HP Forum Archive 15

[Return to Index | Top of Index]

Short & Sweet Math Challenge #12: Squaring Cubes !

*Message #1 Posted by **Valentin Albillo** on 23 Nov 2005, 6:09 a.m.*

Hi all,

Here you are, the next outing in the S&SMC series, rounding out the first dozen with a simple but quirky math challenge to help you flex your HP-programming muscles. So, enter

The Challenge

Find the *smallest* number N whose square is the sum of more than three *consecutive* cubes, all greater than 1.

- All numbers in your solution must be integer, positive, and greater than 1.
- Your program must find and display the solution as: smallest number (N), first cube (C), last cube (L).

For example, these are **not** solutions:

```
  N   C   L             Sum of cubes                           Comments
-------------------------------------------------------------------------------------------------
 10   1   4 ->  1³ + 2³ + 3³ + 4³ = 100 = 10²        1³ is not greater than 1

204  23  25 ->  23³ + 24³ + 25³ = 41616 = 204²       there are only three consecutive cubes
```

Ye Olde Deadline:

Next Monday I'll post my original solutions (plus comments), namely:

- A reference HP-71B solution, which is a 3-line program (129 bytes) which finds the solution in a few minutes (less than 9 seconds running under Emu71 @ 2.4 Ghz). It's written using just standard BASIC, no ROM extensions, so it can be easily converted to any other programming languages, as demonstrated by

- A 54-step RPN program for the HP-15C, which is a direct, optimized conversion of the 71B reference solution. It finds the correct smallest value in under 1 hour 45 min when running on a physical HP-15C.

As a bonus, the almost-automatic conversion process from HP-71B's BASIC to HP-15C's RPN will be discussed in detail as well, statement for statement. Kinda "compiling source code BASIC to RPN", sort of ...

Before I do it, you might want to try your hand at it this next weekend. Solutions for any and all HP calcs are welcome, but you should first read

The Usual Caveat Computat:

- Try to achieve a proper balance between what *you* do and what *your program* does, i.e., the program should work for *you*, not the other way around; remember these challenges are intended for you to try and demonstrate your HP-calc programming abilities.

  In one extreme, you give no thought to the problem at all and simply code a pure brute-force search that takes *ages* to run; in the other extreme, you think long and hard about it and come to an analytical solution of your own, then have your program simply print the answer. The ideal is to think *just enough* to help the program *a little*, then concoct a clever program using those simple shortcuts and let it do the hard work for you, as it was meant to be.

- 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, say, is useless to the intended purposes of this challenge, in fact actually defeats them, *and submitting one is to be considered unpolite behavior*.

Best regards from V.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #2 Posted by ***Gerson W. Barbosa*** *on 23 Nov 2005, 12:00 p.m.,*
in response to message #1 by Valentin Albillo

Hello Valentin,

Sorry for posting just the answer but I am too busy at work to convert the algorithm to RPL or RPN, but I will do it later.

(79 seconds on a 8MHZ 200LX with QBASIC. I have also EMU71 installed on the 200LX. I think the conversion to HP71 Basic should be straightforward, but I am not sure about the efficiency of my algorithm)

```
  N   C   L
 312  14  25
```

Best regards,

Gerson.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #3 Posted by ***Gerson W. Barbosa*** *on 23 Nov 2005, 8:12 p.m.,*
in response to message #1 by Valentin Albillo

Hi Valentin,

Below is the EMU71B version. Not totally brute force but very far from your optimized solution. It takes less than **9 minutes** running under Emu71 @ 0.5 Ghz, that is, at least 12.5 times slower than the optimum solution. This means it would take more than **9 hours** on the 15C. There's also an RPL version (1h 6m 22s according to the 48GX timer..., perhaps due to bad adaption). Is there a prize for the least efficient program? :-)

I just hope at least the solution is correct.

I am looking forward to yours and other people's solutions.

Best regards,

Gerson.

```
-------------------------------------------------

15 N=14
20 N=N+1
30 Q=N*N
35 T=Q^(1/3)
40 FOR C=2 TO T-3
45 S=0
50 L=C
55 IF S>=Q THEN 80
60 S=S+L*L*L
65 L=L+1
70 GOTO 55
80 IF S=Q AND L-C>3 THEN DISP N;C;L-1 @ GOTO 95
85 NEXT C
```

```
90 GOTO 20
95 END


>RUN
 312.000000000  14.0000000000  25.0000000000


--------------------------------------------------



--------------------------------------------------
%%HP: T(2)A(D)F(,);
« TIME 0 'FOUND'
STO 14 'N' STO
  DO 1 'N' STO+ N
SQ DUP 'Q' STO 3
XROOT IP 'T' STO 2
'C' STO
    DO 0 'S' STO C
'L' STO
      WHILE S Q <
      REPEAT L DUP
SQ * 'S' STO+ 1 'L'
STO+
      END
      IF S Q == L C
- 3 > AND
      THEN 1
'FOUND' STO
      END 1 'C'
STO+
    UNTIL C T 3 -
== FOUND OR
    END
  UNTIL FOUND
  END TIME SWAP
HMS- N C 1 - L 1 -
»
--------------------------------------------------
```

Edited: 23 Nov 2005, 8:28 p.m.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #4 Posted by ***James M. Prange (Michigan)*** *on 23 Nov 2005, 9:16 p.m.,*
in response to message #3 by Gerson W. Barbosa

Based on what you've written, how about the RPL program:

```
\<< 312 14 25 \>>
```

;-)

Regards,
James

Edited: 23 Nov 2005, 9:18 p.m.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

*Message #5 Posted by **Gerson W. Barbosa** on 24 Nov 2005, 6:27 a.m.,*
in response to message #4 by James M. Prange (Michigan)

> Quote:
>
> Based on what you've written, how about the RPL program:
>
> \<< 312 14 25 \>>
>
> ;-)

Way shorter and faster! Would the table below (not tested) be of any help for finding a hidden pattern?

Regards,

Gerson

```
-------------------------

  312   14   25
  315   25   29
  323    9   25
  504   28   35
  588   14   34
  720   25   39
 2079   33   65
 2170   96  100
```

```
 2940  118  122
 4472   69  100
 4914   81  108
 5187   64  105
 5880   64  111
 5984  120  136
 6630  144  156
 7497   25  122
 8721  153  170
 8778  144  164
 9360  111  149
10296  133  164
10695   81  149
11024   21  148
13104  105  168
14160  118  177
16296  333  339
16380   78  182
18333   97  194
18810  176  220
22022  144  220
22330  225  259
23247  144  225
31248  217  279



--------------------------
```

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

*Message #6 Posted by **Valentin Albillo** on 24 Nov 2005, 8:17 a.m.,*
in response to message #3 by Gerson W. Barbosa

Hi, Gerson:

Gerson posted:

"Below is the EMU71B version. Not totally brute force but very far from your optimized solution."

> Thanks for your interest in my S&SMC#12 and your very interesting inputs, but first of all, I've never stated that my original solution is "optimized", far from it. I submitted to my own published "Caveats" and did not use specialized mathematical knowledge, just common 'engineering' sense. A truly optimized version would run much faster than mine, and yet I find your times intriguing:

"It takes less than 9 minutes running under Emu71 @ 0.5 Ghz, that is, at least 12.5 times slower than the optimum solution. This means it would take more than 9 hours on the 15C."

> My original solution (again, not 'optimum') runs in 9 seconds under Emu71 @ 2.4 Ghz. Correcting for the clock speed, that means that indeed yours is 12.5 times slower. But my HP-15C version takes 1.75 hours to run so if the same factor applies, yours would take nearly 22 hours to run , not 9 !

" There's also an RPL version (1h 6m 22s according to the 48GX timer..., perhaps due to bad adaption). Is there a prize for the least efficient program? :-)"

> Yes, there is, but I doubt yours qualifies. The least efficient ones are the ones not even posted; nay, not even *written*.

"I just hope at least the solution is correct."

> You can be rest assured that indeed it is. Congratulations !

"I am looking forward to yours and other people's solutions."

> My original (non-optimized) solution will be posted next Monday or sooner if interest in the challenge decays. Thanks again and

Best regards from V.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #7 Posted by ***Gerson W. Barbosa*** *on 26 Nov 2005, 4:31 p.m.,*
in response to message #6 by Valentin Albillo

Hi Valentin,

Thanks for your kind remarks, despite the definitely inefficient algorithm I presented. It was so bad I decided to start over this afternoon, now giving it a little more attention. Since many other excellent solutions have been presented so far, it's possible someone else may have used a similar approach. If such is the case, of course the credit goes to the first poster. I confess I haven't paid attention to the other algorithms as I intended to give it another try. However, I was amazed by the small running time they obtained.

The program below runs in 53 seconds under Emu71 @ 0.5 GHz, but the answer is found in about 11 seconds. There's a weekness in this method though, as the value of K in line 10 is rather arbitrary. Choosing a low K, there'll be no answer at all whereas a higher K will cause the program to take longer than needed. Also, though the solution is evident, it is not presented in the right order.

Best regards,

Gerson.

```
>LIST
10 K=100 @ FOR I=2 TO K-3 @ S=0 @ FOR J=I TO I+2 @ S=S+J*J*J @ NEXT J
20 IF J>K THEN 50
30 S=S+J*J*J @ J=J+1 @ IF FP(SQR(S))=0 THEN PRINT SQR(S);I;J-1
40 GOTO 20
50 NEXT I
>RUN
 323.  9.  25.
 312.  14.  25.
 588.  14.  34.
 315.  25.  29.
 720.  25.  39.
 504.  28.  35.
 2079.  33.  65.
 4472.  69.  100.
 2170.  96.  100.
>
```

Edited to include an RPN version:

This is a version for the HP-34C, the slowest programmable calculator I have. The solution is found in less than ten minutes. (kind of cheating since I have not proven why K=30 was enough :-)

```
001 h LBL A
002 30
004 STO 4
005 2
006 STO 1
007 h LBL 2
008 0
009 STO 0
010 RCL 1
011 STO 2
012 2
013 +
014 STO 3
015 h LBL 3
016 RCL 2
017 g x^2
018 h LSTx
019 *
020 STO + 0
021 1
```

```
022 STO + 2
023 RCL 3
024 RCL 2
025 f x<=y
026 GTO 3
027 h LBL 4
028 RCL 4
029 RCL 2
030 x>y
031 GTO 5
032 g x^2
033 h LSTx
034 *
035 STO + 0
036 RCL 0
037 SQRT
038 h FRAC
039 g x=0
040 GSB 6
041 1
042 STO + 2
043 GTO 4
044 h LBL 5
045 1
046 STO + 1
047 RCL 4
048 3
049 -
050 RCL 1
051 f x<=y
052 GTO 2
053 h RTN
054 h LBL 6
055 RCL 2
056 RCL 1
057 RCL 0
058 SQRT
059 R/S
060 h RTN


   A: 322 (6m 20s)
g Rv: 9
g Rv: 25
 R/S: 312 (+ 3m 2s)
g Rv: 14
g Rv: 25
```

```
 R/S: 315 (+ 4m)
g Rv: 25
g Rv: 29
 R/S: No more solutions for K=30 (+ 23s)
```

Edited: 26 Nov 2005, 10:15 p.m.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

*Message #8 Posted by **Eamonn** on 23 Nov 2005, 10:20 p.m.,*
in response to message #1 by Valentin Albillo

Hi Valentin,

Here is a four line basic program that finds the solution that Gerson posted in 32 seconds on my Sharp PC-1262.

```
10 N=1
20 G=2*((N+5)*N+10)*(2*N+5): FOR X=N TO 2 STEP -1:S= SQR (G): IF S= INT (S) THEN 40
30 G=G+X*X*X: NEXT X:N=N+1: GOTO 20
40 PAUSE S: PAUSE X+1: PAUSE N+4
```

Here is a RPN program for the HP-33s (and HP-32s/sII) that takes about 45 seconds to find the same solution. It should be easy to port to almost any RPN calc. that supports recall arithmetic. Given that it only uses 47 program steps, it may even be possible to fit it into the HP-25 if the user can recall the results from the memory registers.

Best Regards,

Eamonn.

```
P0001 LBL P
P0002 1
P0003 STO N
A0001 LBL A
A0002 RCL N
A0003 STO X
A0004 5
A0005 +
A0006 RCL *N
A0007 10
A0008 +
A0009 RCL N
A0010 2
```

```
A0011 *
A0012 5
A0013 +
A0014 *
A0015 2
A0016 *
A0017 STO G
B0001 LBL B
B0002 RCL G
B0003 SQRT
B0004 FP
B0005 x=0?
B0006 GTO E
B0007 RCL X
B0008 RCL* X
B0009 RCL* X
B0010 STO+ G
B0011 1
B0012 STO- X
B0013 RCL X
B0014 x>y?
B0015 GTO B
B0016 x<>y
B0017 STO+ N
B0018 GTO A
E0001 LBL E
E0002 LASTx
E0003 RCL X
E0004 1
E0005 +
E0006 RCL N
E0007 4
E0008 +
E0009 RTN
```

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #9 Posted by ***Chris Dean*** *on 24 Nov 2005, 5:35 a.m.,*
in response to message #8 by Eamonn

Eamonn

I am slightly confused by your code

Quote:

```
10 N=1
20 G=2*((N+5)*N+10)*(2*N+5): FOR X=N TO 2 STEP -1:S= SQR (G): IF S= INT (S) THEN 40
30 G=G+X*X*X: NEXT X:N=N+1: GOTO 20
40 PAUSE S: PAUSE X+1: PAUSE N+4
```

---

1. It seems as though the setting of G in line 30 is redundant as its value is over written at the start of line 20
2. Does the algorithm use the fact that (1^3 + 2^3 +.... + N^3) = N^2 * (N + 1)^2 / 4?
3. Using S=INT(S) is quite hit and miss, better to use code of the form
IF ABS(S - INT(S + 0.0005)) < 1.0e-6 THEN 40

I hope you are not offended by my comments.

Chris Dean

Edited: 24 Nov 2005, 5:49 a.m.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #10 Posted by ***Eamonn*** *on 24 Nov 2005, 1:14 p.m.,*
in response to message #9 by Chris Dean

Hi Chris,

> Quote:
>
> ---
>
> 1. It seems as though the setting of G in line 30 is redundant as its value is over written at the start of line 20
>
> ---

On line 20, G is initialized before the 'FOR' loop. On line 30, the value in G is updated every time through the 'FOR' loop. The program may be a bit confusing because the initialization of G and the 'FOR' are on the same line. However, the initialization of G does not occur each time through the FOR loop.

Perhaps the program is more understandable if break up lines 20 and 30 and write it like this.

(Note that I have also taken the liberty of fixing a bug in the program on line 20. Instead of counting down to 2, the FOR loop now counts down to 1. The original program found the correct answer, but it didn't test all cases. More details below.)

```
10 N=1
20 G=2*((N+5)*N+10)*(2*N+5)
25 FOR X=N TO 2 STEP -1:S= SQR (G): IF S= INT (S) THEN 40
30 G=G+X*X*X: NEXT X:
35 N=N+1: GOTO 20
40 PAUSE S: PAUSE X+1: PAUSE N+4
```

Here, G is initialized on line 20, but the FOR loop starts on line 25. I admit that the original submission was a bit more cryptic than it could have been - perhaps I should have foused on clarity instead of brevity.

> Quote:
>
> 2. Does the algorithm use the fact that (1^3 + 2^3 +.... + N^3) = N^2 * (N + 1)^2 / 4?

Here are some comments on the modified code above, that I hope make it clearer what is going on:

Line 20 calculates (N+4)^3 + (N+3)^3 + (N+2)^3 + (N+1)^3. This is the sum of the four consecutive cubes, from N+1 to N+4. If you multiply out the cubes in longhand and add them together, you will see that they are the same.

Line 25 starts a for loop counting from N down to 2. The square root of G is calculated and checked to see if it is an integer. If it is, then the program has found the solution and branches to line 40.

In line 30, X^3 is added to G. The first time through the loop, this adds N^3 to G, so that G becomes (N+4)^3 + (N+3)^3 + (N+2)^3 + (N+1)^3 + N^3 - ie the sum of five consecutive cubes from N^3 to (N+4)^3. After the NEXT X statement, X will be decremented by 1. Next time through the loop, (N-1)^3 will be added to G. The time after that (N-2)^3 will be added to G, and so on. The last cube to be added to G that is subsequency tested is 2^3. The last cube to be added is is 1^3, but this value is not tested, because the loop terminates after this is added.

Once the program has tested all sums from (N+4)^3 + ... + 2^3 and not found a solution, it increments N (line 35) and startes testing all the sums for a larger value of N.

On line 40, the results are printed. The challenge dictates that the results should be the square root of the sum (which is S), the smalled value in the series (which is X+1) and the largest value in the series (which is N+4).

> Quote:
>
> 3. Using S=INT(S) is quite hit and miss, better to use code of the form IF ABS(S - INT(S + 0.0005)) < 1.0e-6 THEN 40

Not sure about this. I think that the original test is fine in this case. G is an integer and the test is to see if its square root is also an integer. If the square root is an integer, then the original code should find it OK.

> Quote:
>
> I hope you are not offended by my comments.

Not in the least. Thanks for the feedback.

Best Regards,

Eamonn.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #11 Posted by ***Chris Dean*** *on 24 Nov 2005, 2:45 p.m.,*
in response to message #10 by Eamonn

Thanks for the response. I am enlightened.

HP-25 solution

Message #12 Posted by ***Eamonn*** *on 25 Nov 2005, 11:37 a.m.,*
in response to message #8 by Eamonn

Here is a solution for the HP-25. It works on the HP-25 simulator available fom this website, but I don't have a real HP-25 to test it on to see how long it takes to run. It requires 39 program steps and 3 registers. This program should be able to run on most RPN programmables with minor modifications.

I have also attached an updated version of my BASIC program that is a bit cleaner than the original one I posted. The HP-25 program uses the same algorithm.

Eamonn.

```
5
STO 0
RCL 0
ENTER
ENTER
ENTER
3
```

```
-
STO 1
*
5
+
*
3
-
3
*
STO 2
RCL 1
RCL 1
*
RCL 1
*
STO+ 2
RCL 2
SQRT
FRAC
x=0
GTO 37
1
STO- 1
RCL 1
x!=y
GTO 20
STO+ 0
GTO 3
LASTx
RCL 1
RCL 0
```

Basic program

```
10 N=5
20 G=3*(((N-3)*N+5)*N-3)
30 FOR X=N-3 TO 2 STEP -1:G=G+X*X*X,S= SQR (G): IF S= INT (S) THEN 60
40 NEXT X
50 N=N+1: GOTO 20
60 PAUSE S: PAUSE X: PAUSE N
```

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

*Message #13 Posted by **Thomas Radtke** on 24 Nov 2005, 8:53 a.m.,*
in response to message #1 by Valentin Albillo

Here's my solution for the 48G:

```
<< 1 0 0 0 0 0 0 -> N A B T I R S
<< TICKS 'S' STO
   DO
     -1 1 8 N * + sqrt + 2 / 1 + IP 'A' STO
     N SQ 3 XROOT 1 + IP 'B' STO


     A B FOR I
       I SQ I 1 + SQ * N SQ 4 * - sqrt 4 * 1 + sqrt 1 - 2 / IP 'T' STO
       I SQ I 1 + SQ * T SQ T 1 + SQ * - 'R' STO


       IF R 4 N SQ * - 0 == THEN
         IF A T - 4 >= THEN
           N T 1 +
           TICKS S - B->R 8192 / '1_s' ->UNIT
           KILL
         END
       END
     NEXT


     1 'N' STO+
   UNTIL 0 END
>>
>>
```

Result: 312 14 25, roughly 7'30" used.

I appologize for any bugs, I have no serial cable so had to type in the listing manually:(.

Thomas

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #14 Posted by ***Valentin Albillo*** *on 24 Nov 2005, 9:24 a.m.,*
in response to message #13 by Thomas Radtke

Hi, Thomas:

Thomas posted:

"I appologize for any bugs, I have no serial cable so had to type in the listing manually:(."

Much appreciated. Thanks for your (correct) solution and

Best regards from V.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #15 Posted by ***Don Shepherd*** *on 24 Nov 2005, 12:53 p.m.,*
in response to message #1 by Valentin Albillo

Hi Valentin. It is Thanksgiving Day in the US, and I'll take this opportunity to thank you for this challenge. I have just started teaching middle school math to 11- and 12-year olds, so I really should be planning lessons, but I could not resist this challenge.

Below is the code for 12c platinum, using indirect addressing (such as it is) to store the cubes of 2 - 31 so I would not have to recalculate them. It is always a challenge to use indirect on the 12c series (challenge is really not the proper word, but this is a family forum).

The code takes about 4 or 5 minutes to find solutions for n = 323, 312, 204 (which really does not qualify since it is only 3 cubes), and 315.

Thanks for your always stimulating challenges!

Don Shepherd Louisville, Kentucky USA

```
001  cf0
002  2
003  enter
004  enter
005  enter
006  X
007  X
008  cfj
009  rcl n
010  3
011  0
012  -
013  x=0
014  gto 019
015  3
016  2
017  +
018  gto 004
019  1
```

```
020  sto i
021  sto n
022  rcl cfj
023  sto 0
024  rcl i
025  1
026  +
027  sto pmt
028  sto n
029  rcl cfj
030  sto +0
031  rcl 0
032  sqrt x
033  intg
034  enter
035  X
036  rcl 0
037  -
038  x=0
039  gto 041
040  gto 052
041  rcl 0
042  sqrt x
043  r/s
044  rcl i
045  1
046  +
047  r/s
048  rcl pmt
049  1
050  +
051  r/s
052  rcl pmt
053  1
054  +
055  sto pmt
056  enter
057  3
058  0
059  -
060  x=0
061  gto 064
062  rcl pmt
063  gto 028
064  rcl i
065  1
066  +
```

```
sto i
enter
2
9
-
x=0
gto 000
rcl i
gto 021
```

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #16 Posted by ***Chris Dean*** *on 24 Nov 2005, 6:28 p.m.,*
in response to message #15 by Don Shepherd

It might be worth noting that the identity

SIGMA(1, N^3) = N^2 x (N + 1)^2 / 4

Therefore

SIGMA(C^3, L^3) = SIGMA(1, L^3) - SIGMA(1, (C - 1)^3)

= L^2 x (L + 1)^2 / 4 - (C - 1)^2 x C^2 / 4

= N^2

which greatly simplifies the sums without assuming that the minimum will have 4 elements. With a test for a minimum value the calculation time taken to solve the problem is drastically reduced. I do not think this detracts too much from the fun of the programming exercise.

I have a solution written in Java but would not dream of posting it as that would not be very polite. I am currently without a HP calculator and will be getting a HP-49G+ at Christmas.

Edited: 24 Nov 2005, 6:32 p.m.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #17 Posted by ***Thomas Radtke*** *on 25 Nov 2005, 2:05 a.m.,*
in response to message #16 by Chris Dean

My solution above uses this identity, plus I make some assumption about one of the boundaries of the series for any N and calculate the second boundary to meet N^2. It bites me that Eamonn has presented a different solution that beats mine hands down in any respect;).

Thomas

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

*Message #18 Posted by **Valentin Albillo** on 25 Nov 2005, 6:15 a.m.,*
in response to message #16 by Chris Dean

Hi, Chris:

Chris wrote:

"I have a solution written in Java but would not dream of posting it as that would not be very polite."

> Thanks. I know it's tempting but you should resist. Much appreciated, that you abide by the "rules".

"I am currently without a HP calculator"

> Suffer no more. Visit the web page of the truly wonderful, free HP-71B emulator Emu71 by Jean-François Garnier, and you'll get the functional equivalent of an HP-71B, plus HP-IL, Math ROM, Forth, even Assembler if you feel like it.
>
> It will allow you to enter my original solutions next Monday, other people's in BASIC, or develop your own. At tremendous speeds and using a full PC keyboard and screen for maximum enjoyment and productivity. It even provides a simulated disk drive or two, and up to 320 Kb or more of simulated RAM ! :-)
>
> Moreover, if you get to like the HP-71B functionality and programming, once you receive your 49 for Xmas you can also get HP-71X, another perfect emulator for the HP-71B by Hrastprogrammer, but this time running in your eminently portable 49 instead of a Windows PC.

Best regards from V.

Edited: 25 Nov 2005, 6:23 a.m.

Re: Short & Sweet Math Challenge #12: Squaring Cubes on HP 16C!

*Message #19 Posted by **Marcus von Cube, Germany** on 25 Nov 2005, 11:25 a.m.,*
in response to message #16 by Chris Dean

Since my name is part of the challenge I had to participate ;-)

The identity is worth much more than nothing, I used it in my HP 16C solution. The 16C is not the fastest, it takes his time (12'19" to be precise.)

GSB A starts the process from the beginning. After the run, N is shown; press R/S twice to see the values for C and L.

```
LBL A
 DEC
 2
 0
 WSIZE      set it big enough
 5
LBL 0
 STO 2      this is L
 ENTER
 *          no x^2 on the 16C :-(
 RCL 2
 1
 +
 ENTER
 *          x^2 again
 *          this is 4 times the sum of cubes from 1 to L
 STO 0      intermediate result, used later
 RCL 2
 3
 -
LBL 1
 STO 1      this is C
 1
 -
 ENTER
 *
 RCL 1
 ENTER
 *
 *          this is 4 times the sum of cubes from 1 to C-1
 RCL 0      same for 1 to L (see above)
 x<>y       we want a positive difference
 -          now we have 4 times the sum of cubes from C to L
 SQRT       sets the carry flag if X was not an exact square
 F? 4       query the carry
 GTO 2      next try, because value was not a square
 GTO 9      HEUREKA!
LBL 2
 1          lower bound for C
 RCL 1
```

```
 1
 -          new C := C-1
 x!=y       bound not yet reached?
 GTO 1      store new value for C and continue search
 1
 RCL 2      L
 +
 GTO 0      new value for L, continue search
LBL 9
 2
 /          we used 4 times the sum, correct the root
 R/S        show N
 RCL 1
 R/S        show C
 RCL 2
 R/S        show L
```

Marcus

Re: Short & Sweet Math Challenge #12: Squaring Cubes on HP 16C - Slightly modified

Message #20 Posted by ***Marcus von Cube, Germany*** *on 26 Nov 2005, 4:09 a.m.,*
in response to message #19 by Marcus von Cube, Germany

I did some optimization on the code and saved steps and execution time (now below 11') by rearranging the expression:

N^2 * (N + 1)^2 ==> (N * (N + 1))^2

The changes are after labels 0 and 1:

```
LBL A
 DEC
 2
 0
 WSIZE      set it big enough
 5
LBL 0
 STO 2      this is L
 ENTER
 ENTER
 1
 +
 *          L * (L + 1)
 ENTER      no x^2 on the 16C :-(
 *          this is 4 times the sum of cubes from 1 to L
```

```
 STO 0      intermediate result, used later
 RCL 2
 3
 -
LBL 1
 STO 1      this is C
 ENTER
 ENTER
 1
 -
 *          (C - 1) * C
 ENTER      no x^2 on the 16C :-(
 *          this is 4 times the sum of cubes from 1 to C-1
 RCL 0      same for 1 to L (see above)
 x<>y       we want a positive difference
 -          now we have 4 times the sum of cubes from C to L
 SQRT       sets the carry flag if X was not an exact square
 F? 4       query the carry
 GTO 2      next try, because value was not a square
 GTO 9      HEUREKA!
LBL 2
 1          lower bound for C
 RCL 1
 1
 -          new C := C-1
 x!=y       bound not yet reached?
 GTO 1      store new value for C and continue search
 1
 RCL 2      L
 +
 GTO 0      new value for L, continue search
LBL 9
 2
 /          we used 4 times the sum, correct the root
 R/S        show N
 RCL 1
 R/S        show C
 RCL 2
 R/S        show L
```

Marcus

Re: Short & Sweet Math Challenge #12: Squaring Cubes in 14 seconds on HP 33s

Message #21 Posted by ***Marcus von Cube, Germany*** *on 26 Nov 2005, 5:48 a.m.,*
in response to message #20 by Marcus von Cube, Germany

The same solution ported to the 33s takes only 14 seconds to run!

```
LBL A
 5          initial guess for L
LBL L
 STO L
 ENTER
 ENTER
 1
 +
 *
 x²         (L * (L + 1))²
 STO i      intermediate result
 RCL L
 3
 -          initial guess for C
LBL C
 STO C
 ENTER
 ENTER
 1
 -
 *
 x²         ((C - 1) * C)²
 RCL- i
 +/-        4 times the sum of cubes from C to L
 SQRT
 FP
 x=0?       check if exact square
 GTO Z      HEUREKA!
 1          lower bound for C
 ENTER
 +/-
 RCL+ C     C - 1
 x!=y?
 GTO C      bound not yet reached, store new C and try again
 RCL+ L     L + 1
 GTO L      new L
LBL Z
 LASTx      FP has destroyed the the root, get it back
 2          all values were * 4, correct the root here
 /
 STO N      final result
 VIEW N     show the variables
 VIEW C
 VIEW L
```

Thanks again for the formula, Chris!

Marcus

Re: Short & Sweet Math Challenge #12: Squaring Cubes in 24 seconds on HP 28s (RPL)

Message #22 Posted by ***Marcus von Cube, Germany*** *on 26 Nov 2005, 8:31 a.m.,*
in response to message #21 by Marcus von Cube, Germany

Here is my RPL attempt. On my 28s it takes 24 seconds:

```
<< 0 0 0 0 0 -> sl c l n done       @ local variables
   << 5                             @ initial guess for L
      WHILE
         'l' STO                    @ store new L
         l 1 + l * SQ               @ 4 * sum of cubes from 1 to L
         'sl' STO                   @ save it somewhere
         l 3 -                      @ inititial guess for C
         WHILE
            'c' STO                 @ store new C
            sl                      @ see above
            c 1 - c * SQ            @ 4 * sum of cubes from 1 to C-1
            -                       @ 4 * sum of cubes from C to L
            SQRT DUP                @ Square root on stack
            'n' STO                 @ and in N (for later display)
            FP 0 ==                 @ check if root is integer
            'done' STO              @ save result of check
            c 2 !=                  @ lower bound not reached
            done NOT AND            @ and no solution yet
         REPEAT
            c 1 -                   @ compute new C
         END
         done NOT                   @ no solution yet
      REPEAT
         l 1 +                      @ compute new L
      END
      n 2 /                         @ display corrected N
      c                             @ display C
      l                             @ display L
   >>
>>
'SMC12' STO                         @ give it a name
```

The program uses the logical variable 'done' to determine if the loops should continue or a solution is found already. The REPEAT keyword marks the spot where the decision about loop terminationis on the stack.

The same program takes about 5.5 seconds on a 49g+.

Marcus

Re: Short & Sweet Math Challenge #12: Squaring Cubes in 24 seconds on HP 28s (RPL)

*Message #23 Posted by **Chris Dean** on 27 Nov 2005, 10:41 a.m.,*
in response to message #22 by Marcus von Cube, Germany

This looks like a nice neat, understandable RPL solution.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

*Message #24 Posted by **Valentin Albillo** on 25 Nov 2005, 5:57 a.m.,*
in response to message #15 by Don Shepherd

Hi, Don:

Don wrote:

"It is Thanksgiving Day in the US, and I'll take this opportunity to thank you for this challenge."

> Why, thank you ! I'm glad you like it, thanks to you for your interest.

"I have just started teaching middle school math to 11- and 12-year olds, so I really should be planning lessons, but I could not resist this challenge."

> I really do envy you. Among my many main hobbies, teaching is my second vocation, and teaching mathematics would probably qualify as my first. I've always been absolutely upset by the fact that most kids (most people, actually) abhor mathematics and consider it boring and a real chore while, in my humble opinion, this only reflects the fact that mathematics are usually taught in a most boring way, using the most dreadful methodologies. If properly taught, mathematics can be fun, inspiring, enthralling, and one of the greatest intellectual pleasures there are in this world, even artistic if I may say so.

Below is the code for 12c platinum, using indirect addressing [...] The code takes about 4 or 5 minutes to find solutions for n = 323, 312, 204 (which really does not qualify since it is only 3 cubes), and 315.

> Excellent entry, and for the HP-12C no less. Full marks to you for this.

"Thanks for your always stimulating challenges!"

> You're welcome, indeed. Let me say once more that I really appreciate when people tell me, either in this forum or privately, that they do like my challenges, even if they post no solution, because concocting them takes a precious time that I must literally rob from my family time, which always leaves me with a feeling of guilt. So, it's nice to know that at least it isn't wasted and perhaps I might induce some people to be fond of things mathematical.

Now, that's your opportunity with those kids; make sure they consider the math class as one of the funniest around ! There are no boring maths, just boring teachers ! :-)

Best regards from V.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #25 Posted by ***Chris Dean*** *on 25 Nov 2005, 7:33 a.m.,*
in response to message #24 by Valentin Albillo

Hi Valentin

Thanks for the information, I will download the emulator and give it a whirl. With regards to your other response, in the late seventies I was a mathematics teacher (11-18 year olds) and loved every minute of it!

Regards

Chris Dean

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #26 Posted by ***Don Shepherd*** *on 25 Nov 2005, 4:19 p.m.,*
in response to message #24 by Valentin Albillo

Valentin, many thanks for your kind words. I agree, teaching math to kids is probably the best job in the world. It is certainly the hardest job I have ever had. The problem is, of course, that all kids just aren't interested in learning, and that makes it tougher to teach the kids who do want to learn. I've only been teaching for one month, and already I could write a book about it.

thanks again, Don Shepherd

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #27 Posted by ***Bram*** *on 26 Nov 2005, 3:22 p.m.,*
in response to message #1 by Valentin Albillo

> Quote:
>
> Before I do it, you might want to try your hand at it this next weekend.

Thanks Valentin for again an S&SMC, but alas, wrong weekend for me. I've hardly the time to even read this thread, but I'll definitely will do so anyway another time.
So once no contribution from me.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #28 Posted by ***PeterP*** *on 26 Nov 2005, 9:58 p.m.,*
in response to message #1 by Valentin Albillo

Valentine,

very brute force, yet in a bow in deference to Angel who recently brought to us the QRG to his fabulous SandMathIII, I elected to use some of his functions. And another bow to Meindert, as I use his genial MLDL to load the SandMathIII...

You can choose a starting value for the upper boundary. Using 5 (which is the firts one to fullfill the rules of the challenge) it takes a normal CX roughly 1'34". It has 55 Lines and 94 bytes.

Again, not very pretty, so please no yelling from the more talented crowd. Especially for all the mistakes that I'm sure I have in the code...

Thanks Valentine for those challenges, I love them!

Cheers

Peter

Here is my code: Some explanations: Reg M: Outer loop Counter for upper Bonudary L

Reg N: Inner Loop counter for lower Boundary C (from L-4 to 2)

Reg O: -3.999 Constant

Reg a: Value of Sum of x^3 from 1 to L

Lbl'DD

Sto M "Stores the starting value

Time

Sto 01 "For taking the time

-3.999

Sto O

Lbl 00 "The outer brute force loop for the upper boundary L

Rcl O

Rcl M

INCX "Thanks Angel!

Sto M

+

Sto N

Last X

Enter

X^2

+

X^2

Sto a "This is the Sum(x^3) from 1 to L (L in Reg M)

```
Lbl 01 "Inner brute force loop for lower Boundary C

Rcl a

Rcl N

Int

X^2

Last X

-

X^2

-

Sqrt

Int

Last X

X=Y?

Gto 03 "The first and smallest Solution is found

DSE N

Gto 01

Gto 00

Lbl 03 "Display the solution in Sequence L,C,N,Time

Time

Rcl 01
```

HMS-

x<>y

Rasp "Just a fancy Beep from Angel

View M "This is the upper boundary L

Stop

View N " This is the lower Boundary C

Stop

2

/

View X "This is N

Stop

CLA

"Time:

Arcl Y

Aview

Stop

End

The first version took about 2' and 10" and I had to use some additional "thinking of the programmer" to get it under 2'. For example one can see that the difference between the L sum and the C sum is always an even number. Hence I was able to save the "HalfX" (another handy Angel function) before the "Int, X=Y?", which takes a bit of time out of the loop, yet requires the "2 /" in the end to display the right N. Another time-saver was to calculate X*(X+1) and X*(X-1) out into X^2 +X and X^2-X, which is a tad faster to program on the 41 { [Rcl M, IncX, Rcl M, *] is a bit slower than [Rcl M, X^2, LastX, +]}

```
REPLACE THIS TEXT WITH YOUR LISTING
```

Edited: 26 Nov 2005, 10:02 p.m.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #29 Posted by ***PeterP*** *on 26 Nov 2005, 10:14 p.m.,*
in response to message #28 by PeterP

...and maybe someone can show me how to format the code as nicely and easily as some of the other HP-lovers. Sorry for the layout mess...

Now that I solved it for myself I went through the other posts. I'm impressed with all the different solutions for all the different calcs! Yet a bit surprised that we do not have another explicit 41 solution (my sole true love). I guess it is a bit too oldfashioned...

Thanks Valentin again!

Cheers

Peter

Formatting

Message #30 Posted by ***Karl Schneider*** *on 26 Nov 2005, 10:35 p.m.,*
in response to message #29 by PeterP

Peter --

Link to formatting techniques at the HP Forum main page

The bracketed "[]" commands you need at either end of the source code listing are "pre" and "/pre"

If nobody else responds to your first post (and if you gave it a password), you can still edit it to make it look right.

-- KS

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

Message #31 Posted by ***Gerson W. Barbosa*** *on 26 Nov 2005, 10:39 p.m.,*
in response to message #29 by PeterP

Hi PeterP,

> Quote:
>
> ...and maybe someone can show me how to format the code as nicely and easily as some of the other HP-lovers.

Just click on the Preformatted button then paste your preformatted listing between the [pre] and [/pre] delimiters. You can also type these delimiters manually:

For instance,

[pre]

```
1st line
2nd line
3rd line
```

[/pre]

will produce this:

```
1st line
2nd line
3rd line
```

rather than the following, if you omit them:

1st line 2nd line 3rd line

Regards,

Gerson.

Re: Short & Sweet Math Challenge #12: Squaring Cubes !

*Message #32 Posted by **PeterP** on 27 Nov 2005, 1:08 a.m.,*
in response to message #28 by PeterP

Thanks so much for the formatting tips, here is the formatted version. Learned something!

```
Some explanations:
Reg M: Outer loop Counter for upper Boundary L
Reg N: Inner Loop counter for lower Boundary C (from L-4 to 2)
Reg O: -3.999 Constant
Reg a: Value of Sum of x^3 from 1 to L {aka Sum(x^3;1,L)}


Lbl'DD  "This is were it all begins
  STO M   "Stores the starting value for L, normally will be 5
  Time
  STO 01  "For taking the time
  -3.999
  STO O   "For faster inner loop calculation, numbers are slow...


Lbl 00  "The outer brute force loop for the upper boundary L
  RCL O
  RCL M
  INCX    "Thanks Angel!
  STO M
  +
  STO N   "Inner loop boundaries for C; they are from L-4 to 1 excl
  Last X
  Enter
  X^2
  +
  X^2
  STO a   "This is the Sum(x^3) from 1 to L (L in Reg M)


  Lbl 01  "Inner brute force loop for lower Boundary C, calculating
           Sum(x^3;1-L) - Sum(x^3,1-C) = sum(x^3;C,L)
           and checks if it is a full square
    RCL a
    RCL N
    Int
    X^2
    Last X
    -
    X^2     "now we have sum(x^3;1,C)
```

```
    -       "Sum(x^3;1,L) - Sum(x^3;1,C)
    Sqrt
    Int
    Last X
    X=Y?    "Is it a full Square?
    GTO 03  "The first and smallest Solution is found
    DSE N   "if not, decrease C and hence add one element to the
             sum
    GTO 01  "Rinse and repeat
GTO 00  "Increase L


Lbl 03  "Display the solution in Sequence L,C,N,Time
  Time
  RCL 01
  HMS-    "Calculate elapsed time
  x<>y
  Rasp    "Just a fancy Beep from Angel
  View M  "This is the upper boundary L
  Stop
  View N  "This is the lower Boundary C STOp
  2
  /       "See explanation in original post
  View X "This is N
  Stop
  CLA
  "Time:
  ARCL Y
  Aview
  Stop
End
```

S&SMC#12: My original solutions and comments

*Message #33 Posted by **Valentin Albillo** on 28 Nov 2005, 7:28 a.m.,*
in response to message #1 by Valentin Albillo

Hi all:

First of all, many thanks for the overwhelming number of excellent solutions you've posted in so many versions for so many different HP models (and the odd SHARP), it's been really great to see so many of you interested in my humble challenge. Also, your kind comments are much appreciated, even by the ones who couldn't post a solution this time (Hi, Bram!), thank you so much.

Now these are my two original solutions, the reference one for the HP-71B, and the one derived from it for the HP-15C. As I told Mr. Barbosa in an early post, they aren't optimal in any sense, because though I fully knew about the formulae that could compute the sum of N cubes directly, without iteratively adding the cubes up, one at a time, I decided I would provide a solution similar to the one an individual not knowing the existence of said formulae could come up with, by using common engineering sense to speed the search instead of blindly relying in a pure brute force approach.

The HP-71B reference solution

That being so, the reference solution for the HP-71B does not use any formula at all, as seen in the following 3-line, 129 byte, GOTO-less program (arbitrary line numbers):

```
1 FOR N=15 TO 500 @ S=N*N @ L=INT((S/4)^(1/3)) @ FOR C=L TO 3 STEP -1
2 S=S-C*C*C @ IF S=0 THEN DISP N;C;L @ END ELSE IF S<0 THEN S=S+L*L*L @ L=L-1
3 NEXT C @ NEXT N @ DISP "Not found"
```

```
>RUN

 312  14  25
```

which takes some 9 seconds under Emu71 @ 2.4 Ghz, and a few minutes in a physical HP-71B. A brief explanation of its workings:

- Since the challenge asks for more than three consecutive cubes, and 1^3 is not legal, we must search from $2^3+3^3+4^3+5^3$ = 224 > 14.96^2 up, so we begin our search at N=15.
- We further arbitrarily restrict our search up to N=500, but this value doesn't affect execution time at all since the search will end as soon as the minimum solution is found, so we could put N=1000000 instead and it would make no difference.
- Since there must be at least 4 consecutive cubes in the solution, we'll start adding cubes from a maximum value depending on N, and go down till either we find a solution, or we reach 1. The maximum value for each cube must be near 1/4th of the value of N, since at least four of them will be added up and they're roughly the same size.
- Then, starting from the greater four cubes, we keep on adding smaller cubes till a solution is found. If our sum exceeds N, we subtract the largest cube and keep on adding smaller ones, and this process is repeated until either we've found a solution or we've reached to 1. In that case, this value of N can't be so expressed and we proceed to the next value for N.

Eventually the solution is found:

$$312^2 = 14^3 + 15^3 + 16^3 + 17^3 + 18^3 + 19^3 + 20^3 + 21^3 + 22^3 + 23^3 + 24^3 + 25^3 = 97344$$

for a total of **12** (S&SMC#**12**) consecutive cubes being added up. As some people discovered, there's a near-solution:

$$315^2 = 25^3 + 26^3 + 27^3 + 28^3 + 29^3 = 99225$$

adding up only 5 cubes. But the requirements were for the *minimum* value of N, and 312 is less than 315.

The HP-15C derived solution

There are two main points in creating a reference solution for the HP-71B, namely:

- Anyone can enter and run it, regardless of they having a physical HP-71B, thanks to the incredible and **free** Emu71, a near-perfect emulator for the HP-71B created by Jean-François Garnier, with the added advantages of much faster execution times and the convenience of full PC keyboard and screen. A solution for other models would only be accessible for people having those physical models or their emulators, possibly non-free, or much more onerous to get and install.
- HP-71B BASIC is extremely easy to understand, most specially for native English speakers, being quite similar to natural language, so it's actually very easy to translate the reference solution to any other programming language such as RPN, RPL, or C#, say. That wouldn't be the case for an RPN and, most specially, RPL solution that by their very nature are extremely cryptic and difficult to convert to non-stack machines.

To demonstrate this, the following HP-15C RPN solution was produced from the above reference solution for the HP-71B:

```
01 LBL A     19 RCL RAN#  33 X=0?     47 GTO 0
02   3       20 Y^X       34 GTO 3    48 LBL 3
03 1/X       21 INT       35 TEST 3   49 RCL 0
04 STO RAN#  22 STO 1     36 GTO 4    50 R/S
05 .003      23 RCL+I     37 RCL 1    51 RCL 3
09 STO I     24 STO 3     38 RCL*1    52 INT
10  15       25 LBL 1     39 RCL*1    53 R/S
12 STO 0     26 RCL 3     40 STO+2    54 RCL 1
13 LBL 0     27 INT       41 DSE 1
14 RCL 0     28 ENTER     42 LBL 4
15 X^2       29 X^2       43 DSE 3
16 STO 2     30  *        44 GTO 1
17  4        31 STO-2     45 ISG 0
18  /        32 RCL 2     46 LBL 0
```

which is 54 steps long and takes 1 h 45 min to find the solution. You can insert a pause statement (PSE) after step 14 RCL 0, in order to see how the search progresses. To run it:

```
RUN -> N
R/S -> C
R/S -> L
```

The conversion process (i.e.: compilation) from reference 71B BASIC to 15C RPN is easy and almost automatic, and goes like this:

- First, let's map the BASIC variables used to RPN registers, like this:

```
0: N
1: L
2: S
3: C
```

- Also, we'll optmize for speed by storing a needed constant (the lower limit of the cube search, in RPN's ISG/DSE convention) in register I, and 1/3 (for cube roots) in register #RAN:

```
I   : 0.003
#RAN: 1/3
```

  Now, we'll simple translate statement for statement, like this:

- (required RPN setup)

```
01 LBL A      (entry point)
02   3
03 1/X
04 STO RAN#   (speed optimization, for cube roots)
05 .003
09 STO I      (speed optimization, for ISG/DSE)
```

- 1 FOR N=15 TO 500

```
10  15
12 STO 0      (N=15; the upper limit is ignored, search till a solution is found)
13 LBL 0      (it's a FOR, so we'll loop to here)
```

- @ S=N*N

```
14 RCL 0      (N)
15 X^2
16 STO 2      (S=N^2)
```

- @ L=INT((S/4)^(1/3))

```
17  4
18  /         (S/4)
19 RCL RAN#   (1/3)
20 Y^X
21 INT
22 STO 1      (L=INT(S/4)^(1/3))
```

- @ FOR C=L TO 3 STEP -1

```
23 RCL+I      (we construct L.003, for ISG/DSE)
24 STO 3      (C=L.003)
25 LBL 1      (it's a FOR so we'll loop to here
```

- 2 S=S-C*C*C

```
26 RCL 3      (C, with .003 attached)
27 INT        (C by itself)
28 ENTER
29 X^2        (C^2)
30  *         (C^3)
31 STO-2      (S=S-C^3)
```

- @ IF S=0 THEN DISP N;C;L @ END

```
32 RCL 2      (S)
33 X=0?       (S=0?)
34 GTO 3      (yes, go deal with that case)
   ...
48 LBL 3      (here we deal with a solution)
49 RCL 0      (N)
50 R/S        (stop for the user to see it)
51 RCL 3      (C with 0.003 attached)
52 INT        (C by itself)
53 R/S        (stop for the user to see it
54 RCL 1      (L, and last step: end program execution)
```

- ELSE IF S<0 THEN S=S+L*L*L

```
35 TEST 3     (did we exceed N?)
36 GTO 4      (not yet, go on)
37 RCL 1      (yes, recall L)
38 RCL*1      (L^2)
39 RCL*1      (L^2)
40 STO+2      (put it back in the running sum)
```

- @ L=L-1

```
41 DSE 1      (decrease L by one)
42 LBL 4      (the search continues here)
```

- 3 NEXT C

```
43 DSE 3      (decrease C by one and see if we can loop)
44 GTO 1      (yes, go loop)
```

- @ NEXT N

```
45 ISG 0      (increment N by one)
46 LBL 0      (place holder, we'll loop indefinitely)
47 GTO 0      (go loop for another N)
```

- @ DISP "Not found"

```
              (no code, it never happens)
```

That's all. Thanks for your interest and

Best regards from V.

Re: S&SMC#12: My original solutions and comments

Message #34 Posted by ***Marcus von Cube, Germany*** *on 28 Nov 2005, 8:40 a.m.,*
in response to message #33 by Valentin Albillo

Hi Valentin,

the most interesting difference between your solution and mine is the way the search is done.

> Quote:
>
> Find the smallest number N whose square is the sum of more than three consecutive cubes, all greater than 1.

You take the directions literally in that you are building a sequence of squares and try to break each down to a sum of consecutive cubes.

It looks like Gerson and Thomas have used your approach but some of us seem to prefer *reverse* thinking: Build an ascending sequence of sums of cubes that adhere to the rules and check, if the result is an exact square.

The latter aproach seems to be faster, even if the formula that reduces the sum to a simpler term is not used (see Eamonn's solution).

Since your program needs the cube root to estimate L, it is not suitable for the 16C (my "starter" on this challenge). Luckily, the 16C has SQRT, even in integer mode.

What if the solution must be found *without* access to a square root implementation. What would be the quickest way to determine, whether an integer is an exact square? This is kind of a mini challenge...

Marcus

Re: S&SMC#12: My original solutions and comments

*Message #35 Posted by **Valentin Albillo** on 28 Nov 2005, 10:33 a.m.,*
in response to message #34 by Marcus von Cube, Germany

Hi, Marcus:

Marcus posted:

"You take the directions literally in that you are building a sequence of squares and try to break each down to a sum of consecutive cubes."

> Yes, I was trying to come up with the kind of solution that would be obtained when following the instructions 'faithfully'. Doing the opposite and using formulae was optimum but less 'obvious', so to say.

*"What if the solution must be found *without* access to a square root implementation. What would be the quickest way to determine, whether an integer is an exact square?"*

> You can try a set of prime powers' congruences but, in my experience, the fastest, simplest way is to compute an integer square root by Newton's iteration (i.e: $x_1=(x_0+N/x_0)/2$), which converges quadratically and runs very fast in any hardware's integer mode as the division by 2 is a mere shift), then check the resulting value by squaring it and testing against N.
>
> If the machine can't do integer division either, then a simple Newton iteration can produce reciprocals without division, using just multiplications, and once you've got the (presumably scaled) reciprocal, a final multiplication will give you N/x. Also, there are methods to compute integer square roots directly, by guessing and shifting, similar to long division performed by hand.

Best regards from V.

Re: S&SMC#12: My original solutions and comments

*Message #36 Posted by **Gerson W. Barbosa** on 28 Nov 2005, 5:36 p.m.,*
in response to message #34 by Marcus von Cube, Germany

07734 Marcus!

> Quote:

> It looks like Gerson and Thomas have used your approach but some of us seem to prefer reverse thinking: Build an ascending sequence of sums of cubes that adhere to the rules and check, if the result is an exact square.

Writing down the sequence of cubes on paper yesterday I had a glimpse of this approach. But I didn't like my second program because I had to guess the larger cube was below 30 in order to find the solution in less than 10 minutes on my slow 34C. I have to find out what I did wrong. Congratulations for your nicely explained RPL program.

Regards,

Gerson.

Re: S&SMC#12: My original solutions and comments

Message #37 Posted by ***Marcus von Cube, Germany*** *on 28 Nov 2005, 6:31 p.m.,*
in response to message #36 by Gerson W. Barbosa

Gerson,

the trick is to start with $5^3+4^3+3^3+2^3$ and then start shifting the bounds [L,C] = [5,2]:

1. Step: check, if the sum is a square; exit it true.

2. Step: reduce the lower bound C by 1. If it reaches 1, go to the next step, else go back to 1.

3. Step: increase the upper bound L by 1 and set the lower bound C to L-3; start over with step 1.

The algorithm arrives at the solution but It needs to be proved that the sequence of numbers which are tested in step 1 are really in ascending order so that the solution is the true minimum asked for in the challenge.

(In particular: is SUM(x=2 to L-4,x^3) < L^3 ? The term to the left is removed from the sum when L is increased and C is recalculated.)

Marcus

P.S.: "07734": I really had to type that one into an old calc with 7-segment display and turn it over... :-)

Edited: 28 Nov 2005, 6:48 p.m.