
1E10
/
STO+ Z
10
STO* Z  ; digit has been added to the tail of the number
RCL(i)  ; number of occurances of 'ranked value'
RCL T   ; position of digit, the 'ranked value'
x<>y?
GTO A
ISG i
GTO C
RCL X
```

thank you

Re: Short & Sweet Math Challenge #10: For the 15C

Message #13 Posted by ***Arnaud Amiel*** *on 5 June 2005, 2:44 p.m.,*
in response to message #1 by Valentin Albillo

Having given up on using the 55, I still wanted to try it in RPN. This is the same algorithm as I posted for the 49. The following program was written for the 15C on which it takes about 14mn to complete. It should be easy to port it to some other calcs with a dozen of inderectly addressable registers.

```
This just loops through B until convergence:


LBL A
CLEAR REG
GSB C
STO .3
LBL 0
GSB B
GSB C
RCL .3
TEST 5     ?X=Y
RTN
R
```

```
STO .3
GTO 0
```

This tidy the content of registerS 0-9

```
LBL B
.00901
STO .1
LBL 1
0
STO .0
.00901
STO .2
LBL 2
RCL .2
STO I
RCL .1
INT
RCL (i)
TEST 6     ?X#Y
GTO 3
1
STO+ .0
LBL 3
ISG .2
GTO 2
RCL .1
STO I
RCL .0
STO (i)
ISG .1
GTO 1
RTN
```

This takes the content of re 0-9 and returns a number

```
LBL C
.00901
STO .1
0
LBL 4
RCL .1
STO I
R|
10
*
```

```
RCL (i)
+
ISG .1
GTO 4
RTN
```

I hope there is no typo...

Arnaud

Re: Short & Sweet Math Challenge #10: For the 15C

Message #14 Posted by ***Valentin Albillo*** *on 6 June 2005, 9:38 a.m.,*
in response to message #13 by Arnaud Amiel

Hi, Arnaud:

Thanks for your interest in S&SMC#10, a few comments on your program for the HP-15C:

- You do use the constant ".00901" in several places. First of all, you don't need the "01" part, as all HP models implementing ISG and DSE consider the default increment to be 1 when not specified, so your constant need be just ".009", which is two steps shorter and faster.

- Also, as you're using it in several places, it will save program steps and be faster too if you store said constant in a register at the beginning of your program, then recall it as needed. As the constant is between 0 and 1, an ideal place is in the #RAN register, so simply

  ```
  .009
  STO #RAN
   ...
  RCL #RAN
  STO .1
  ```

  would save a lot of bytes and run faster, too, while still not using any numbered register.

- Remember the HP-15C does have RCL arithmetic, so steps such as

  ```
  RCL (i)
    +
  ```

  can be shortened to RCL+ (i), again saving steps and running time (as well as one stack level). Also, you don't need a final RTN, end-of-program-memory acts as a default RTN if encountered during program execution.

- You do make use of "." registers. Your program would be shorter and faster if you rearranged your logic to use .0 to .9 indirectly, saving 0-9 for direct operations instead. For instance, your sequence

  ```
  1
  STO+ .0
  LBL 3
  ```

  would then become a single ISG 0 instruction.

- You've arranged your logic to count 'upwards', which requires ISG and the ".009" constant. It would save program steps and execution time if you rearranged it to count 'downwards', changing that .009 to 9 and those ISG to DSE.

- The routine at LBL B is called just once (GSB B). You would save three steps (GSB, LBL, RTN) and a lot of time (as label search is so slow) by simply inserting its steps directly at the location of the call.

"I hope there is no typo..."

The best way to ensure this is simply to re-enter the program in your calculator from your own post. If it runs ...

Thanks for your contribution and

Best regards from V.

Re: Short & Sweet Math Challenge #10: For the 15C

*Message #15 Posted by **Arnaud Amiel** on 6 June 2005, 11:55 p.m.,*
in response to message #14 by Valentin Albillo

Thanks a lot for your comments. I am not really keen on rpn and this was my first program that I could call slghtly "evolved". I am much more comfortable with RPL although I am not well acquainted with all that was introduced in the 49...

Once again thanks a lot. And I am looking forward to your 15C solution.

Arnaud

[Return to Index | Top of Index]