# HP Forum Archive 14

## My solutions & Comments [LONG]

*Message #1 Posted by* ***Valentin Albillo*** *on 26 Nov 2003, 4:53 a.m.*

Hi,

Thanks to all of you who took interest in my humble Mini-Challenge and did contribute your solutions, I hope you had a good time attacking it. My original solutions have more or less been duplicated by some of you (congratulations !), so these are my original versions and relevant comments:

- The standard, 'loop' solution:

```
01  LBL A
02  STO I
03  STO+I
04    1
05  LBL 0
06  DSE I
07  RCL*I
08  DSE I
09  GTO 0
10  RTN
```

  This is pretty straightforward, short and sweet, it's probably the simplest approach anyone would try, and has the maximum range (up to N=60), but it has a big problem: it's painfully slow. Something more elaborate is definitely required.

- Pascal's solution, based in

```
f(n) = (2n)!/(2^n * n!)
```

  is more or less the first one to be stumbled upon by any sophisticated user dealing with this problem. It's fast and short, but regrettably N cannot exceed 34 because else (2n)! would exceed 10^100. This is my original version for this kind of solution, which was one step shorter and used RI instead of R0, because R0 is dedicated for matrix operations, while RI is *not* specific for indexing as a number of registers can work as indexes in the HP-15C:

```
01  LBL A
02  STO I
03  RCL+I
04   x!
05   2
06  RCL I
07   y^x
08  LASTX
09   x!
10   *
11   /
12  RTN
```

- Stephan's solution (and several others as well) is the second one to be found, in an attempt to expand the range beyond 34. It makes good use of the [Py,x] function, using this formula:

  ```
  f(n) = P(2*n, n)/2^n
  ```

  which works up to n = 52, but no further. My original version for this kind of solution used instead the formula

  ```
  f(n) = P(2*n-1, n)/2^(n-1)
  ```

  which, though more complicated, can still be evaluated in just one more step. The idea was that P(2*n-1, n) is a number *smaller* than P(2*n, n) is (exactly half, to be precise) so perhaps the range could be extended to n = 53, instead of n = 52. Unfortunately that doesn't pay here, because while P(106, 53) exceeds 1E100, P(105, 53) exceeds that limit too, so nothing is gained in this case. My program evaluating that formula was:

  ```
  01  LBL A
  02  STO I
  03  RCL I
  04  DSE I
  05  RCL+I
  06  X<>Y
  07  Py,x
  08   2
  09  RCL I
  10  y^X
  11   /
  12  RTN
  ```

  which works for N=1 to N=52. Notice that for N=1, the DSE I at step 04 does decrement I to 0 and so causes step 05 to be skipped, but as it is a RCL+I, it would simply add 0 to the X register, so skipping it is of no consequence whatsoever and thus the final result is valid.

  The problem with this kind of solution is that while it greatly extends the range from N <= 34 to N <= 52, it still doesn't reach the maximum possible value N = 60, and worse, for large N it's *very* slow.

- Finally, Csaba was the first to stumble upon my intended original best solution for this mini-challenge, making use of the beautiful formula:

```
        f(n) = 2^n * (n-1/2)!/Sqrt(Pi)
```

  which nicely avoids the evaluation of (2*n)!, so greatly extending the range for N. It's quite peculiar that now the term 2^n appears in the numerator, rather than the denominator.

  That this formula actually works is easily demonstrated by making use of two simple, well-known facts, namely:

```
        n! = n * (n-1)!    and   (1/2)! = Sqrt(Pi)/2
```

  given that, we have:

```
        2^n * (n-1/2)! / Sqrt(Pi)


            =  2^n * (n-1/2)*(n-3/2)* ... * 5/2 * 3/2 * (1/2)! / Sqrt(Pi)


            =  2^n * (2*n-1)/2 * (2*n-3)/2 * ... 5/2 * 3/2 * (1/2)! /Sqrt(Pi)


            =  2^n * (2*n-1) * (2*n -3) * ... * 5 * 3 * Sqrt(Pi)/2 / 2^(n-1) / Sqrt(Pi)


            =  (2*n-1) * (2*n-3) * ... * 5 * 3,  q.e.d.
```

  and my original solution was:

```
         01  LBL A
         02    2
         03  X<>Y
         04  y^x
         05  LASTX
         06   .
         07   5
         08   -
         09   X!
         10   Pi
         11  Sqrt
         12   /
         13   *
         14  RTN
```

  which differs from Csaba's only in that he does first the multiplication, then divides by Sqrt(Pi), while I did exactly the opposite. In general, it's more advisable to first perform the division, then the multiplication, in order to avoid any possible overflow though in this particular case there's no overflow for N=60 in either case, but Csaba's version has a maximum limit of N = 60.436+ without overflowing, while my version extends the maximum limit to N = 60.555+, which is the theoretically largest possible value such that f(N) < 1E100.

There's also another important difference in this case between multiplying first, dividing second (as Csaba does) and doing the opposite as my solution does, and it has to do with rounding errors. For N = 10, both solutions perform as follows:

```
    Csaba's solution:   f(10) = 654,729,074.8
My original solution:   f(10) = 654,729,075
         Exact value:   f(10) = 654,729,075
```

this is, Csaba's ordering results in a greater rounding error in this particular case. It's thus advisable to perform the division first for reasons other than to avoid potential overflows and reduced maximum range.

My improved original solution also catered for the fact Stefan rediscovered, i.e., by performing a one-time initialization consisting in storing 1/2 and Sqrt(Pi) in registers you can save as much as 4 program steps and roughly 1 full second ! My original final best solution was this (with 0.5 in R0, Sqrt(Pi) in R1):

```
01  LBL A
02    2
03  X<>Y
04  y^x
05  LASTX
06  RCL-0
07   X!
08  RCL/1
09   *
10  RTN
```

which differs from Csaba's implementation of Stefan's idea not only in that the division is performed first, like in the previous solution, but also in that it uses R0 and R1 instead of Csaba's R8 and R9, the rationale being that R0 and R1 are *permanent* registers, which will *always* exist whatever the allocation of memory might be, while R8 and R9 could easily be unavailable, allocated for program steps or as part of the matrix/advanced functions pool. It may seem a minor point, but when you're all set for the very optimum solution, every detail counts.

IMHO, this is the best solution possible. It's very short (only 12 steps in machines without recall arithmetic, such as the HP-11C and HP-34C), has the largest possible range (up to N = 60.555+), works even for non-integer arguments, uses the least resources, and in my Brazilian-made HP-15C runs in 2.7 seconds for N = 52. I acquired this HP-15C recently (for US $100 !) and it came with no batteries so I fitted three new ones. Perhaps this accounts for its speed, or else perhaps Brazilian units feature a newer, slightly faster CPU, but it's certainly the fastest HP-15C among all my five units, and a pleasure to use !

Finally, some assorted values of f(N) as computed by my solution above:

```
f(Pi) =    19.40741413
f(10) =    654,729,075
```

```
                f(52) =    2.835225443 E+82
                f(60) =    6.972993462 E+98


                f(60.5557) = 9.998554144 E+99
```

Best regards from V.

## Valentin: Thanks for this challenge!

*Message #2 Posted by* ***Tizedes Csaba*** *on 26 Nov 2003, 5:14 a.m.,*
*in response to message #1 by Valentin Albillo*

And a solution with 8 steps ;)

```
LBL A
ENTER
RCL*2
RCL+1
*
RCL+0
e^x
RTN


 1.940320757E-2 STO 2
 2.804609382    STO 1
-8.634919234    STO 0
```

I don't tested it. It had +100%..-800% error on N=1..60

Thank you again, it was very interesting. It maked me a long sleepless night...! :)

Csaba

## Re: Valentin: Thanks for this challenge!

*Message #3 Posted by* ***Valentin Albillo*** *on 26 Nov 2003, 6:08 a.m.,*
*in response to message #2 by Tizedes Csaba*

Hi, Csaba:

Csaba posted:

**"Thank you again, it was very interesting. "**

You're welcome. Thanks to you both for your kind words and for your interest and contributions, though I must say I'm always in panic that no matter how quirky or obscure I may devise a 'challenge', Csaba is certain to solve it within 24 hours at most ! Worse yet, you usually duplicate my solutions nearly step-for-step !! :-)

Best regards from V.

## Re: My solutions & Comments [LONG]

*Message #4 Posted by **Patrick** on 26 Nov 2003, 7:33 p.m.,*
*in response to message #1 by Valentin Albillo*

Very nice, Valentin. I actually worked on this a bit (less than for your previous challenges I have to say) but couldn't get the speed to the level you advertized. Again, well done.

In the past, you've pointed out how I have assumed certain restrictions in your challenges that didn't exist. Well, not to be a party pooper, I have to point out one of yours now.

Your solution criteria mentions that the main evaluation is speed. You also mention resources used but you give the would-be puzzle solver no idea how to play these two competing factors off against each other. Therefore, I believe I would be justified in using speed alone to score my solution. That said, the fastest solution would be one which merely recalls the function value from pre-stored values in the registers (using the I register). The program would be very quick indeed and work for rather large values of n, at least 60.

Of course, I like your solution much better, but this underscores the problem of coming up with good evaluation criteria.

## Re: My solutions & Comments [LONG]

*Message #5 Posted by **Valentin Albillo** on 27 Nov 2003, 7:46 a.m.,*
*in response to message #4 by Patrick*

Hi, Patrick:

Patrick posted:

**"Your solution criteria mentions that the main evaluation is speed. [...] I believe I would be justified in using speed alone to score my solution. [...] the fastest solution would be one which merely recalls the function value from pre-stored values in the registers (using the I register). [...] this underscores the problem of coming up with good evaluation criteria."**

Strictly speaking, you're right, Patrick, but then it all comes to plain common sense, and fair judgment. I assume participants in this forum have lots and lots of both, so that no unnecessary, cumbersome wording has to be included to explicitly exclude each and every undesired possibility.

Else, I would have to resort to lots of 'small-print' nonsense and nitpicking just to avoid someone 'bending the rules' outrageously, and that would spoil all the fun, mine in devising the challenge, and contributors' in having to read and abide by the miriad silly details, don't you think ?

Particularizing for my latest challenge, when I stated that "speed" was "the main consideration", I intended that statement to be taken at face value within reason, this is to mean that having the choice between a 10-step, 8-seconds routine and, say, a 20-step, 3-seconds routine, the last was to be preferred. Though not stated explicitly, I did not intend it to mean that a 60-register (equivalent to 420-step), 0.5-seconds routine would be preferrable to the 20-step, 3-seconds one, because that would be utterly preposterous, and though technically speaking it would meet the 'published' conditions, common sense dictates that such a solution isn't really true to the spirit of the challenge, which was to provide an 'Arguably Useful ...' solution.

If you understood otherwise, perhaps it has something to do with the very different legal practices and jurisprudence between our own countries, so that expectations and experiences in legal matters aren't that mutually understandable.

Whatever the case, I will proceed likewise for any further 'mini-challenges' I may devise in the future: give explicitly some conditions and restrictions, and let contributor's common sense to fill in any gaps.

Thanks for pointing this out and

Best regards from V.

## Re: My solutions & Comments [LONG]

*Message #6 Posted by* ***hugh steers*** *on 27 Nov 2003, 8:02 p.m.,*
*in response to message #5 by Valentin Albillo*

thanks for posting the solutions, these were all interesting.

something ive been pondering over is the one i like best is your 14 step solution not requiring preloaded registers. ive always hated having to do this, even when it is faster, since it make a dependency on external state.

so my question is, can this be reduced in size even further. i've tried but cant make any progress. the line i thought most promising was to express f(n) as (n-.5)!/sqrt(pi)/(.5)^n. the idea was to combine the 2 and .5 constants in the program. however my attempts require stack manipulation that brings the total back to 14.

any ideas?

## Re: My solutions & Comments [LONG]

*Message #7 Posted by* ***Valentin Albillo*** *on 28 Nov 2003, 5:39 a.m.,*
*in response to message #6 by hugh steers*

Hi, Hugh:

Hugh posted:

**"thanks for posting the solutions, these were all interesting."**

I'm glad you liked them, and the 'mini-challenge' as well. In case you didn't have the opportunity, there were a number of them posted by me a few months ago, available in the archives.

**"ive always hated having to do this, even when it is faster, since it make a dependency on external state."**

Yes, I mostly agree. However, my first HP was an HP-25, and its severe limitations (only 49 steps for programs, no subroutines) meant that you were forced to resort to externally initialize constants in order to fit most complex programs, as you couldn't affort using program steps to do it, at all. After a while, you became accustomed to it, and thus initializing things externally became second nature eventually.

It was also a useful trick for much more advanced machines, such as the HP-41C, where you could manually store a large number of constants in registers, then initialize them 'externally' with a programmable RDATAX command. This would stop your program, ask for a data card, and load the numeric contents in the specified registers.

In the case of this 'mini-challenge', if it were a 'production' routine to be used as part of a larger program (some Taylor Series approximation, for example), it would make sense to pre-initialize those two constants, .5 and Sqrt(Pi) if only in order to gain additional speed.

**"so my question is, can this be reduced in size even further. i've tried but cant make any progress. the line i thought most promising was to express f(n) as (n-.5)!/sqrt(pi)/(.5)^n. the idea was to combine the 2 and .5 constants in the program. however my attempts require stack manipulation that brings the total back to 14."**

See ? :-)

Seriously, I don't think it can be reduced further, unless some other shorter formula does exist, which I doubt.

Thanks for your interest, and best regards from V.

## [VPN] Re: My solutions & Comments [LONG]

*Message #8 Posted by* ***Veli-Pekka Nousiainen*** *on 28 Nov 2003, 8:04 a.m.,*
*in response to message #1 by Valentin Albillo*

Valentin (not 50km walker Kononen, but 50 step programmer Albillo) wrote:

"original final best solution was this (with 0.5 in R0, Sqrt(Pi) in R1):

01 LBL A 02 2 03 X<>Y 04 y^x 05 LASTX 06 RCL-0 07 X! 08 RCL/1 09 * 10 RTN

which differs from Csaba's implementation of Stefan's idea not only in that the division is performed first, like in the previous solution, but also in that it uses R0 and R1 instead of Csaba's R8 and R9, the rationale being that R0 and R1 are *permanent* registers, which will *always* exist whatever the allocation of memory might be, while R8 and R9 could easily be unavailable, allocated for program steps or as part of the matrix/advanced functions pool. It may seem a minor point, but when you're all set for the very optimum solution, every detail counts. IMHO, this is the best solution possible. It's very short (only 12 steps in machines without recall arithmetic, such as the HP-11C and HP-34C), has the largest possible range (up to N = 60.555+), works even for non-integer arguments, uses the least resources, and in my Brazilian-made HP-15C runs in 2.7 seconds for N = 52. I acquired this HP-15C recently (for US $100 !) and it came with no batteries so I fitted three new ones. Perhaps this accounts for its speed, or else perhaps Brazilian units feature a newer, slightly faster CPU, but it's certainly the fastest HP-15C among all my five units, and a pleasure to use !"

AND I like to ask if replacing line 02 "2" with "RCL 0" and inserting after that "1/X" would result to a faster timing? Will it force yet another register to convert to program steps. Just curious since I have zero (0) HP-15C )-`: [VPN]

## Re: [VPN] Re: My solutions & Comments [LONG]

*Message #9 Posted by* ***Valentin Albillo*** *on 28 Nov 2003, 10:05 a.m.,*
*in response to message #8 by Veli-Pekka Nousiainen*

Hi, VPN:

VPN posted:

**"AND I like to ask if replacing line 02 "2" with "RCL 0" and inserting after that "1/X" would result to a faster timing?"**

No.

**"Will it force yet another register to convert to program steps."**

Yes.

**"Just curious since I have zero (0) HP-15C )-`: "**

Too bad. I suggest you do something about it. I suggest you try the following no-nonsense approach:

- Try and get one in eBay. Lately there have been a lot for sale, and so prices are dropping somewhat. I would suggest looking for one EXACTLY AFTER XMAS, because by then most people will have no money left in their pockets at all, and/or the slightest desire to engage in an auction to fork out some hefty amount for a very expensive, not-that-essential old gadget. I got two of mine a year ago exactly that way, for less than $100. Remember, just after Christmas (let's say January 7th, or so).
- MoHP Classified, same strategy.
- Post a message at your faculty (if student) or workplace . Many colleagues do have some old Voyagers at home, or their relatives do, and would welcome the opportunity to get rid of them. I got one excellent HP-15C that way, for under $80, his previous owner didn't even remember the model, he asked me if I wanted "an old calc, an HP-11C I think" he had at home, but when he brought it to me, it was actually an HP-15C !

Best of lucks. Try the after-Xmas trick and tell me !

Best regards from V.

## Re: [VPN] Re: My solutions & Comments [LONG]

*Message #10 Posted by* ***Veli-Pekka Nousiainen*** *on 28 Nov 2003, 10:26 a.m.,*
*in response to message #9 by Valentin Albillo*

Valentin: "Best of lucks. Try the after-Xmas trick and tell me !"

Thank you for the answers (which were exactly what I was afraid of) and thank you for the Xmas approach suggestion. I'll keep you informed if I ever get a HP-15C (again - sniff) [VPN]

PS: I try to include [VPN] to every subject and as "signature" since some clever spambots already use vpn-at-welho.com (with different reply to e-mail, mine is always something like DROP_vpn@...) Be aware of fake vpn, I will certainly not give anyone a mortgage loan, bigger brsts/pns, nor want help to transfer money from N-i.g+e_r*i/a [I'll keep the $$$ there (-; ] [VPN]