

HP Forum Archive 17

[[Return to Index](#) | [Top of Index](#)]

Short & Sweet Math Challenge #19: Surprise ! [LONG]

Message #1 Posted by [Valentin Albillo](#) on 18 June 2007, 8:14 p.m.

Hi all,

Almost 3 months have passed by since my latest *S&SMC #18: April 1st's Spring Special*, so it's high time to indulge in a new Short & Sweet Math Challenge^(tm), this time's *S&SMC#19: "Surprise !"*, where an assortment of individual subchallenges are issued, all of them having in common *a most unexpected, surprising result*, and further they're *graded* according to their difficulty, from the easiest (**F**) to the hardest (**A**), so that all of you may try your hand to at least some of them, if not all.

Three optional variants are suggested to allow you to *upgrade* your mark to a **C+**, **B+**, or **A+**. I'll provide my original solutions plus extensive comments to all graded problems A-F (but not for A+, B+, C+, these are truly left as an exercise for the reader, though I'll provide the final results and some comments as well).

The usual rules do strictly apply, briefly:

- Do **not** *google* for the answers, that's *pathetically lame* and unworthy of you
- Giving just the numeric result is *worthless* as well and gets you *no grade at all*: you *must* supply **code** for your favorite HP handheld (other makes also allowed), which must be a calculator/handheld computer or emulator/simulator, no code for desktop/laptop PC-class machines or PDAs.

I'll post my original solutions within a week of sorts, so you'll have plenty of time to concoct and post your solutions. Enjoy ! :-)

The Graded Challenge

Grade F:

Write a program to compute (but not output) the N^{th} term for the following sequences:

$$\text{Sequence 1: } u_1 = 0, u_2 = 1, u_{n+2} = u_{n+1} + u_n/n$$

$$\text{Sequence 2: } u_1 = 0, u_2 = 1, u_{n+2} = u_n + u_{n+1}/n$$

where the number of terms N (which can be distinct for each sequence) is either hardcoded or input, your choice. After computing (but not outputting) the N^{th} term, u_N , your program must then compute *and output* just the following values:

$$\text{Sequence 1: } L = N*1 / (u_N)^1$$

$$\text{Sequence 2: } L = N^2 / (u_N)^2$$

Now use your program to help you *try and predict*, for each sequence, the corresponding theoretical exact limit of L when the number of terms N tends to infinity.

I'll give my solution for the HP-71B which is a 2-line program which computes and outputs the corresponding values of L for $N = 13$ and $N = 13000$, respectively.

Grade D:

Using a model which either has built-in *graphics capabilities* or else can send output to a (perhaps emulated) *printer* (not necessarily graphical, alphanumeric output will do just fine), write a program which, given N , will autoscale, label, and plot the **ISin(X, N)** function, for arguments X belonging to the fixed interval $[0, 2\pi]$ at increments of $\pi/40$, say, where

ISin(X, N) is defined as **Sin(Sin(Sin(... Sin(X) ...)))**

i.e., the iterated Sine of X , where X is assumed to be in radians, and N is a positive integer, which is the number of iterations. For instance, this could be the resulting output in an alphanumeric-only printer for the case of $N=1$:

```

Maxim = 1.000
0.000 0.000
0.157 0.156
0.314 0.309
0.471 0.454
0.628 0.588
0.785 0.707
0.942 0.809
1.100 0.891
1.257 0.951
1.414 0.988
1.571 1.000
1.728 0.988
1.885 0.951
2.042 0.891
2.199 0.809
2.356 0.707
2.513 0.588
2.670 0.454
2.827 0.309
2.985 0.156
3.142 0.000
3.299 -0.156
3.456 -0.309
3.613 -0.454
3.770 -0.588
3.927 -0.707
4.084 -0.809
4.241 -0.891
4.398 -0.951
4.555 -0.988
4.712 -1.000
4.869 -0.988
5.027 -0.951
5.184 -0.891
5.341 -0.809
5.498 -0.707
5.655 -0.588
5.812 -0.454
5.969 -0.309
6.126 -0.156

```

which, of course, is **ISin(X, 1)** i.e. just **Sin(X)**. Notice the *auto-scaling* so that the plot takes the *whole width* of the output, the *labeled* and properly *rounded* X and Y coordinates, and the labeled *Maxim*, which is the maximum absolute value of the Y coordinate: you need to know it in order to autoscale the plot, so you might as well output it.

Once you've succeeded, run your program for various values of N such as **10, 100, 1000, ...**, (or **2, 4, 8, ...**), and, just by looking at the plots themselves and the values of *Maxim*, *try and describe what happens for increasing values of N* and *predict what happens in the limit when N tends to infinite*.

I'll give my solution for the HP-71B which is an 8-line program which produces sample output like the above for any N .

Grade C:

Find and output *all* values of $N < 10000$ such that N is *prime* and

$$P(N) = S(N)$$

where $P(N)$ gives the number of primes $\leq N$ and $S(N)$ gives the sum of the factorials of the digits of N .

For instance:

$$P(7) = 4, \text{ because there are 4 primes } \leq 7, \text{ namely } 2, 3, 5, 7$$

$$S(1234) = 33, \text{ because } 1! + 2! + 3! + 4! = 33$$

Your program must be *optimized primarily for speed*, then for program size, and it must exhaust the range, not stop altogether upon finding a solution.

I'll give my solution for the HP-71B, which is a simple *6-line program* which delivers the goods very quickly.

Note: Get a C+ !

Among the infinity of positive integers $N \geq 10000$, there's just *one and only one* additional solution. Find it !

Grade B:

Given a positive integer N , write a program to compute and output the value of S , defined as

$$S = \sum_{n=1}^N \frac{1}{(r_n)^2}$$

where r_n is the n^{th} positive root of $\tan(x) = x$, where x is expressed in radians.

Your program must be *optimized primarily for speed*, then size, and must be efficient enough to allow you to compute S for *large* values of N in *reasonable* times. It must also take special care *not to miss* any roots in range or *misidentify* a pole for a root.

Once written, compute the value of S for $N = 100, 200, \dots, 1000, 2000, \dots$, then try and *predict what the exact limit value should be when N tends to infinite*.

I'll give several very short and fast solutions for the HP-71B, to illustrate some important points. Running on Emu71, they can compute and process up to, say, *1,000,000 roots*, in very reasonable times.

Note: Get a B+ !

If you manage to solve the above, you can get a B+ by computing and, most specially, *identifying* the exact sum but this time using

$$(r_n)^2 + 1$$

in the denominators instead of simply $(r_n)^2$

Grade A:

As you may know, the limit of the sum of the reciprocals of the natural numbers 1 to N does *diverge* when N tends to infinite:

$$S = \sum_{N=1}^{\infty} \frac{1}{N} \rightarrow \infty$$

However, if we simply leave *out* of the sum just *those terms whose denominators include the digit 9* (such as 199, 34343900132, ...), the infinite sum then nicely *converges* to a finite value.

This said, you must write a program to compute *as accurately as possible* this finite value of the infinite sum. You must optimize your program for *both accuracy and speed*.

I'll give my original solutions for the HP-71B, which are *6-* and *10-line* programs. Both of them do compute the limit to a full 12-digit accuracy fairly quickly, but the longer one does it *60+ times faster*.

Note: Get an A+ !

Compute the limit value of the infinite sum excluding instead those terms which include the digit string '42', as in "HP42S" (i.e: 42, 78042998, 142883213044, etc). Also, see if you can *predict the approximate value* of the infinite sum when excluding terms which do include some given arbitrary non-periodic string of digits (say, 314, or 314159, or your phone number, etc)

That's all. Hoping you'll enjoy this S&SMC and waiting for your clever solutions ...

Best regards from V.

Edited to correct a typo in the initial conditions of Grade F above, as pointed by Peter P. Thanks a lot, Peter ! :-)

Edited to add a C+ exercise for the reader

Edited: 20 June 2007, 4:27 a.m. after one or more responses were posted

failed at F already

Message #2 Posted by **PeterP** on 18 June 2007, 11:37 p.m.,
in response to message #1 by Valentin Albillo

Just got home and saw with great delight another S&SMC!

However, not surprisingly I stumble already on the first one. There is some slight hope for me that I don't fully understand the nomenclature of the challenge, but more likely I don't get it. Please give me kind nudge...

You write (replacing brackets for your subscript)

$$u(1) = 0; u(2) = 0; u(n+2) = u(n+1) + u(n)/n$$

Setting n = 1 I interpret this as: $u(1+2) = u(1+1) + u(1)/1$. Substituting u(1) and u(2) gives me a u(3) of 0 which then in consequence means u(n) is zero |n

Please someone be so kind and help me see what my mistake is!

Thanks for another challenge Valentin, I know my productivity level for tomorrow already (and thanks to my early stumble I will get some sleep tonight :-)

Cheers

Peter

My Bad !! :-(Thanks, Peter ! :-)

Message #3 Posted by [Valentin Albillo](#) on 19 June 2007, 1:41 p.m.,
in response to message #2 by PeterP

Hi, Peter:

Actually, it was a typo on my part: I was very busy keying in all of this rather long stuff late in the dark at 3 AM and despite my best efforts to ensure full correctness, a typo crept in.

I've already edited it out in my main post above, so please resume your efforts with the now corrected initial conditions for both sequences featured in **Grade F** (there were a couple of 0's where 1's should have been).

Thanks a lot for spotting it out and sorry for the inconvenience.

Best regards from V.

My first 71b submission!

Message #4 Posted by [PeterP](#) on 19 June 2007, 3:32 p.m.,
in response to message #3 by Valentin Albillo

Okay, here is my clunky 71b code - the first try for an official SSMC on the 71b for me! (hey, one has to treasure the small pleasures in life, right?) And I readily admit that the recursive capability of the 71b made this one much easier to program on the 71b than a similar attempt on the 41c would have been.

Those two harmless looking sequences do indeed seem to converge to surprising numbers (e and pi).

First sequence:

```
10 Def FNV(n)
20 If n = 0 Then FNV = 0 Else if n = 1 then FNV = 1 Else FNV = FNV(n-1) + FNV(n-2)/(n-2)
30 End Def
40 Input "N=?";n @ Disp n/FNV(n) @ Beep
```

even reasonably low n (e.g. 13 as suggested by Valentin) lead to fairly accurate representation of e. for N=13 I get 2.71828182854

Second sequence:

```
10 Def FNV(n)
20 If n = 0 Then FNV = 0 Else if n = 1 then FNV = 1 Else FNV = FNV(n-2) + FNV(n-1)/(n-2)
30 End Def
40 Input "N=?";n @ Disp 2*n/(FNV(n))^2 @ Beep
```

This sequence seems to converge to pi, but MUCH much slower. I have not installed emu71 yet (I know, I know, sorry guys....) so I will not run it until 13000 in my physical 71b as Valentin seems to suggest, lest I miss my appointment with the end of the universe.

The max I dared was 13 again and that gave me 3.548026... Being this is pretty far from pi, I might go ahead and install emu71... :-)

Re: My first 71b submission!

Message #5 Posted by [Egan Ford](#) on 19 June 2007, 4:56 p.m.,
in response to message #4 by PeterP

Quote:

This sequence seems to converge to pi, but MUCH much slower. I have not installed emu71 yet (I know, I know, sorry guys....) so I will not run it until 13000 in my physical 71b as Valentin seems to suggest, lest I miss my appointment with the end of the universe.

Recursion can be a bit slow (even on EMU71). Try something like the following for sequence 2:

```
10 N=13000
20 T=TIME
30 U1=0 @ U2=1
40 FOR I=3 TO N
50 U3=U1+U2/(I-2)
60 U1=U2 @ U2=U3
70 NEXT I
80 DISP 2*N/U3^2;"IN";TIME-T;"SEC"
```

EMU71 output: 3.14171348646 IN 1.48 SEC
71B output: 3.14171348646 IN 618.29 SEC

Re: My first 71b submission!

Message #6 Posted by **PeterP** on 19 June 2007, 9:34 p.m.,
in response to message #5 by Egan Ford

Thanks Egan, very nice! Another little trick learned, very much appreciated!

Cheers

Peter

F is for FORTH

Message #7 Posted by **Egan Ford** on 20 June 2007, 11:21 p.m.,
in response to message #5 by Egan Ford

Quote:

```
EMU71 output: 3.14171348646 IN 1.48 SEC
71B output: 3.14171348646 IN 618.29 SEC
```

For fun I fabricated a fast 71B FORTH follow up. Given that this is a stack-based solution you could easily use a forty one.

I must say that the 71B + Math ROM + JPCx + 41/FORTH ROM is becoming my favorite calc. RPN/FORTH/BASIC + I/O--very nice.

The code below works like my small basic post above, but ~2x faster.

```
: F1
CLOCK FTOI
0. 1.
13001 3 DO
  FENTER
  Z RCL
  I 2 - ITOF
  F/
  F+
LOOP
13000 ITOF Y RCL F/ F.
." IN "
CLOCK FTOI
SWAP - ." SEC"
;
```

```
OUTPUT: 2.71828183554 IN 331 SEC
```

```
: F2
CLOCK FTOI
0. 1.
13001 3 DO
FENTER
I 2 - ITOF
F/
Z RCL
F+
LOOP
FENTER F* 2 13000 * ITOF Y RCL F/ F.
." IN "
CLOCK FTOI
SWAP - . ." SEC"
;
```

```
OUTPUT: 3.14171348646 IN 324 SEC
```

Grade C

Message #8 Posted by **PeterP** on 19 June 2007, 10:23 p.m.,
in response to message #4 by PeterP

Another rather long program, which however works... And it brought me into contact with the JPC-rom...

the answer appears to be 6521, which is a prime number, has 843 as the sum of the factorials and there are 843 prime numbers lower than 6521.

the code below first generates $p(x)$ up until the maximum relevant number <10000 (which is 6543, see below). It then loops through all possible numbers, calculates S , checks if N is prime and if $s=p(n)$, taking advantage of a few things.

Generally a digit needs to be 'worth' more than its factorial. So say 5! is 120 yet that 5 occupies the first digit in the number n . it provides only $p(5) \leq 5$ number of primes but contributes $120 > 5$ to the S , the sum of the factorials. based on this general observation and the handy function $nprim(1,n)$ from the JPC-Rom we can quickly say that

a) no digits >6 are allowed, as they 'contribute' too much via $n!$ to S than what they can contribute to N , even if they are the highest digit in N

b) for similar reasons, the second lowest digit can only go until 4 and the third lowest until 5.

c) the lowest digit can only go until 3. However, as N has to be prime we can also eliminate 2 here. this leaves us with $(0->6)*(0->5)*(0->4)*(1,3) = 7*6*5*4*2$ or 1680 test we have to make.

d) given that this is much less than 10000 one could have also replaced the first loop which generates $p(x)$ with the function $nprim(1,N)$ for those 1680 iterations. It turns out that this is slower than generating $p(x)$ for all 6543 numbers.

anyway, here is the listing

```
10 destroy all
20 dim p(6543) @ c=0
30 for j=1 to 6543
40   if prim(j)=0 then c=c+1
50   p(j)=c
60 next j
70 disp j;c @ beep !just for feedback
80 for b=0 to 6
90   if b>0 then b2=b*10^3 @ b3=fact(b) else b2=0 @ b3=0
100  for c=0 to 5
```

```

110  if b+c>0 then c2=c*10^2 @ c3=fact(c) else c2=0 @ c3=0
120  for d=0 to 4
130    if a+b+c+d>0 then d2=d*10 @ d3=fact(d) else d2=0 @ d3=0
140    for e = 1 to 3 step 2
150      n=b2+c2+d2+e @ s=b3+c3+d3+fact(e)
160      if p(n)<>s then 190
170      if prim(n)<>0 then 190
180      disp n;s;p(n)
190    next e
200  next d
210 next c
220 next b
230 destroy all
240 disp "Done!" @ beep 880

```

It runs approximately 5 sec on EMU71 (i got inspired and did install it. What a NICE piece of software!!!) and approximately 5 minutes on my real hp71 (side question: is there a similar speed-hack for the 71b as there is for the 41C?) One interesting side comment: I remember that there sometimes is a debate if 1 is a prime number or not. even the JPC-rom has different opinions here (i think the x-version says no, the version says yes). If one were to claim 1 as a prime number, the solution would be 13, which has a factorial sum $s(13)=7$ and 7 prime numbers (1,2,3,5,7,11,13).

This is fun!

Cheers

Peter

Edited: 20 June 2007, 1:22 p.m. after one or more responses were posted

Re: Grade C

Message #9 Posted by [Egan Ford](#) on 19 June 2007, 11:26 p.m.,
in response to message #8 by PeterP

1 is not a prime number.

<http://primes.utm.edu/notes/faq/one.html>

Re: Grade C

Message #10 Posted by [Valentin Albillo](#) on 20 June 2007, 4:45 a.m.,
in response to message #8 by PeterP

Hi, Peter:

Peter wrote:

"Another rather long program, which however works... And it brought me into contact with the JPC-rom..."

My, my, you're learning a lot, aren't you ? And having fun while at it ! :-)

"the answer appears to be 6521, which is a prime number, has 843 as the sum of the factorials and there are 853 prime numbers lower than 6521."

He, he, my time to correct your typo: you mean "there are **843** prime numbers lower than 6521", not 853.

"It runs approximately 5 sec on EMU71 (i got inspired and did install it. What a NICE piece of software!!!) and approximately 5 minutes on my real hp71"

There's something weird about those timings. I've found that a properly installed instance of Emu71 runs 250-350 times faster than a real HP-71B, so if it takes 5 minutes in the physical 71B, it should take approximately just *one second* when running under Emu71, not five, assuming you're using a 2 Ghz CPU or so. Perhaps you should check your installed Emu71 or stop other tasks while timing it, or check different hardware.

"If one were to claim 1 as a prime number [...]"

... you could claim as well that 3 is even, i.e., you *can't*. 1 is neither composite nor prime, it's a *unit* and that allows it an special status as regards to primality or compositeness. As such, 1 is *never* counted as a primer number.

"This is fun!"

I'm truly glad you're enjoying it. I've added a C+ variant to the challenge above, see if you dare to conquer it as well. It's not that easy, you know ... :-)

Best regards from V.

Re: Grade C

Message #11 Posted by [PeterP](#) on 20 June 2007, 1:24 p.m.,
in response to message #10 by Valentin Albillo

Thanks Valentin, I corrected my post. Turns out it was even worse, there was a slight typo in the listing as well..

C+ sounds daring, your looong time ago post about how to teach a 71 or 41 large number/high precision arithmetic rears its ugly head...

Cheers

Peter

Re: Grade C

Message #12 Posted by [Valentin Albillo](#) on 20 June 2007, 8:00 p.m.,
in response to message #11 by PeterP

Hi again, Peter:

Peter wrote:

"C+ sounds daring, your looong time ago post about how to teach a 71 or 41 large number/high precision arithmetic rears its ugly head..."

Not so, fortunately. The only other solution is well within the normal range for REAL numbers in the HP-71B and, matter of fact, any other HP calculator ever produced.

A little sleuthing will allow you to easily set an upper limit and useful, time-reducing heuristics for the search.

Best regards from V.

C+ solved

Message #13 Posted by [PeterP](#) on 21 June 2007, 1:14 a.m.,
in response to message #12 by Valentin Albillo

hmmm..

I new that my original approach to C would not work, so I changed it around. The following code delivers the first solution in ~1sec in EMU71 and a second solution(?) in ~14min.

5,114,903 seems to be another solution. There are 363035 prim numbers until it and the sum of its factorials is also 363035 unless I made a mistake, which is, alas, very likely.

The code below uses fprim from the JPC-rom to move from one prime to the next and then calculates the sum of the factorial using the string-access brackets. Given that for the second solution we knew we had to look for higher numbers it made sense to let the prim numbers lead the way rather than the factorials as in my original solution.

The nice thing is that this code is even shorter. If one were to take out the niceties like giving it a start-number, measuring the time etc it would be down to about 7 lines...

(30,40+50,60,70,80,90,150)

```

10 Input "Start=?";S @ P=Nprim(1,S) @ C=0
20 A$=Time$
30 Loop
40 T=Fprim(S+1) @ P=P+1 @ S=T
50 T$=STR$(T) @ L=Len(T$) @ X=0
60 For I=1 to L
70 X=X+Fact(Val(T$[I,I])) @ If X>P Then I=L+1
80 Next I
90 If X=P Then
100 Disp "Found a solution!! ";"Prim=";T;"FactSum=";X;"NumOfPrim=";P
110 Disp "StartTime=";A$;" EndTime=";Time$
120 Disp "Press key to continue...";Keywait$
130 Endif
140 C=C+1 @ C>500 then C=0 @ Disp "Cur Prim=";T;"P=";P
150 End Loop

```

Re: C+ solved

Message #14 Posted by [Egan Ford](#) on 21 June 2007, 2:11 a.m.,
in response to message #13 by PeterP

Quote:

5,114,903 seems to be another solution.

Actually I think its 5224903.

I have a 50g program that works the same way. A quick and dirty that runs slow.

Save this as 'DIGITFACT'

```

\<< 0 \-> X
\<< DUP
  WHILE 0 >
  REPEAT DUP 10 MOD ! X + 'X' STO 10 / IP DUP
  END DROP X
\>>
\>>

```

Save and run this:

```

\<< 843 \-> P
\<< 6521
  DO NEXTPRIME DUP P 1 + 'P' STO DIGITFACT P
  UNTIL ==
  END P
\>>
\>>

```

Edited: 21 June 2007, 2:21 a.m.

Re: C+ solved

Message #15 Posted by [PeterP](#) on 21 June 2007, 9:41 a.m.,
in response to message #14 by Egan Ford

yes, sorry, my typo. As you can tell from the time I run it, it was rather late....I even ran it twice and wrote down the result on a notepad here next to my computer. It says in big happy letters 5224903. Did'nt know I am that dyslexic (see the above type in my original solution of 853 instead of 843...) Thanks for pointing it out.

Cheers

Peter

Edited: 21 June 2007, 9:53 a.m.

Re: C+ solved

Message #16 Posted by [Arnaud Amiel](#) on 28 June 2007, 7:45 a.m.,
in response to message #14 by Egan Ford

I came up with something so similar it is spooky. Just I use STO+ and INCR and my functions are called S and N.

If I have time this weekend I want to put some Assembly and sysRPL in there.

Arnaud

Grade D

Message #17 Posted by [PeterP](#) on 20 June 2007, 7:10 p.m.,
in response to message #4 by PeterP

Learned some more about string-handling, yet do not have the patience to make it as pretty as VA's plot. However, from running the pgm it would seem that the curve becomes more and more a rectangular step function with n increasing. The question is, if that 'rectangle' will eventually become a flat line. Here is my amateurs guess: $\sin(x)=x-x^3/3$. In the limit, x^3 should converge faster to 0 than x so I'd say it stays a rectangle (meaning the integral of $|\sin(x,n)|$ over $[0,\pi]$ is >0 for $\lim n \rightarrow \infty$). But that's just a guess.

Here is the rather boring and un-elegant code. (Side-comment: as a beginner/novice the 71b is so much easier to program that I see I do significantly less fiddling than when working with the 41. The outcome is that I gets results faster, yet the code is often less elegant or intriguing, more straight forward, yet easier to follow. Unless it is authored by VA, naturally... :-)

```
10 Destroy N,Y,P,D$,I,J,C @ Radians
20 Input "N=?";N @ Dim y(80) @ p=0 @ c=80 @ dim d$[80]
30 Def fns(x,n)
40   t=x
50   for j=1 to n @ t=sin(t) @ next j
60   fns=t
70 end def
80 m1=100 @ m2=-100
90 for i=0 to 80
100  y(i) = fns(i*pi()/40,n)
110  m1 = min(m1,y(i)) @ m2 = max(m2,y(i))
120 Next i
130 d=m2-m1 @ s=d/c
140 disp "Min=";m1;'Max=";m2
150 for i=0 to 80
160  d$=space(c)
170  p=int((y(i)-m1)/s+0.5)
180  p=max(0,p) @ p=min(p,80)
190  d$[p]="x" @ Disp d$
```

```
200 disp keywait$
210 Next i
220 Disp "Done!"
```

Re: Grade D

Message #18 Posted by [Dave Shaffer \(Arizona\)](#) on 21 June 2007, 6:48 p.m.,
in response to message #17 by PeterP

I, too, think you will approach a square wave (i.e. a line above and below zero, by the same distance). Whether it stays a square wave or goes to zero depends on precision limits on calculating $\sin(x)$ where x is small. If for x small enough, $\sin(x)$ exactly returns x , then the square wave amplitude will be the x value at which $\sin(x)$ no longer gets any smaller.

For example, on my 42S, I find that taking a sine starting at 0.0004 requires some 50 key presses of the sin button until you get an answer that is not 0.0004 but rather 0.000399999 (remember: $\sin(\epsilon) \approx \epsilon$, for small ϵ). For larger values of x (i.e. where $\sin(x)$ was bigger to begin with), $\sin(x)$ will get smaller more quickly than for those values of x which gave a smaller value of $\sin(x)$ to begin with. Hence, the curve will flatten out more quickly where x was nearer to $\pi/2$ or $3\pi/2$.

Grade Bish

Message #19 Posted by [PeterP](#) on 21 June 2007, 1:33 p.m.,
in response to message #4 by PeterP

THIS IS A CORRECTED POST

my original post had an error in the if-then clause in Def FN due to a misunderstanding on my part on how the if-then with a followin @ works.

Now things are more 'surprising'. A little nifty free graphic package (its free, super easy and allows all sorts of graphs of functions, integrals, etc. It can be found [here](#)) helped me see the periodicity in the roots (every πi) with the first one after 0 being between $1.25*\pi$ and $1.5*\pi$ which I used for FNROOT.

The value for B seems to be 1/5th (I get 0.199989... for $n=100000$)

The value for B+ comes out to something like 0.194526... and I have no idea about the 'true' value.

Here is the now corrected code. However, WHY those are the correct answers, I could never figure out. I admit to cheating in the end and found the explanation - I would have NEVER found this one out!

```
10 Def FN(x)
20 f= tan(x)-x
30 if abs(f)<=1e-40 then f=0
35 fnf=f
40 end Def
50 Input "N=?";N @ S=0 @ Input "ad to square of root (0 or 1)",a
60 For i=1 to N
70 L=(i+0.25*PI()) @ U=L+0.25*PI()
80 Y=FNROOT(L,U,FNF(FVAR)) @ S=S+1/(a+y^2) @ if mod(i,500)=0 then disp i,2*s
90 next i
100 s=s*2 @ Disp "Sum=";s
```

Cheers

Peter

Edited: 21 June 2007, 7:06 p.m.

Re: Grade Bish ... not :-)

Message #20 Posted by [Valentin Albillo](#) on 22 June 2007, 5:51 a.m.,
in response to message #19 by PeterP

Hi, Peter:

Peter posted:

"THIS IS A CORRECTED POST"

Most unfortunately, the corrections fell short of perfection, there are still a considerable number of errors in your listing. Please take what follows as well-meaning *advice, not criticism*, given solely to allow you to improve your skills:

First of all, there are many syntax errors in your listing:

```
10 Def FN(x)
      ^
      |-- It should probably be FNF(x), the F is missing
```

```
20 f= tan(x)-x
30 if abs(f)<=1e-40 then f=0
35 fnf=f
40 end Def
```

While correct, this is unnecessarily convoluted, and the simpler and obvious:

```
10 DEF FNF(X)=TAN(X)-X
```

would be both shorter and faster, not being a multi-line user defined function and not needing to use an additional variable, not to mention the gains of supressing the unnecessary test.

Which is more, you can simply use **TAN(FVAR)-FVAR** directly in FNROOT, thus getting rid of the user-defined function definition, which will again save bytes and be much faster, as invoking user-defined functions is costly in the HP-71B.

```
50 Input "N=?";N @ S=0 @ Input "ad to square of root (0 or^
  1)",a
      ^
      |-- this must be a ";" instead of a ","

70 L=(i+0.25*PI() @ U=L+0.25*PI()
      ^
      |-- this parenthesis is never closed; besides,
      I don't quite get what PI() is intended to be,
      unless it's some ROM- or LEX keyword which
      does require that weird syntax; the built-in
      PI function does *not* admit "()"

80 Y=FNROOT(L,U,FNF(FVAR)) @ S=S+1/(a+y^2) @
  if mod(i,500)=0 then disp i,2*s
      ^
      |-- Whence came this 2 ?
      Also, y*y would be much faster than y^2

100 s=s*2 @ Disp "Sum=";s
      ^
      |-- ditto, whence came this 2 ? No such 2 does
      appear anywhere in the challenge's infinite sum
```

I think most of these errors are due to you painstakingly typing the listing into the posted message, either from your HP-71B display or from the Emu71 listing. As such, you're prone to typos, uneven casing, etc.

May I suggest that you simply develop your solution in Emu71, then, once it runs fine, simply execute LIST to have the listing appear in Emu71's simulated 80-column display, then *copy* the listing from Emu71's display and *paste* it in your message's text area. That would ensure absolute correction, no typos whatsoever, and would be both much faster and less tiring for you.

Now, I've corrected the above mentioned typos, but nevertheless the corrected listing does not produce the values you mention, far from it. I've traced it and it doesn't seem to properly generate the roots, as it generates and adds to the sum the same roots many times, omits others, and at times generates poles instead of roots.

Perhaps my corrections weren't enough, or perhaps something else is missing from your listing. Please follow my advice on generating and copy/paste-ing listings directly from Emu71, and re-post your routine again. I hope that, in the end, this will be both fun and instructive for you.

Thanks a lot for your interest, and

Best regards from V.

Re: Grade Bish ... not :-)

Message #21 Posted by **PeterP** on 23 June 2007, 1:33 p.m.,
in response to message #20 by Valentin Albillo

Valentin,

Thanks for your thoughtful comments and teaching, please know that yours (and all the others) kind advice is very much appreciated by me! I knew that I will open myself up to public failure when I decided to participate in this S&SMC with the 71b, where I can barely manage the 'novice' grade as of now. However, pretty much all in live for me is about learning so I'm just fine with 'failing' on the way.

Please find below some comments with regards to your advice. As you inducted already, there are a good many errors which happen due to me trying to type it in form my 71b. I took me a little while to figure out how to copy from a dos window (haven't used DOS in ages) but towards the end you will find the copy&paste original code from me (which actually does seem to produce the results mentioned). Using your suggestions, however, makes the code not only shorter but also run about twice as fast!!

Quote:

Which is more, you can simply use TAN(FVAR)-FVAR directly in FNROOT, thus getting rid of the user-defined function definition, which will again save bytes and be much faster, as invoking user-defined functions is costly in the HP-71B

This is an awesome trick, thanks a lot! On a general level (you also mention later on that y^*y is faster than y^2) is there anywhere a reference which shows the timing for various commands of the 71b and suggestions on how to improve execution speed. We had quite detailed lists for the 41 and knowing them makes a huge difference in final execution speed. Maybe you can point me in the right direction if there were such a list. Also, we could double the speed of the 41c on the hardware side and I wonder if there is a similar hack for the 71b available.

Quote:

I don't quite get what PI() is intended to be, unless it's some ROM- or LEX keyword which does require that weird syntax; the built-in PI function does *not* admit "()"

Thanks for pointing this one out. Funnily enough my 71b does not complain at all when I type pi(). When reading Joe Horn's book I misinterpreted the discussion he briefly has about PI and RES that PI is a function and hence needs the brackets. Clearly I was mistaken but as my 71b (or EMU71) did not complain I was never made aware of my mistake. Until VA came along to the rescue :-)

Quote:

Whence came this 2 ?

I over-read the word "positive" in your challenge. As the code only sums up the right hand (positive) side, I multiplied by 2 to get the total. Clearly, as in first grade, reading the question properly is 50% of the battle to getting the answer right...

Quote:

May I suggest that you simply develop your solution in Emu71, then, once it runs fine, simply execute LIST to have the listing appear in Emu71's simulated 80-column display, then *copy* the listing from Emu71's display and *paste* it in your message's text area. That would ensure absolute correction, no typos whatsoever, and would be both much faster and less tiring for you.

Oh yes indeed, you may! It is not only buggy (as figura shows) but bloody damn annoying, too! Please find below a copy and paste version of my original code, which runs and brings the mentioned results, but does not include all your other corrections and suggestions.

```
10 REAL Y
20 DEF FNF(X)
30 F=TAN(X)-X
40 IF ABS(F)<=1.E-30 THEN FNF=0 ELSE FNF=F
50 END DEF
60 INPUT "N=?";N @ S=0 @ INPUT "ad to square root of (0 or 1)";A
70 FOR I=1 TO N
80 L=(I+.25)*PI() @ U=L+.25*PI()
90 Y=FNROOT(L,U,FNF(FVAR)) @ S=S+1/(A+Y^2) @ IF MOD(I,500)=0 THEN DISP I;2*S
100 NEXT I
110 DISP "Sum=";2*S
```

This code takes about 1min26sec for N=10000 while a code with your suggestions takes only 46sec! That's an awesome improvement, thanks for sharing it with me. This is especially interesting, as the only reason to do the multiline Def FN was the advice that can be found in the manual for the math-rom when discussing FNROOT. They suggest this if statement to avoid 'overly long' run times. Did you ever find an application where their statements is correct and how does one best decide when to use it and when to ignore?

Again, thanks a lot for your kind advice. Unfortunately I will be out of a computer for the rest of the weekend, but I will see what I can do for A on Monday. And I'm looking forward to more learning ;-)

Cheers

Peter

Re: Short & Sweet Math Challenge #19: Grade C 15C solution.

Message #22 Posted by [Egan Ford](#) on 19 June 2007, 11:55 a.m.,
in response to message #1 by Valentin Albillo

Quote:

Grade C: Find and output *all* values of $N < 10000$ such that N is *prime* and

$$P(N) = S(N)$$

where $P(N)$ gives the number of primes $\leq N$ and $S(N)$ gives the sum of the factorials of the digits of N .

Answer: 6521

Time to solution: ~15 minutes*

Time to completion: ~21 minutes*

UPDATE: Normal speed 15C time to solution: ~9 hours, 40 minutes.

UPDATE: Normal speed 15C time to completion: ~14 hours, 45 minutes.

The program below will count up to 9999 checking for primes along the way. All numbers < 10000 will have prime factors < 100. If prime and < 100, then cache the prime number for future checks. A matrix is used for the cache. As primes are discovered P(N) is incremented and S(N) calculated and compared to P(N). If S(N)=P(N) and prime then display N, R/S, repeat, end at 10000.

*NOTE: Do not input this in your 15C and run it. It will take a very long time. I used Nonpareil after increasing the frequency 4651 times (no, it does not run 4651 times faster, there are other factors). UPDATE: Measured ~38x faster with modified Nonpareil.

```

001 LBL A           ;LBL A
002 CLEAR REG
003 FIX 0
004 CF 0
005 f MATRIX 0
006 2
007 5
008 ENTER
009 1
010 f DIM E
011 f MATRIX 1
012 3
013 STO E
014 5
015 STO 2
016 2
017 STO 3
018 LBL 0           ;LBL 0
019 4
020 10^X
021 RCL 2
022 X>=Y?          ;TEST 9
023 RTN
024 f MATRIX 1
025 LBL 1           ;LBL 1
026 RCL E
027 X=0?
028 GTO 2
029 /
030 FRAC
031 X=0?
032 GTO 3
033 1
034 STO+ 0
035 RCL 2
036 GTO 1
037 LBL 2           ;LBL 2
038 F? 0
039 GTO 6
040 9
041 7
042 RCL 2
043 X=Y?           ;TEST 5
044 SF 0
045 STO E
046 LBL 6           ;LBL 6
047 1
048 STO+ 3
049 GSB B
050 RCL 3
051 X!=Y?          ;TEST 6
052 GTO 3
053 RCL 2
054 R/S

```



```

055 LBL 3           ;LBL 3
056 2
057 STO+ 2
058 GTO 0
059 LBL B           ;LBL B
060 0
061 STO 5
062 RCL 2
063 LBL 4           ;LBL 4
064 X=0?
065 GTO 5
066 1
067 0
068 /
069 ENTER
070 FRAC
071 1
072 0
073 *
074 x!
075 STO +5
076 X<>Y
077 INT
078 GTO 4
079 LBL 5           ;LBL 5
080 RCL 5
081 RTN

```

Edited: 20 June 2007, 5:47 p.m. after one or more responses were posted

Re: Short & Sweet Math Challenge #19: Grade C 15C solution.

Message #23 Posted by [Thibaut.be](#) on 19 June 2007, 3:30 p.m.,
in response to message #22 by Egan Ford

Hi Valentin,

Why not, for the next S&SMC, calculate the ideal rhythmic on orabidoo ?

Re: Short & Sweet Math Challenge #19: Grade C 15C solution.

Message #24 Posted by [Valentin Albillo](#) on 20 June 2007, 5:25 a.m.,
in response to message #23 by Thibaut.be

Hi Thibaut,

Thibaut posted:

"Why not, for the next S&SMC, calculate the ideal rhythmic on orabidoo ?"

... Meaning !?

Best regards from V.

Re: Short & Sweet Math Challenge #19: Grade C 15C solution.

Message #25 Posted by [Thibaut.be](#) on 20 June 2007, 7:26 p.m.,
in response to message #24 by Valentin Albillo

Quote:

"Why not, for the next S&SMC, calculate the ideal rhythmic on orabidoo ?"

... Meaning !?

Best regards from V.

... meaning that it would be nice to make a cross-over between 2 passion. It happens that both of us enjoy HP calcs AND Mike Oldfield... So I'm looking forward to seeing a S&SMC combining maths AND Mike's music.

Any inspiration ?

Re: Short & Sweet Math Challenge #19: Grade C 15C solution. [OT]

Message #26 Posted by [Valentin Albillo](#) on 20 June 2007, 7:54 p.m.,
in response to message #25 by Thibaut.be

[Caveat reader: Off topic]

Hi, Thibaut:

Thibaut posted:

"It happens that both of us enjoy HP calcs AND Mike Oldfield..."

That's correct, last time I checked I had 100+ CD/DVD (about 2% of my whole CD/DVD collection) of his music, counting official releases, bootlegs, really really rare 'rarities', fan-made releases, the works !

"So I'm looking forward to seeing a S&SMC combining maths AND Mike's music. Any inspiration ?"

Intriguing request ... I think it might be possible, but would require people to also be aware of and pretty knowledgeable about specific works of Mike (say "Amarok", for instance) but the hard fact is he isn't known that well (if at all) in the US or most any other place in the world save Spain, UK, and some European countries, most regrettably.

100+ years from now, when he gets recognition as a true 'classic' genius 'a la Mozart' the situation will improve, hopefully.

Thanks for your appreciation of both my math efforts and most specially Mike's music, and

Best regards from V.

Re: Short & Sweet Math Challenge #19: Grade C 15C solution.

Message #27 Posted by [Egan Ford](#) on 19 June 2007, 6:34 p.m.,
in response to message #22 by Egan Ford

Edited submission, a bit smaller/2x faster/a bit cleaner.

Re: Short & Sweet Math Challenge #19: Grade C 15C solution.

Message #28 Posted by [Valentin Albillo](#) on 20 June 2007, 5:20 a.m.,
in response to message #22 by Egan Ford

Hi, Egan:

Nice effort ! I've added a C+ extension to the challenge above, perhaps you might want to conquer that as well, though it's far more challenging.

Thanks a lot for your interest and very nice solution (for the HP-15C no less), and

Best regards from V.

Re: Short & Sweet Math Challenge #19: Grade C 15C solution.

*Message #29 Posted by **Egan Ford** on 20 June 2007, 12:42 p.m.,
in response to message #28 by Valentin Albillo*

Quote:

Nice effort ! I've added a C+ extension to the challenge above, perhaps you might want to conquer that as well, though it's far more challenging.

Thanks, I'll work it if I have the time. All my spare time (what little I have) is being used for A/A+.

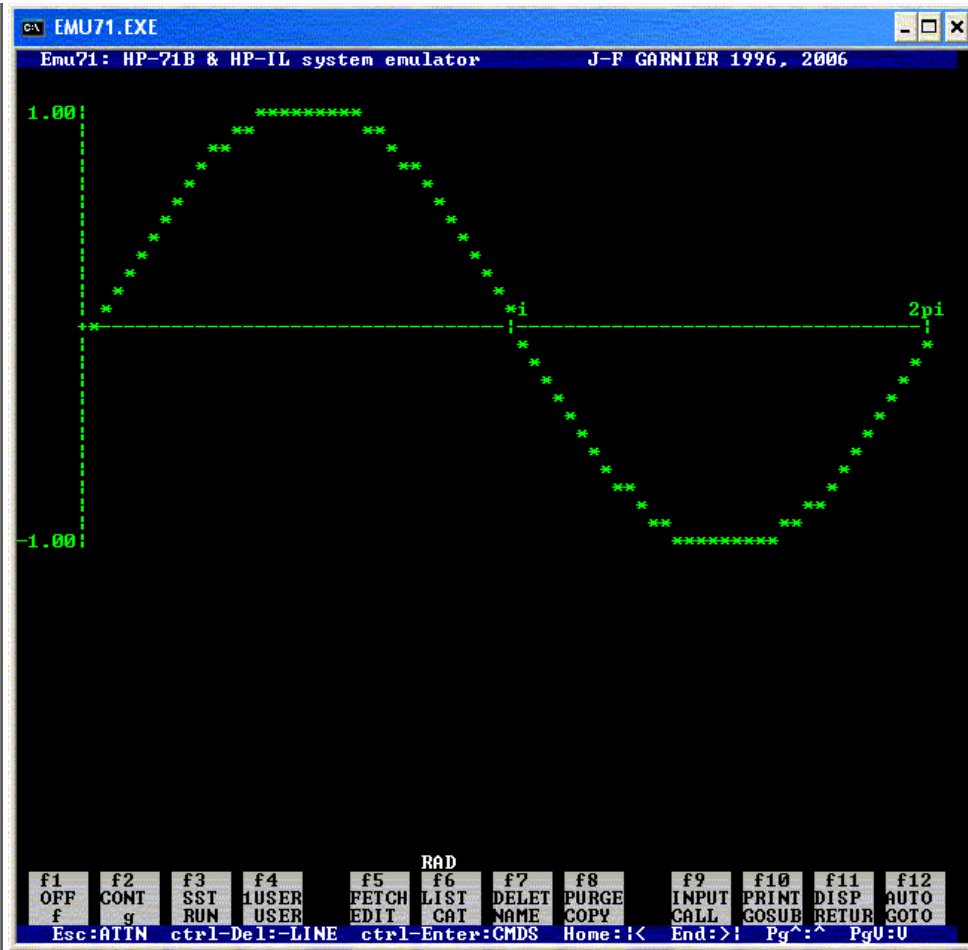
BTW, I updated my entry with a normal speed 15C. 9 hours, 40 minutes.

Edited: 20 June 2007, 12:46 p.m.

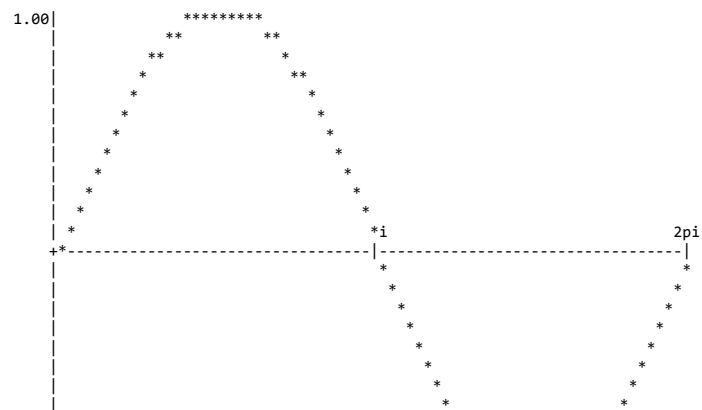
Re: Short & Sweet Math Challenge #19: Grade D, the movie

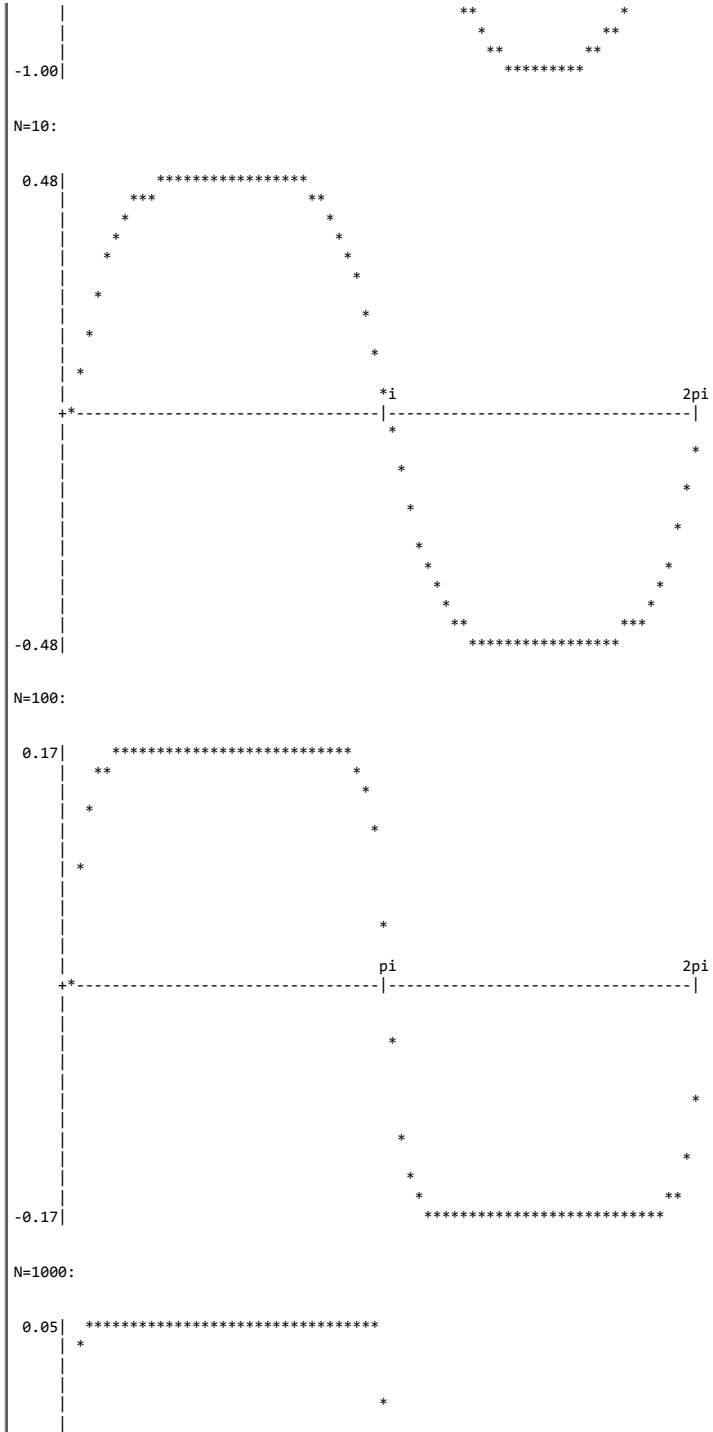
*Message #30 Posted by **Egan Ford** on 22 June 2007, 7:16 p.m.,
in response to message #1 by Valentin Albillo*

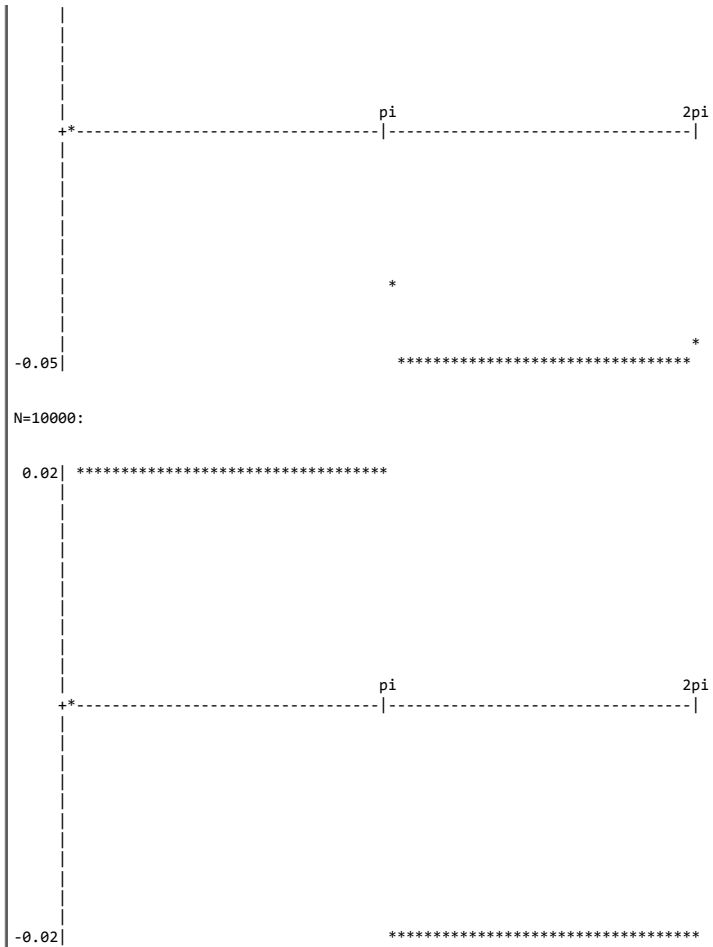
Here is my Grade D 71B BASIC submission. Since you already demonstrated a vertical plot, I opted for a horizontal plot. You'll need EMU71 or an 80 Column HP-IL video adapter.



N=1:







Clearly as N increases it is getting square as others have stated.

Code:

```

10 DESTROY ALL @ OPTION BASE 1 @ DELAY 0,0 @ RADIANS
20 P=2 @ L=25 @ C=77
30 FIX P
40 DIM L$(L)[C+1]
50 N=1
60 M=PI/2
70 FOR I=1 TO N
80 M=SIN(M)
90 NEXT I
100 CALL DRAWAXIS(L$( ),L,C,P,M)
110 FOR I=0 TO 2*PI STEP 2*PI/(C-P-4)
120 Y=I
130 FOR J=1 TO N
140 Y=SIN(Y)
150 NEXT J
160 CALL DRAWPT(L$( ),L,C,P,M,I,Y)

```

```

170 NEXT I
180 CALL PLOT(L$( ), L)
190 END
200 SUB DRAWAXIS(L$( ), L, C, P, M)
210 FOR I=1 TO L
220 L$(I)[P+4, P+4]=CHR$(124)
230 NEXT I
240 FOR I=P+4 TO C
250 L$(L/2)[I, I]="- "
260 NEXT I
270 L$(L/2)[P+4, P+4]="+"
280 L$(1)[1, LEN(STR$(-M))]=STR$(-M)
290 L$(L)[2, LEN(STR$(M))+1]=STR$(M)
300 L$(L/2)[(C-P+4)/2+2, (C-P+4)/2+2]=CHR$(124)
310 L$(L/2+1)[(C-P+4)/2+2, (C-P+4)/2+3]="pi"
320 L$(L/2)[C, C]=CHR$(124)
330 L$(L/2+1)[C-1, C+1]="2pi"
340 END SUB
350 SUB PLOT(L$( ), L)
360 FOR I=L TO 1 STEP -1
370 DISP L$(I)
380 NEXT I
390 END SUB
400 SUB DRAWPT(L$( ), L, C, P, M, X, Y)
410 V=Y/(2*M)*L+L/2+.5
420 H=X/(2*PI)*(C-P-4)+(P+4)+.5
430 IF V>L THEN V=L
440 IF V<1 THEN V=1
450 L$(V)[H, H]="*"
460 END SUB

```

Awesome ! :-)

Message #31 Posted by [Valentin Albillo](#) on 22 June 2007, 9:09 p.m.,
in response to message #30 by Egan Ford

Hi, Egan:

What can I say !? Awesome idea, this video of yours ! :-)

It nicely demonstrates in a vivid way just what's happening and it would certainly make an excellent material to show to a class.

I do have a similar, video-producing script for Mathematica showing the same ongoing process (only in 3D), but I didn't entertain the idea of doing likewise for Emu71, way to go !

Thanks for your continued interest in my challenges, kudos to your outstandingly clever contributions, and

Best regards from V.

Re: Short & Sweet Math Challenge #19: A--

Message #32 Posted by [Gerson W. Barbosa](#) on 22 June 2007, 11:07 p.m.,
in response to message #1 by Valentin Albillo

Hello Valentin,

This is no solution and actually deserves an F--. It seems to converge indeed, but very slowly. I have tried to compute the summation for decades 2 through 6, that is, for N=100 through N=1000000. I am not sure this eliminates all occurrences of digit 9. Anyway, this is not the way to go, as the HP-71B limits will be reached before the convergence is found. There has surely to be a clever way, but it is far beyond my very limited skills.

```

10 DESTROY ALL
30 FOR D=2 TO 6
40 N=10^D
50 S=LOG(N)+.577215664902
60 FOR I=1 TO N/10
70 S=S-1/(10*(I-1)+9)
80 NEXT I
90 FOR I=N/10+90 TO 9*N/10 STEP 100
100 FOR J=0 TO 8
110 S=S-1/(I+J)
120 NEXT J
130 NEXT I
140 FOR K=2 TO LGT(N)
150 FOR I=9*10^(K-1) TO 10^K-10 STEP 10
160 FOR J=0 TO 8
170 S=S-1/(I+J)
180 NEXT J
190 NEXT I
200 NEXT K
210 PRINT S
220 NEXT D

```

```

>RUN
4.77685709833
6.59022027355
8.54139178089
10.5158103624
12.4929996836

```

The results above are approximate. For exact results line 50 should be replaced, but this makes the program even slower:

```
50 S=0 @ FOR I=1 TO N @ S=S+1/I @ NEXT I
```

```

4.78184876505
6.59072019014
8.54144177977
10.5158153649

```

As always, I am looking forward to your and other people's smart solutions.

Best regards,

Gerson.

Re: Short & Sweet Math Challenge #19: A--

Message #33 Posted by [hugh steers](#) on 23 June 2007, 7:18 a.m.,
in response to message #32 by Gerson W. Barbosa

hi gerson, i too have an F- solution :-)

my first step was to write a simple, add up the terms without 9s program to see how it went and get a feel for any convergence. however, it converges very slowly indeed. in fact, it looks like it grows just like the harmonic series, ie slowly but actually. but presumably it does converge. the program below runs on the HP50 using hplua and after 10 million terms it's up to 13.65 and still growing fairly well.

i think i need a more cunning idea to get further.


```

tens = {1,10,100,1000,10000,100000,1000000,10000000,100000000}
nines = {9,90,900,9000,90000,900000,9000000,90000000,900000000}
function inc(n)
  local v
  local i
  local l = #n
  for i = 1,l do
    v = n[i] + tens[i]
    if v != nines[i] then
      n[i] = v
      for i = i + 1,l do
        v = v + n[i]
      end
      return v
    end
  end
  n[i] = 0
end
i = 1 + 1
v = tens[i]
n[i] = v
return v
end
function sum()
  local s = 1
  local n = { 1 }
  for i = 1,100 do
    for j = 1,100000 do
      s = s + 1/inc(n)
    end
    print(i, s)
  end
end
function go()
  sum()
end

```

Re: Short & Sweet Math Challenge #19: A--

Message #34 Posted by [Gerson W. Barbosa](#) on 23 June 2007, 9:55 a.m.,
in response to message #33 by hugh steers

Hi Hugh,

Looks like there's something wrong with my program. It's correct for N=100 and N=1000 but I am not sure about the rest. Perhaps some 9s have escaped. For N=10,000,000 I obtained 14.47; 13.65 seems to be more plausible. Since I am unable to find the solution, I thought of at least estimating the value the sum converges to by means of curve fitting:

x	y
2	4.78
3	6.59
7	13.65

I would need more data, though.

Regards,

Gerson.

Re: Short & Sweet Math Challenge #19: A--

Message #35 Posted by [hugh steers](#) on 23 June 2007, 11:49 a.m.,
in response to message #34 by Gerson W. Barbosa

oops, i missed a zero. the 13.65 is for 100 million terms, each main loop of the program is 1 million.

out of interest, here's a listing in 10s of millions of the total compared to the normal harmonic series' sum.

```

1 12.569954 16.695311
2 12.907506 17.388459
3 13.103535 17.793924
4 13.242197 18.081606
5 13.342811 18.304749
6 13.423368 18.487071
7 13.492294 18.641221
8 13.552527 18.774753
9 13.605026 18.892536
10 13.650922 18.997896
11 13.692797 19.093207
12 13.731294 19.180218
13 13.766824 19.260261
14 13.798917 19.334369
15 13.828992 19.403362
16 13.857288 19.467900
17 13.884003 19.528525
18 13.908823 19.585683
19 13.932289 19.639750
20 13.954658 19.691044
21 13.976028 19.739834
22 13.996292 19.786354
23 14.015530 19.830806
24 14.034024 19.873365
25 14.051829 19.914187
26 14.068947 19.953408
27 14.085249 19.991148
28 14.101013 20.027516
29 14.116275 20.062607
30 14.131064 20.096509
31 14.145235 20.129299
32 14.158972 20.161047
33 14.172325 20.191819
34 14.185315 20.221672
35 14.197874 20.250659
36 14.210045 20.278830
37 14.221914 20.306229
38 14.233496 20.332897
39 14.244486 20.358873
40 14.254336 20.384191
41 14.263987 20.408883
42 14.273447 20.432981
43 14.282724 20.456511
44 14.291746 20.479501
45 14.300597 20.501974
46 14.309288 20.523953
47 14.317823 20.545459
48 14.326163 20.566512
49 14.334337 20.587132
50 14.342373 20.607334
51 14.350277 20.627137
52 14.358032 20.646555
53 14.365625 20.665603
54 14.373099 20.684295
55 14.380458 20.702645
56 14.387703 20.720663

```

```

57 14.394792 20.738363
58 14.401777 20.755754
59 14.408661 20.772849
60 14.415448 20.789656
61 14.422108 20.806185
62 14.428664 20.822446
63 14.435132 20.838446
64 14.441513 20.854194
65 14.447794 20.869699
66 14.453971 20.884966
67 14.460070 20.900004
68 14.466091 20.914819
69 14.472034 20.929418
70 14.477873 20.943807
71 14.483642 20.957991
72 14.489342 20.971977
73 14.494975 20.985771
74 14.500518 20.999376
75 14.505991 21.012799
76 14.511403 21.026045
77 14.516754 21.039117
78 14.521896 21.052020
79 14.526845 21.064759
80 14.531743 21.077338
81 14.536591 21.089760
82 14.541386 21.102031
83 14.546116 21.114152
84 14.550799 21.126128
85 14.555437 21.137963
86 14.560031 21.149659
87 14.564562 21.161219
88 14.569049 21.172648
89 14.573494 21.183948
90 14.577898 21.195121
91 14.582250 21.206171
92 14.586556 21.217100
93 14.590823 21.227911
94 14.595053 21.238606
95 14.599240 21.249188
96 14.603379 21.259660
97 14.607482 21.270022
98 14.611551 21.280279
99 14.615585 21.290431
100 14.619570 21.300482

```

A-ish

Message #36 Posted by [PeterP](#) on 24 June 2007, 12:08 a.m.,
in response to message #35 by hugh steers

Hmm, I used a much shorter and much dumber code which takes about 13min for a sum of 10m terms. However it seems to be wrong as it gives a slightly different value than what you have posted. I was wondering though if your $n=10m$ means a) that you summed up 10m terms or b) summed up all terms that exclude 9 and are $\leq 10m$? Clearly a) would be a larger number than b).

My code is trying to compute b) and gives 12.2061531537 for $n=10m$ which is slightly smaller than your value. Can you spot the error in my code?

```

10 INPUT "n=?";N @ S=0 @ C=0 @ T$=TIME$
20 FOR X=1 TO N @ IF POS(STR$(X),"9")<>0 THEN 30
30   S=S+1/X
40 NEXT X
50 DISP "Done! "; "Sum=";S;"X=";X;"Start=";T$;" End=";TIME$

```

Clearly I have no clue about the final value. One way I was contemplating for a while but would have no chance of finishing without google (and even then it is doubtful) is to first calculate the value of $1/n$ for all n (maybe there is a good way to get the value of the rieman function for integers only, $s=1$ and up to a certain n in a smart way. There should be something like this around. And - wasn't Euler who originally came up with it in integer space??) and then subtract all values that have a 9 in it. But without a good idea to get $\sum(1/n)$ that is not very helpful.

The only good thing about my code (if it is correct) is that it can be easily adapted to A+ by changing the string we search for in line 20. So, for example, the value for $n=10000000$ excluding all numbers that have '42' in them is 16.2113166917. This is, not surprisingly, much closer to the simple $\sum(1/n)$ than when we exclude all numbers with a 9.

As an adendum and maybe it is useful for others my idea mentioned above came about when I calculated the number of terms one has to exclude from the simple $1/n$ sum with N increasing. This percentage is steadily increasing and seems to tend to 100% in the limit so the series does indeed seem to converge. yet to what number, I would not know. The first column is N . The second column is the 10% exclusion from having a 9 at the second place (e.g. 9000 for $n=10,000$) which excludes 10% of all terms(not necessarily 10% of the value though...) The third column is the discount from the previous max N times the 90% and the fourth column is the total discount. Maybe in an example I can better explain what I tried to calculate:

$N1=10$. We only ignore 1 number (9) which is 10%

$N2=100$. We ignore all number from 90-99, which is 10%. From the remaining 90% we ignore every one with 9 which is 10% (the final number from $N1$) for a total of $10\% + 90\%*10\% = 19\%$

$N3=1000$. We ignore all numbers from 900-999, which is 10%. from the remaining 90% we exclude 19% (see above from $N2$) for a total of $10\%+90\%*19\%=27\%$
and so on and so forth

maybe this is helpful to people smarter than me to come up with a clever code idea...

N	9 in 2nd place	prev disc	final disc
1.E+01	0.10	-	0.10
1.E+02	0.10	0.09	0.19
1.E+03	0.10	0.17	0.27
1.E+04	0.10	0.24	0.34
1.E+05	0.10	0.31	0.41
1.E+06	0.10	0.37	0.47
1.E+07	0.10	0.42	0.52
1.E+08	0.10	0.47	0.57
1.E+09	0.10	0.51	0.61
1.E+10	0.10	0.55	0.65
1.E+11	0.10	0.59	0.69
1.E+12	0.10	0.62	0.72
1.E+13	0.10	0.65	0.75
1.E+14	0.10	0.67	0.77
1.E+15	0.10	0.69	0.79
1.E+16	0.10	0.71	0.81
1.E+17	0.10	0.73	0.83
1.E+18	0.10	0.75	0.85
1.E+19	0.10	0.76	0.86
1.E+20	0.10	0.78	0.88

Cheers

Peter

Edited: 24 June 2007, 12:39 a.m.

Re: A-ish

Message #37 Posted by [hugh steers](#) on 24 June 2007, 7:37 a.m.,
in response to message #36 by PeterP

hi peter,

your code looks good. the discrepancy, as you pointed out, is that my code does, in fact, calculate (a) and yours calculates (b).

yes, i am summing 10M terms of 1/X without 9s and you are summing 1/X for the first 10M integers without 9s :-)

Re: A-ish

Message #38 Posted by [Gerson W. Barbosa](#) on 24 June 2007, 7:03 p.m.,
in response to message #36 by PeterP

Hello Peter,

Quote:

Hmm, I used a much shorter and much dumber code which takes about 13min for a sum of 10m terms.

13 minutes at what clock? On my jurassic computer your program takes 7 minutes for the first one million terms. But at least your program works and has proved helpful for an estimate of the answer (unlike mine, which was wrong for decades greater than 3).

Quote:

Clearly I have no clue about the final value.

Well, we can use data from your program to get an estimate of the result. The idea is getting the sum up to the first decades. I was intending to compute until the eighth decade (100,000,000), but no result past nearly two hours... I use your program to calculate the first six decades sums and used your result for the seventh decade. I put the data on a spread sheet and did an exponential fit of the differences between successive sums. Then I used the fit equation to calculate the differences and from those I calculated the next sums. As you can see, the differences tend to zero as the decades grow and the sum appears to converge to something close to 22.9. The correlation factor is excellent, but some additional data might be necessary to confirm this result. You can noticed I discarded the first two somes to get a better correlation factor.

From the table, it's evident the direct summation would take ages to complete. Even if we went up to 10^{100} , we would get four or five significant figures at most. So, there has to be another way...

Best regards,

Gerson.

D Sn Difference

(from Peter's program)

1	2.81785714286	
2	4.78184876505	1.96399162219
3	6.59072019014	1.80887142509
4	8.22318440268	1.63246421254
5	9.69287779213	1.46969338945
6	11.01565186020	1.32277406807
7	12.20615315370	1.19050129350

x (D)	y (Difference)
4	1.63246421254
5	1.46969338945
6	1.32277406807
7	1.19050129350

D Sn Difference

(from the exponential fit)

8	13.27779798383	1.07164483013
9	14.24238769904	0.96458971522

Exponential fit:

$$y = 2.4872351255 * \exp(-0.1052471262 * x)$$

10 15.11061688519 0.86822918615
 11 15.89211176098 0.78149487579
 12 16.59553690567 0.70342514469
 13 17.22869132486 0.63315441918
 14 17.79859491952 0.56990359466
 15 18.31156631696 0.51297139745
 16 18.77329292663 0.46172660966
 17 19.18889399743 0.41560107081
 18 19.56297737587 0.37408337843
 19 19.89969059402 0.33671321815
 20 20.20276685406 0.30307626004
 21 20.47556641894 0.27279956488
 22 20.72111386825 0.24554744931
 23 20.94213163229 0.22101776404
 24 21.14107017618 0.19893854389
 25 21.32013516861 0.17906499243
 26 21.48131193651 0.16117676790
 27 21.62638747672 0.14507554021
 28 21.75697026878 0.13058279206
 29 21.87450810844 0.11753783965
 30 21.98030415960 0.10579605117
 31 22.07553140281 0.09522724320
 32 22.16124564023 0.08571423742
 33 22.23839720159 0.07715156136
 34 22.30784148061 0.06944427902
 35 22.37034841896 0.06250693835
 36 22.42661104270 0.05626262374
 37 22.47725314602 0.05064210332
 38 22.52283620736 0.04558306133
 39 22.56386561459 0.04102940724
 40 22.60079626842 0.03693065383
 41 22.63403762587 0.03324135745
 42 22.66395824006 0.02992061419
 43 22.69088984642 0.02693160636
 44 22.71513104069 0.02424119427
 45 22.73695058949 0.02181954881
 46 22.75659041024 0.01963982075
 47 22.77426825328 0.01767784304
 48 22.79018011615 0.01591186287
 49 22.80450241664 0.01432230049
 50 22.81739394876 0.01289153212
 51 22.82899764333 0.01160369457
 52 22.83944215268 0.01044450934
 53 22.84884327702 0.00940112434
 54 22.85730524837 0.00846197135
 55 22.86492188618 0.00761663781
 56 22.87177763753 0.00685575136
 57 22.87794851342 0.00617087589
 58 22.88350293150 0.00555441808
 59 22.88850247464 0.00499954314
 60 22.89300257370 0.00450009906
 61 22.89705312211 0.00405054841
 62 22.90069902907 0.00364590696
 63 22.90398071742 0.00328168835
 64 22.90693457185 0.00295385443
 65 22.90959334230 0.00265877045
 66 22.91198650704 0.00239316475
 67 22.91414059955 0.00215409251
 68 22.91607950265 0.00193890310
 69 22.91782471332 0.00174521067
 70 22.91939558102 0.00157086771
 71 22.92080952228 0.00141394125
 72 22.92208221371 0.00127269143
 73 22.92322776588 0.00114555217

$$r^2 = 0.9999994812$$

74	22.92425887975	0.00103111387
75	22.92518698746	0.00092810771
76	22.92602237912	0.00083539166
77	22.92677431687	0.00075193775
78	22.92745113759	0.00067682072
79	22.92806034531	0.00060920772
80	22.92860869443	0.00054834912
81	22.92910226460	0.00049357017
82	22.92954652812	0.00044426353
83	22.92994641064	0.00039988251
84	22.93030634571	0.00035993507
85	22.93063032401	0.00032397830
86	22.93092193754	0.00029161353
87	22.93118441948	0.00026248194
88	22.93142068001	0.00023626053
89	22.93163333861	0.00021265859
90	22.93182475304	0.00019141444
91	22.93199704557	0.00017229253
92	22.93215212643	0.00015508086
93	22.93229171503	0.00013958860
94	22.93241735901	0.00012564399
95	22.93253045142	0.00011309241
96	22.93263224613	0.00010179471
97	22.93272387176	0.00009162563
98	22.93280634418	0.00008247242
99	22.93288057777	0.00007423359
100	22.93294739558	0.00006681781

Re: S&SMC#19: A-ish

Message #39 Posted by *Valentin Albillo* on 25 June 2007, 4:37 a.m.,
in response to message #38 by Gerson W. Barbosa

Hi, Gerson:

"[...] it's evident the direct summation would take ages to complete. Even if we went up to 10^{100} , we would get four or five significant figures at most."

Yes, it would take geological ages to get a few correct digits, and probably more than the whole life of the Universe to get 12 correct figures, which is what we must expect from 12-digit HP models.

Matter of fact, your last result from the exponential fit, 22.9329+, is about correct to just the four leftmost figures (22.93). Your fit overshoot the mark about $D=71$, which was closest to the actual sum even if still a little too high.

"So, there has to be another way..."

Indeed, there is. It is very easy to demonstrate that the sum of the harmonic series after taking out those terms including any given digit or sequence of digits, in any numerical base, always converges, if incredibly slowly.

But accurately finding the actual value of the sum to high precision is entirely another matter and the naive approach of simply adding up enough terms is doomed to fail. For this particular series, for instance, adding up all terms up to 10^{29} (which would takes ages even on a very fast computer, if at all possible) still gets short of the final sum by more than 1.

Thanks for your interest in this challenge of mine, and most specially for your hard work, keen ideas, and very worthy results.

Best regards from V.

Re: S&SMC#19: A-ish

Message #40 Posted by *Gerson W. Barbosa* on 25 June 2007, 10:05 a.m.,
in response to message #39 by Valentin Albillo

Hi Valentin,

Quote:

Yes, it would take geological ages to get a few correct digits, and probably more than the whole life of the Universe to get 12 correct figures, which is what we must expect from 12-digit HP models.

Yes, it appears we'd have to go as far as 10^{280} to get 12 digits. How long would take this to be done on the HP-71B (or even on Emu71) and how much energy would be required for this task ? :-)

Quote:

For this particular series, for instance, adding up all terms up to 10^{29} (which would takes ages even on a very fast computer, if at all possible) still gets short of the final sum by more than 1.

You're right! And even if we summed every term up to 10^{50} we'd be still one tenth of a unit away from the answer, as we can see in the table.

Thanks for your encouraging words, but I think this was just an engineering approach to get an estimation of the result. I hope someone comes up with a really beautiful mathematical way of tackling the problem before you post the solution.

Best regards,

Gerson.

Re: S&SMC#19: A-ish

Message #41 Posted by *hugh steers* on 25 June 2007, 6:06 p.m.,
in response to message #40 by Gerson W. Barbosa

pathetically lame googlers present.... Fischer's algorithm for the harmonic 9's

hplua on the hp50g in 11 seconds, transcript:

```
hplua -i valentina2.lua
Lua 5.1.2 Copyright (C) 1994-2007 Lua.org, PUC-Rio
HPLua version 0.4
```

```
=harmonic9(23)
> 22.92067661926415034816367
>
```

listing:

```
function ncr(n, r)
  -- combinations nCr
  local v = 1
  local rr = 1
  while r > 0 do
    v = v * n
    n = n - 1
    rr = rr * r
    r = r - 1
  end
  return v / rr
end
```



```

    rr = rr * r
    r = r - 1
    n = n - 1
end
return v/rr
end

function nextBeta(bl)
    local n = #bl + 1
    local k;
    local tn = 10^n;
    local t = ((11^n) - tn)*10
    for k = 2,n do
        local a = (tn*(10^(1-k)) - (10^k) + 1)*bl[n-k+1]
        t = t - a * ncr(n,k);
    end
    bl[n] = t / (tn - 9)/n;
    return bl[n];
end

function zetaConstant(n, eps)
    -- riemann zeta function for integers 2...
    local s = 1
    local t = 2;
    local s1;

    if n == 2 then
        t = math.pi;
        s = t*t/6;
    elseif n == 4 then
        t = math.pi;
        t = t*t;
        s = t*t/90;
    elseif n == 3 then
        -- apery's constant
        s = 1.2020569031595942853997382
    else
        repeat
            s1 = s
            s = s + 1/(t^n)
            t = t + 1;
        until (math.abs(s - s1) < eps)
    end
    return s;
end

function harmonic9(n)
    -- method of Fischer
    local v = 10*math.ln(10)
    local bl = {}

    -- b0
    nextBeta(bl)
    for i = 2,n do
        -- try for 20 digits with eps = 10^-22
        v = v - (10^(-i))*nextBeta(bl)*zetaConstant(i,1e-22);
    end
    return v;
end
end

```

i would prefer a nice way to calculate Apery's constant. ie zeta(3), but it didnt converge quick enough, so i settled for the ugly constant

Re: S&SMC#19: A-ish

Message #42 Posted by [Valentin Albillo](#) on 25 June 2007, 8:05 p.m.,
in response to message #41 by hugh steers

Hi, Hugh:

Hugh posted:

"pathetically lame googlers present...."

That's what you say!

"I would prefer a nice way to calculate Apery's constant. ie zeta(3), but it didnt converge quick enough, so i settled for the ugly constant"

You can get rid of it by the usual trick of improving convergence by subtracting the slowly-convergent part and applying an exponentially fast-converging series to what's left. Tootoba, try an RPL-version of this one-liner:

```
10 SUB APERY(A) @ A=7*PI^3/180 @ FOR K=1 TO 3 @ A=A-2/K^3/(EXP(2*PI*K)-1) @ NEXT K
```

```
>CALL APERY(A) @ PRINT A
```

```
1.20205690316
```

where you can increase the upper limit (3) to achieve whatever numeric precision you want in negligible time. The given value, 3, is enough to get you 12 correct digits in just 3 iterations.

Thanks for your interest and

Best regards from V.

Re: S&SMC#19: A-ish

Message #43 Posted by [hugh steers](#) on 26 June 2007, 4:13 a.m.,
in response to message #42 by Valentin Albillo

nice!, 7 seems to be enough,

```
> =apery(7)
1.202056903159594285399739
```

```
function apery(n)
  local a = 7*(math.pi^3)/180
  for k = 1,n do
    a = a - 2/(k^3)/(math.exp(2*math.pi*k)-1)
  end
  return a;
end
```

Re: S&SMC#19: A-ish

Message #44 Posted by [Egan Ford](#) on 25 June 2007, 8:52 p.m.,
in response to message #41 by hugh steers

Quote:

Fischer's algorithm for the harmonic 9's

The cat is out of the bag now. I assume you got this from the same paper that I have been using to work on A+.

Re: S&SMC#19: A-ish

Message #45 Posted by [PeterP](#) on 26 June 2007, 2:11 a.m.,
in response to message #44 by Egan Ford

Egan, Gerson et all,

Are you guys talking about [this paper](#)? I was just following your threads (having decided that without google I will not be able to do anything further. And in my little black book I already got further than I could have hoped for my first 71b challenge) when I saw you mentioning the 'fisher paper'. Could not find that one but the paper linked above has a good resemblance with the challenge we were faced, missing '9s', '42' and 314159 and all.

Please let me know if you have identified a better source to understand the challenge. Quite honestly, given the wide area of mathematicians that according to the paper have gnawed at this and only very recently, I did not feel too bad to not have come up with any idea...

The thing that I enjoy the most about this challenges is the tid-bits of mathematics that Valentin and you others allow me to learn. And in such a nice interactive environment. Really cool! Can't wait for Valentin to lift the veil even further for me!

Cheers

Peter

PS:Gerson - thanks for the tip on the Euler constant book - it has already made its way into an Amazon shopping basket :-)

Re: S&SMC#19: A-ish

Message #46 Posted by [Egan Ford](#) on 26 June 2007, 2:27 a.m.,
in response to message #45 by PeterP

Yes.

Re: S&SMC#19: A-ish

Message #47 Posted by [hugh steers](#) on 26 June 2007, 4:07 a.m.,
in response to message #46 by Egan Ford

yes, same here. the Fischer's part was the first and easier bit. i had got stuck on my own.

Re: S&SMC#19: A-ish

Message #48 Posted by [Gerson W. Barbosa](#) on 28 June 2007, 9:23 a.m.,
in response to message #41 by hugh steers

Hello Hugh,

I tried to port your code to RPL on my HP-28S, just to have an idea of the running time on a vintage calculator, but I got stuck at this part:

```
local b1 = {}
-- b0
nextBeta(b1)
```

Perhaps I ought to learn a little *Lua* first. So far the only thing I know it is the Portuguese word for Moon (Spanish & Italian: *Luna*, French: *Lune*) :-)

As of the apery or zeta3 constant, the algorithm provided by Valentin in fact requires only three terms for 12 digits. But in this case I'd simply type in the constant.

Congratulations for your nice work!

Gerson.

Re: S&SMC#19: A-ish

Message #49 Posted by [hugh steers](#) on 28 June 2007, 9:42 a.m.,
in response to message #48 by Gerson W. Barbosa

hi gerson,

lua has untyped variables. `foo = {}`, assigns the empty list to `foo`. you have to have a list to start inserting, otherwise `foo` is null.

some examples:

```
-- this is a comment
foo = {}
foo[1] = 12
foo[2] = "hello"
foo[3] = {}
```

```
moo = #foo -- gets length which is 3.
```

```
foo[#foo + 1] = 7 -- add to end, lists expand automatically
```

```
function addSomething(list)
-- lists passed by reference, we can modify it
list[#list + 1] = 42
end
```

```
addSomething(foo)
```

annoyingly lua lists start from 1 not zero. this is my biggest lua pet hate. so much that im thinking of adding a base mode (like 71b basic) for hplua.

so, the following, sets `b1` to be the empty list and the `nextBeta` function computes the next beta and adds it to the list.

```
local b1 = {} -- b0 nextBeta(b1)
```

hope this helps,

Edited: 28 June 2007, 9:43 a.m.

71b version

Message #50 Posted by [hugh steers](#) on 28 June 2007, 9:52 a.m.,
in response to message #49 by hugh steers

what i'd suggest for the 71b is to simply have a pre-sized array for the beta values because, for a given precision, you know how many you need.

in 71b basic, i'd just have a global array b[] and nextBeta filled in the next slot using the preceeding values.

Re: S&SMC#19: A-ish

Message #51 Posted by [Gerson W. Barbosa](#) on 28 June 2007, 1:20 p.m.,
in response to message #49 by hugh steers

Hi Hugh,

Thanks for the tips. I had my HP-28S fixed [here](#). Since I am in the city I took the time to go there again this morning because of a recurring problem in the keyboard. Meanwhile, I will use Power48 on the PalmTX. If I succeed, I will let you know when I return.

Re: Short & Sweet Math Challenge #19: A--

Message #52 Posted by [PeterP](#) on 25 June 2007, 6:10 p.m.,
in response to message #32 by Gerson W. Barbosa

intrigued by the 0.57... number I went to google to find out what it means. Ahh! (light bulb going off) There is indeed a fast way to calculate/estimate $s(1/n)$ as I had thought and you have used that way. Thanks for showing me, Gerson!

Cheers

Peter PS: BTW - my laptop is not that powerful - 1.7ghz Pentium M I believe... Not sure why it would run longer on yours...

Re: Short & Sweet Math Challenge #19: A--

Message #53 Posted by [Gerson W. Barbosa](#) on 25 June 2007, 10:15 p.m.,
in response to message #52 by PeterP

Hi Peter,

Quote:

intrigued by the 0.57... number I went to google to find out what it means.

If you google for *site:hpmuseum.org EulerGamma* you'll find this has been discussed here recently. Just a few months ago I wasn't aware of it either (not that I have become an expert, but at least now I can recognize it when I see it :-)

You might find this book interesting:

[Havil, J. Gamma: Exploring Euler's Constant](#)

I haven't read it yet. The introduction in pdf format is available there. The rest seems to be worth reading.

Quote:

There is indeed a fast way to calculate/estimate $s(1/n)$ as I had thought and you have used that way.

I guess you hadn't seen Hugh Steer's post (*on 25 June 2007, 6:06 p.m*) when you wrote this :-)

Best regards,

Gerson.

Re: Short & Sweet Math Challenge #19: A--

Message #54 Posted by [Paul Guertin](#) on 26 June 2007, 6:55 a.m.,
in response to message #53 by Gerson W. Barbosa

Quote:

You might find this book interesting:

Havil, J. Gamma: Exploring Euler's Constant

I haven't read it yet. The introduction in pdf format is available there. The rest seems to be worth reading.

Definitely recommended. I read it when it came out, and IMHO it's one of the best college-level pop math books. In the same style, and also worth checking out, is Paul Nahin's *Dr. Euler's Fabulous Formula* about Fourier series and more.

Re: Short & Sweet Math Challenge #19: A--

Message #55 Posted by [Gerson W. Barbosa](#) on 26 June 2007, 7:21 a.m.,
in response to message #54 by Paul Guertin

Thanks for the book recommendation, Paul. It's been quite a long time since the last time I ordered something from Amazon. Now that the Brazilian Real to American Dollar exchange rate is below 2 (1 USD = 1.94 BRL), after having gone as high as 3.3 just a couple of years ago, it's time to go shopping :-)

Peter, for the same reason, I'm going to get a 2GHz+ notebook. They have never been so cheap here. I will only have to choose between HP and Dell. My desktop Pentium III 500Mhz is getting tired of running all those nice emulators and simulators :-)

Gerson.

Edited: 26 June 2007, 7:27 a.m.

Re: Short & Sweet Math Challenge #19: A--

Message #56 Posted by [PeterP](#) on 26 June 2007, 10:56 a.m.,
in response to message #55 by Gerson W. Barbosa

Gerson,

IMHO I would avoid Dell. There has been a steady decline in customer support with Dell computers (there was a big blog story recently where a very disappointed customer started blogging his interactions with Dell - it got nicknamed Dell Hell and he became quite famous and Dell under a lot of pressure). I also do not like their keyboards (nothing beats the IBM/Lenovo keyboards for me, but that's just me and keyboard feel is something very personal).

For me, good sources of rankings and research can be found [here at PC World](#) and [here at PC Magazine](#) . My favorite place to buy, but I do not know if they ship to Brazil is [Newegg](#) where you can also find lots of customer reviews.

Thanks for all your tips, I always enjoy reading your posts here.

Cheers

Peter

Re: Short & Sweet Math Challenge #19: A--

Message #57 Posted by **Gerson W. Barbosa** on 28 June 2007, 12:57 p.m.,
in response to message #56 by PeterP

Hello Peter,

Thanks for your advice. Because of the Brazilian keyboard (the one with ç - 'c' with a cedilla mark beneath - and the signs ´ ` ~ ^ in the proper position) I will buy locally. I'll take a look at the reviews in the magazines you have suggest though.

Best regards,

Gerson

Re: Short & Sweet Math Challenge #19: A--

Message #58 Posted by **PeterP** on 26 June 2007, 10:58 a.m.,
in response to message #54 by Paul Guertin

Thanks Paul, appreciate the info!

Cheers

Peter

Re: Short & Sweet Math Challenge #19: Surprise ! [Edited with results]

Message #59 Posted by **Arnaud Amiel** on 25 June 2007, 8:24 a.m.,
in response to message #1 by Valentin Albillo

I didn't have much time as I am just moving into a new flat but I recognised that Grade F could be solved by a hp55. I had been waiting for along time for a SSMC that could be handled by the 55 so here it is:

This program takes the number of iterations as argument:

```

      2
      -
      STO 5
      0
      STO 1
      1
      STO 2
      STO 4
13    RCL 1
      RCL 4
      /
      RCL 2
      STO 1
      +
      STO 2
      RCL 4
      1
      +

```

```

      STO 4
      RCL 5
      X=Y 36
      GTO 13
36    2
      +
      RCL 2
      /
      GTO 00

```

For N=13, it takes about 20s to approximate e with a value of 2.944805312.

I knew I could not get N=13000 in a reasonable time so for the other serie I used SysRPL on my hp49g+

```

!NO CODE
!RPL
::
  0LastRomWrd!
  CK1NOLASTWD
  CKREAL
  1LAMBIND
  %0
  %1
  1GETLAM
  COERCE
  #2-
  ONE
  DO
  DUPUNROT
  INDEX@
  UNCOERCE
  %/
  %+
  LOOP
  SWAPDROP
  %SQ_
  1GETABND
  %2
  %X
  SWAP
  %/
;

```

This comes close to Pi (3.14219691996) in about 50s for N=13000. A very similar program gives 2.71849095023 for the first serie (N=13000).

Unfortunately I had no time for the other challenges for now. I will look later.

Arnaud

Edited: 26 June 2007, 9:33 a.m. after one or more responses were posted

Re: Short & Sweet Math Challenge #19: Surprise ! [LONG]

Message #60 Posted by [Valentin Albillo](#) on 25 June 2007, 9:04 a.m.,
in response to message #59 by Arnaud Amiel

Thanks, Arnaud !

Would you please edit your post to include the actual numbers you got as approximations to Pi and E ?

Best regards from V.

Re: Short & Sweet Math Challenge #19: A+ for 71B

Message #61 Posted by **Egan Ford** on 27 June 2007, 3:07 p.m.,
in response to message #1 by Valentin Albillo

Quote:

Note: Get an A+ !

Compute the limit value of the infinite sum excluding instead those terms which include the digit string '42', as in "HP42S" (i.e: **42**, 780**42**998, **142**883213044, etc). Also, see if you can *predict the approximate value* of the infinite sum when excluding terms which do include some given arbitrary non-periodic string of digits (say, **314**, or **314159**, or your phone number, etc)

Unlike A I do not see a requirement for *full 12-digit accuracy*. I can provide 10/11 digits however for any *given arbitrary non-periodic string of digits*. Well almost any, there is a memory/speed trade off. I.e. more digits = more memory = more speed. Each digit consumes ~1800 bytes of memory. You'll need extra memory in your 71B for strings > 8 digits.

I lost a lot of development time due to issues with DEF FN and scope. After switching to CALL SUB my mystery problems went away. This 164 line/2792 byte program is far from optimal or clean, and it is littered with hacks in the interest of my time.

The algorithm is based solely on <http://users.ox.ac.uk/~ball2009/SchmelzerBaillie.pdf>. Accuracy is approx 2/3 of machine accuracy. I.e. 10 of 15 digits on a 71B.

Accuracy:

N	71B	Actual	Difference	Rel Diff
9	22.9206766194	22.9206766193	0.0000000001	4.36E-12
42	228.446304138	228.446304159	0.0000000021	9.19E-12
314	2299.82978273	2299.82978278	0.0000000005	2.17E-13
314159	2302582.33377	2302582.33386	0.0000000009	3.91E-16
phone	23025850932.6	23025850929.6	3.0000000000	1.30E-10

If you increase D0 (line 30) the output will be more verbose. Example output with D0=1 (unless specified all output is from EMU71):

DEPLETE: 9

```
T = |  | 0 1 2 3 4 5 6 7 8 9 |
    |-----|
    | 1 | 1 1 1 1 1 1 1 1 1 0 |
```

Z(9)= 22.9206766194 IN 4.1 SEC USING 1883 BYTES

DEPLETE: 9

```
T = |  | 0 1 2 3 4 5 6 7 8 9 |
    |-----|
    | 1 | 1 1 1 1 1 1 1 1 1 0 |
```

A = | .9 |

Z(9)= 22.9206766194 IN 1262.74 SEC USING 1886 BYTES (71B, D0=2)

DEPLETE: 42

```
T = |  | 0 1 2 3 4 5 6 7 8 9 |
    |-----|
```

```

  | 1 | 1 1 1 1 2 1 1 1 1 1 |
  | 2 | 1 1 0 1 2 1 1 1 1 1 |
  +-----+

```

Z(42)= 228.446304138 IN 11.6 SEC USING 3600 BYTES

DEplete: 42

```

  | 0 1 2 3 4 5 6 7 8 9 |
  +-----+
  | 1 | 1 1 1 1 2 1 1 1 1 1 |
  | 2 | 1 1 0 1 2 1 1 1 1 1 |
  +-----+

```

A = | .9 .8 |
 | .1 .1 |

Z(42)= 228.446304138 IN 4451.03 SEC USING 3598 BYTES (71B, D0=2)

DEplete: 314

```

  | 0 1 2 3 4 5 6 7 8 9 |
  +-----+
  | 1 | 1 1 1 2 1 1 1 1 1 1 |
  | 2 | 1 3 1 2 1 1 1 1 1 1 |
  | 3 | 1 1 1 2 0 1 1 1 1 1 |
  +-----+

```

Z(314)= 2299.82978273 IN 27.78 SEC USING 5350 BYTES

DEplete: 314159

```

  | 0 1 2 3 4 5 6 7 8 9 |
  +-----+
  | 1 | 1 1 1 2 1 1 1 1 1 1 |
  | 2 | 1 3 1 2 1 1 1 1 1 1 |
  | 3 | 1 1 1 2 4 1 1 1 1 1 |
  | 4 | 1 5 1 2 1 1 1 1 1 1 |
  | 5 | 1 1 1 2 1 6 1 1 1 1 |
  | 6 | 1 1 1 2 1 1 1 1 1 0 |
  +-----+

```

A = | .9 .8 .8 .8 .8 .8 | <= added with D0=2
 | .1 .1 .1 .1 .1 .1 |
 | 0 .1 0 0 0 0 |
 | 0 0 .1 0 0 0 |
 | 0 0 0 .1 0 0 |
 | 0 0 0 0 .1 0 |

Z(314159)= 2302582.33377 IN 222.31 SEC USING 10798 BYTES

DEplete: phone

```

  | 0 1 2 3 4 5 6 7 8 9 |
  +-----+
  | 1 | 1 1 1 1 1 1 1 1 2 1 |
  | 2 | 3 1 1 1 1 1 1 1 2 1 |
  | 3 | 1 4 1 1 1 1 1 1 2 1 |
  | 4 | 1 1 1 1 1 1 1 1 2 5 |
  | 5 | 1 1 1 1 1 1 1 6 2 1 |
  | 6 | 1 7 1 1 1 1 1 1 2 1 |
  | 7 | 1 1 1 1 1 1 1 1 2 8 |
  | 8 | 1 1 1 1 1 1 1 1 2 9 |
  | 9 | 1 10 1 1 1 1 1 1 2 1 |
  |10 | 0 1 1 1 1 1 1 1 2 1 |
  +-----+

```

Z(phone)= 23025850932.6 IN 729.86 SEC USING 18507 BYTES

Code (Update):

```

10 DESTROY ALL @ STD @ OPTION BASE 0
20 T0=TIME @ M0=MEM
30 D0=2
40 INPUT "DELETE: ";N$
50 B0=10
60 P0=20
70 INTEGER T(LEN(N$)-1,10)
80 REAL A(LEN(N$)-1,LEN(N$)-1)
90 REAL B(LEN(N$)-1,LEN(N$)-1)
100 CALL T(T(,),N$)
110 IF D0>0 THEN DISP @ CALL PT(T(,),N$) @ DISP
120 CALL A(A(,),N$)
130 IF D0>1 THEN CALL PA(A(,),N$) @ DISP
140 CALL B(B(,),A(,),N$)
150 DIM I0$(90*LEN(N$))[8]
160 REAL I0(90*LEN(N$))
170 C0=0
180 FOR I=1 TO P0+1
190 FOR J=1 TO LEN(N$)
200 CALL PSI(I,J,1,N$,T(,),P0,I0(),I0$(C0),C0,R)
210 Z=Z+R
220 IF D0>2 THEN DISP "R[";I;";";J;";1]=";R
230 IF D0>2 THEN DISP "Z[";I;";";J;";1]=";Z
240 NEXT J @ NEXT I
250 REAL Q(LEN(N$)-1)
260 REAL U(LEN(N$)-1)
270 FOR J=1 TO LEN(N$)
280 CALL PSI(P0+1,J,1,N$,T(,),P0,I0(),I0$(C0),C0,R)
290 Q(J-1)=R
300 NEXT J
310 MAT U=B*Q
320 Z=Z+CNORM(U)
330 DISP "Z(";N$;";")=";Z;"IN";TIME-T0;"SEC USING";M0-MEM;"BYTES"
340 DISP "L(";N$;";")=";B0^LEN(N$)*LOG(B0);"L(N)=";STR$(B0);"^LEN(N)*LOG(";STR$(B0);")"
350 END
360 SUB S(I,J,K,T(,),N$,X,A,H)
370 H=0
380 IF I>1 THEN 430
390 FOR D=1 TO 9
400 IF T(0,D)=J THEN H=H+1/(X*D+A)^K
410 NEXT D
420 GOTO 500
430 FOR L1=1 TO LEN(N$)
440 FOR M1=1 TO 10
450 IF T(L1-1,M1-1)<>J THEN 480
460 CALL S(I-1,L1,K,T(,),N$,10*X,M1-1+A,H1)
470 H=H+H1
480 NEXT M1
490 NEXT L1
500 !
510 END SUB
520 SUB PSI(I,J,K,N$,T(,),P0,I0(),I0$(C0),C0,R)
530 I$=STR$(I)&" "&STR$(J)&" "&STR$(K)
540 FOR J0=0 TO C0
550 IF I0$(J0)=I$ THEN R=I0(J0) @ GOTO 880
560 NEXT J0
570 DEF FNC(K,W)
580 FNC=(-1)^W*(FACT(K+W-1)/(FACT(W)*FACT(K-1)))

```

```

590 END DEF
600 DEF FNA(K,W,M)
610 IF M=0 AND W=0 THEN X=1 ELSE X=M^W
620 FNA=10^(-K-W)*FNC(K,W)*X
630 END DEF
640 DEF FNW(I,K)
650 FNW=MAX(FLOOR((P0+1)/(I-1)+1-K)+1,0)
660 END DEF
670 ! IF I>3 THEN 710
680 IF I>2 THEN 720
690 CALL S(I,J,K,T(,),N$,1,0,H)
700 Z1=H
710 GOTO 830
720 Z1=0
730 FOR W=0 TO FNW(I,K)
740 FOR L=1 TO LEN(N$)
750 CALL PSI(I-1,L,K+W,N$,T(,),P0,I0(),I0$,C0,Z0)
760 ! DISP Z0
770 FOR M=0 TO 9
780 ! DISP FNA(K,W,M)
790 IF T(L-1,M)=J THEN Z1=Z1+FNA(K,W,M)*Z0
800 NEXT M
810 NEXT L
820 NEXT W
830 R=Z1
840 I0$(C0)=I$
850 I0(C0)=R
860 C0=C0+1
870 ! DISP "[";I;" ";J;" ";K;"]=";Z1
880 END SUB
890 SUB T(T(,),N$)
900 IF LEN(N$)>1 THEN 960
910 FOR C=0 TO 9
920 T(0,C)=1
930 IF VAL(N$)=C THEN T(0,C)=0
940 NEXT C
950 GOTO 1240
960 ! DO MULTI DIGIT
970 DIM S$(LEN(N$)-1)[80]
980 S$(0)=""
990 FOR I=1 TO LEN(N$)-1
1000 S$(I)=N$[1,I]
1010 NEXT I
1020 FOR R=0 TO LEN(N$)-1
1030 FOR C=0 TO 9
1040 T(R,C)=1
1050 NEXT C
1060 NEXT R
1070 FOR I=0 TO LEN(N$)-1
1080 S2$=S$(I)
1090 FOR J=0 TO 9
1100 A$=S2$&&STR$(J)
1110 IF N$=STR$(J) THEN T(I,J)=0 @ GOTO 1220
1120 IF A$=N$ THEN T(I,J)=0 @ GOTO 1220
1130 IF POS(A$,N$)>0 THEN T(I,J)=0 @ GOTO 1220
1140 L=0 @ M=0 @ F=0
1150 FOR K=1 TO LEN(N$)-1
1160 IF LEN(A$)<LEN(S$(K)) THEN 1210
1170 IF A$[LEN(A$)-LEN(S$(K))+1,LEN(A$)]<>S$(K) THEN 1210
1180 F=1
1190 IF LEN(S$(K))>M THEN M=LEN(S$(K)) @ L=K+1
1200 IF F=1 THEN T(I,J)=L
1210 NEXT K
1220 NEXT J
1220 NEXT I

```

```

1230 NEXT I
1240 END SUB
1250 SUB PT(T(,),N$)
1260 DIM A$(80)
1270 V$=CHR$(124)
1280 DISP " .- "
1290 DISP " "&V$&" "&V$&" 0 1 2 3 4 5 6 7 8 9 "&V$&"
1300 DISP "T = "&V$&" ----- "&V$&"
1310 FOR R=0 TO LEN(N$)-1
1320 IF LEN(STR$(R+1))=1 THEN E$=" " ELSE E$=" "
1330 A$=" "&V$&" "&E$&STR$(R+1)&" "&V$&" "
1340 FOR C=0 TO 9
1350 IF LEN(STR$(T(R,C)))=1 THEN E$=" " ELSE E$=" "
1360 A$=A$&E$&STR$(T(R,C))&" "
1370 NEXT C
1380 DISP A$&CHR$(124)
1390 NEXT R
1400 DISP " "&CHR$(96)&"- "
1410 END SUB
1420 SUB A(A(,),N$)
1430 FOR R=0 TO LEN(N$)-1
1440 FOR C=0 TO LEN(N$)-1
1450 IF R=0 THEN A(R,C)=.8
1460 IF R=1 THEN A(R,C)=.1
1470 IF R=0 AND C=0 THEN A(R,C)=.9
1480 IF R>1 AND C=R-1 THEN A(R,C)=.1
1490 NEXT C
1500 NEXT R
1510 END SUB
1520 SUB PA(A(,),N$)
1530 DIM A$(80)
1540 FOR R=0 TO LEN(N$)-1
1550 IF R=0 THEN A$="A =" ELSE A$=" "
1560 A$=A$&" "&CHR$(124)&" "
1570 FOR C=0 TO LEN(N$)-1
1580 IF A(R,C)=0 THEN A$=A$&" "
1590 A$=A$&STR$(A(R,C))&" "
1600 NEXT C
1610 A$=A$&CHR$(124)
1620 DISP A$
1630 NEXT R
1640 END SUB
1650 SUB B(B(,),A(,),N$)
1660 REAL I(LEN(N$)-1,LEN(N$)-1)
1670 MAT I=IDN
1680 MAT B=I-A
1690 MAT B=INV(B)
1700 MAT B=B-I
1710 END SUB

```

Edited: 28 June 2007, 11:43 a.m. after one or more responses were posted

Re: Short & Sweet Math Challenge #19: A+ for 71B

Message #62 Posted by [Dave Shaffer \(Arizona\)](#) on 27 June 2007, 7:23 p.m.,
in response to message #61 by Egan Ford

Note that your largest value (23025850929.6 for the phone number) appears to be a power-of-10 multiple of the value of $\ln(10)$. My '41CX gives $\ln(10) = 2.302585093$ and QuattroPro gives 2.302585092994 .

It appears that the longer the sequence of omitted numbers, the closer the sum gets to a multiplied value of $\ln(10)$. In fact, it looks like if you omit a sequence of numbers that is n digits long, the result approaches 10^n times $\ln(10)$. Incredibly fascinating, and for which I have NO explanation!

What happens if you try this in a different base number system (such as octal, hexadecimal, ...)?

Re: Short & Sweet Math Challenge #19: A+ for 71B

Message #63 Posted by [Egan Ford](#) on 27 June 2007, 7:42 p.m.,
in response to message #62 by [Dave Shaffer \(Arizona\)](#)

Quote:

It appears that the longer the sequence of omitted numbers, the closer the sum gets to a multiplied value of $\ln(10)$. In fact, it looks like if you omit a sequence of numbers that is n digits long, the result approaches 10^n times $\ln(10)$. Incredibly fascinating, and for which I have NO explanation!

That's the 2nd part of A+, I forgot to point that out.

Read the PDF link above section 6 for an explanation.

Edited: 27 June 2007, 8:05 p.m.

Re: Short & Sweet Math Challenge #19: A+ for 71B

Message #64 Posted by [Egan Ford](#) on 27 June 2007, 8:02 p.m.,
in response to message #62 by [Dave Shaffer \(Arizona\)](#)

Quote:

What happens if you try this in a different base number system (such as octal, hexadecimal, ...)?

Same, e.g.:

$$\begin{aligned} z(123456(\text{base}8)) &= 545107.239019 \\ 8^6 * \ln(8) &= 545113.123502 \end{aligned}$$

$$\begin{aligned} z(123456654321(\text{base}8)) &= 142898134658 \\ 8^{12} * \ln(8) &= 142898134647 \end{aligned}$$

Re: Short & Sweet Math Challenge #19: A+ for 71B

Message #65 Posted by [hugh steers](#) on 28 June 2007, 6:21 a.m.,
in response to message #61 by [Egan Ford](#)

Hi Egan,

Excellent stuff! i didnt have the time to go on and implement the full general algorithm, but im impressed with your 71b version.

so valentin, do you claim to also have a version of this for the 71b or do you have a completely different method?

Re: Short & Sweet Math Challenge #19: A+ for 71B

Message #66 Posted by [Egan Ford](#) on 28 June 2007, 10:52 a.m.,
in response to message #65 by [hugh steers](#)

Thanks. It took a lot longer than I expected. My BASIC/71B skills are weak, in hindsight I should have used UserRPL or Lua.

I am looking forward to Valentin's solution. I am sure it will be properly structured, efficient, and easier to read.

Re: Short & Sweet Math Challenge #19: A+ for 71B

Message #67 Posted by [Valentin Albillo](#) on 29 June 2007, 4:26 a.m.,
in response to message #66 by Egan Ford

Hi, Egan:

Egan posted:

"I am looking forward to Valentin's solution. I am sure it will be properly structured, efficient, and easier to read."

Thanks for your highly flattering confidence in my skills but, self-quoting from my original mail (#1) where I proposed the challenge:

"I'll provide my original solutions plus extensive comments to all graded problems A-F (but not for A+, B+, C+, these are truly left as an exercise for the reader [...])"

This thread is already too long as it is (what with a couple of gigantic postings by myself), and I don't think it proper to make it even more so by posting a longish piece of code. In general, all the solutions I post are always 10 lines or less (generally much less), that's why I call it "Short and [...]". Longer code is always left as an exercise.

There's also the fact that I wouldn't want to seem cocky by posting a shorter, more polished piece of code. This isn't meant as a competition between me and gentle contributors but as a way to have some fun, exercise our HP calc-programming muscles, and learn some interesting math facts on the way.

Your program works perfectly, you get the A+ grade, and this is perfectly fine as far as I'm concerned; my own implementation is based in exactly the same algorithm as yours and so it doesn't bring any new exciting ideas or techniques, it's just more of the same and it isn't worth posting here for the reasons mentioned above.

Thanks for your interest and

Best regards from V.

Re: Short & Sweet Math Challenge #19: A+ for 71B

Message #68 Posted by [Gerson W. Barbosa](#) on 28 June 2007, 9:08 a.m.,
in response to message #61 by Egan Ford

Really impressive work, Egan!

I am amazed at your understanding of the paper and your being skillful enough to provide a code for the HP-71B. A++ IMVHO :-)

Best regards,

Gerson.

Re: Short & Sweet Math Challenge #19: A+ for 71B

Message #69 Posted by [Egan Ford](#) on 28 June 2007, 11:46 a.m.,
in response to message #68 by Gerson W. Barbosa

I just wish I understood 71B/BASIC better. This was the longest BASIC program I have ever done.

Edited: 28 June 2007, 11:47 a.m.

Re: Short & Sweet Math Challenge #19: A+ for 71B

Message #70 Posted by [Egan Ford](#) on 28 June 2007, 11:40 a.m.,
in response to message #61 by Egan Ford

I found a fixed a few minor bugs with digit strings > 10 digits. I also fixed the formatting of T to look better with > 9 digits. Lastly I added $10^{\text{LEN}(N)} \cdot \text{LOG}(10)$ as a comparison.

New code replaced old code in above post. New output:

DEPLETE: 123456789012

```

T =
  |  | 0 1 2 3 4 5 6 7 8 9 |
  |-----|
  | 1 | 1 2 1 1 1 1 1 1 1 1 |
  | 2 | 1 2 3 1 1 1 1 1 1 1 |
  | 3 | 1 2 1 4 1 1 1 1 1 1 |
  | 4 | 1 2 1 1 5 1 1 1 1 1 |
  | 5 | 1 2 1 1 1 6 1 1 1 1 |
  | 6 | 1 2 1 1 1 1 7 1 1 1 |
  | 7 | 1 2 1 1 1 1 1 8 1 1 |
  | 8 | 1 2 1 1 1 1 1 1 9 1 |
  | 9 | 1 2 1 1 1 1 1 1 1 10 |
  |10 |11 2 1 1 1 1 1 1 1 1 |
  |11 | 1 12 1 1 1 1 1 1 1 1 |
  |12 | 1 2 0 1 1 1 1 1 1 1 |

```

```

A =
  | .9 .8 .8 .8 .8 .8 .8 .8 .8 .8 .8 |
  | .1 .1 .1 .1 .1 .1 .1 .1 .1 .1 .1 |
  | 0 .1 0 0 0 0 0 0 0 0 0 |
  | 0 0 .1 0 0 0 0 0 0 0 0 |
  | 0 0 0 .1 0 0 0 0 0 0 0 |
  | 0 0 0 0 .1 0 0 0 0 0 0 |
  | 0 0 0 0 0 .1 0 0 0 0 0 |
  | 0 0 0 0 0 0 .1 0 0 0 0 |
  | 0 0 0 0 0 0 0 .1 0 0 0 |
  | 0 0 0 0 0 0 0 0 .1 0 0 |
  | 0 0 0 0 0 0 0 0 0 .1 0 |
  | 0 0 0 0 0 0 0 0 0 0 .1 |

```

Z(123456789012)= 2.30258509312E12 IN 1181.09 SEC USING 22560 BYTES

L(123456789012)= 2.30258509299E12 L(N)= $10^{\text{LEN}(N)} \cdot \text{LOG}(10)$

Re: Short & Sweet Math Challenge #19: A+ for 71B

Message #71 Posted by [Egan Ford](#) on 28 June 2007, 2:25 p.m.,
in response to message #70 by Egan Ford

Just for fun:

DEPLETE: 12345678900987654321

```

T =
  |  | 0 1 2 3 4 5 6 7 8 9 |
  |-----|
  | 1 | 1 2 1 1 1 1 1 1 1 1 |

```


2	1	2	3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	1	2	1	4	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	1	2	1	1	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5	1	2	1	1	1	6	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	1	2	1	1	1	1	7	1	1	1	1	1	1	1	1	1	1	1	1	1
7	1	2	1	1	1	1	1	8	1	1	1	1	1	1	1	1	1	1	1	1
8	1	2	1	1	1	1	1	1	9	1	1	1	1	1	1	1	1	1	1	1
9	1	2	1	1	1	1	1	1	1	10	1	1	1	1	1	1	1	1	1	1
10	11	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
11	12	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
12	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	13
13	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	14
14	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	15
15	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	16
16	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	17
17	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	18
18	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	19
19	1	2	20	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
20	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

A =

.9	.8	.8	.8	.8	.8	.8	.8	.8	.8	.8	.8	.8	.8	.8	.8	.8	.8	.8	.8	.8
.1	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1
0	.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	.1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	.1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	.1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	.1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	.1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	.1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	.1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	.1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.1	0	0	0

Z(12345678900987654321)= 2.30258509299E20 IN 8406.35 SEC USING 40012 BYTES
 L(12345678900987654321)= 2.30258509299E20 L(N)=10^LEN(N)*LOG(10)

S&SMC#19: My Original Solutions & Comments [LONG]

Message #72 Posted by [Valentin Albillo](#) on 28 June 2007, 12:38 p.m.,
 in response to message #1 by Valentin Albillo

Hi all,

Thanks to all of you who were interested in this S&SMC#19, most specially to the ones who worked really hard at the individual Grades and posted solutions. I'll give now my original solutions for the HP-71B (very easy to convert to other similarly capable models), plus additional comments.

Grade F:

Write a program to compute the N^{th} term for the following sequences:

$$\text{Sequence 1: } u_1 = 0, u_2 = 0, u_{n+2} = u_{n+1} + u_n/n$$

$$\text{Sequence 2: } u_1 = 0, u_2 = 0, u_{n+2} = u_{n+1}/n + u_n$$

where the number of terms N (which can be distinct for each sequence) is either hardcoded or given, your choice. After computing the N^{th} term, u_N , your program must then compute and output the following values:

$$\text{Sequence 1: } L = N \cdot 1 / u_N^1$$

$$\text{Sequence 2: } L = N \cdot 2 / u_N^2$$

Now use your program to help you try and predict, for each sequence, the corresponding theoretical exact limit of L when the number of terms N tends to infinity.

My original solution for the HP-71B is the following trivial 3-line (135 byte) program which computes and outputs the corresponding values of L for $N = 13$ and $N = 13000$, respectively.

```
1 DESTROY ALL @ STD @ K1=13 @ K2=13000
2 U=0 @ V=1 @ FOR N=1 TO K1 @ W=U/N+V @ U=V @ V=W @ NEXT N @ DISP N*1/U^1
3 U=0 @ V=1 @ FOR N=1 TO K2 @ W=V/N+U @ U=V @ V=W @ NEXT N @ DISP N*2/U^2
```

>RUN

```
2.71828182845
3.14195515673
```

As you may see, the sequences (very quickly) converge to $e = 2.71828182846$ and (very slowly) to $\pi = 3.14159265359$, respectively.

The **Surprise !** factor comes from the fact that both sequences are very, very similar (as can be seen both from their definitions and from the program listing above, where lines 2 and 3 are almost a slightly particularized copy-paste version of one another), yet one converges to the *uber*-important constant π , the other to the no-less-important constant e , suggesting a certain symmetry, a relationship, a "common origin" for both constants.

Some of you went to also compute 13,000 terms for the first sequence. This does no good, as 13 terms are all that is needed to achieve full 12-digit precision. After that, rounding errors do accumulate and by the time the 13,000-th term is reached, most of the precision is gone.

Grade D:

Write a program which, given N , will autoscale, label, and plot the $ISin(x, N)$ function, for arguments X in radians belonging to the fixed interval $[0, 2\pi]$ at increments of $\pi/40$, say, where

$$ISin(x, N) = \sin(\sin(\sin(\dots \sin(x) \dots)))$$

i.e., the iterated Sine of X , where N , a positive integer, is the number of iterations

Looking at the plot itself and the value of $Maxim$, try and predict what happens for increasing values of N and in the limit when N tends to infinity.

Given this challenge's nature, I was absolutely expecting that HP graphics models would be put to good use here, much more so than non-graphics ones, so I was utterly surprised that *not a single RPL solution was posted at all*. Go figure ! ...

My original solution for the (non-graphics) HP-71B is the following simple 8-line (268-byte) program:

```

10 DESTROY ALL @ OPTION ANGLE RADIANS @ D=3 @ FIX D @ INPUT "N=";N
20 K=FNS(PI/2,N) @ DISP "Maxim =" ;K @ FOR X=0 TO 2*PI STEP PI/20
30 Y=FNR(FNS(X,N)/K,D) @ U=45 @ V=FNR(Y*25+45,0) @ DISP X;Y;
40 IF V>U THEN DISP TAB(U);".";TAB(V);"*" ELSE DISP TAB(V);"*";
50 IF V<U THEN DISP TAB(U);"." ELSE IF V=U THEN DISP
60 NEXT X
70 DEF FNR(X,N)=INT(X*10^N+.5)/10^N
80 DEF FNS(X,N) @ FOR N=N TO 1 STEP -1 @ X=SIN(X) @ NEXT N @ FNS=X

```

where **FNS(X, N)** is a user-defined function that can be used from the program itself or directly from the command line to compute the Iterated Sine of X with N iterations, for instance

```
>FNS(1,1)
```

```
.841470984808 ( = SIN(1) )
```

```
>FNS(PI/3,5)
```

```
.594535945122 ( = SIN(SIN(SIN(SIN(SIN(PI/3)))))) )
```

and **FNR(X, N)** is a convenience function that returns X rounded to N digits (where N can be *negative* to achieve rounding to the *left* of the decimal point), for instance:

```
>FNR(PI,3)
```

```
3.142
```

```
>FNR(100*PI, -1)
```

```
310
```

Running the program for increasing values of N we get plots like the following for N = 10:

```

Maxim = 0.481
0.000 0.000      *
0.157 0.314      :
0.314 0.565      :
0.471 0.737      :
0.628 0.845      :
0.785 0.911      :
0.942 0.951      :
1.100 0.976      :
1.257 0.990      :
1.414 0.998      :
1.571 1.000      :
1.728 0.998      :
1.885 0.990      :
2.042 0.976      :
2.199 0.951      :
2.356 0.911      :
2.513 0.845      :
2.670 0.737      :
2.827 0.565      :
2.985 0.314      :
3.142 0.000      *
3.299 -0.314     *

```

```

3.456 -0.565      *      :
3.613 -0.737      *      :
3.770 -0.845      *      :
3.927 -0.911      *      :
4.084 -0.951      *      :
4.241 -0.976      *      :
4.398 -0.990      *      :
4.555 -0.998      *      :
4.712 -1.000      *      :
4.869 -0.998      *      :
5.027 -0.990      *      :
5.184 -0.976      *      :
5.341 -0.951      *      :
5.498 -0.911      *      :
5.655 -0.845      *      :
5.812 -0.737      *      :
5.969 -0.565      *      :
6.126 -0.314      *      :

```

and for N = 1000:

```

Maxim = 0.055
0.000 0.000      *      :
0.157 0.946      *      :
0.314 0.987      *      :
0.471 0.995      *      :
0.628 0.998      *      :
0.785 0.999      *      :
0.942 0.999      *      :
1.100 1.000      *      :
1.257 1.000      *      :
1.414 1.000      *      :
1.571 1.000      *      :
1.728 1.000      *      :
1.885 1.000      *      :
2.042 1.000      *      :
2.199 0.999      *      :
2.356 0.999      *      :
2.513 0.998      *      :
2.670 0.995      *      :
2.827 0.987      *      :
2.985 0.946      *      :
3.142 0.000      *      :
3.299 -0.946      *      :
3.456 -0.987      *      :
3.613 -0.995      *      :
3.770 -0.998      *      :
3.927 -0.999      *      :
4.084 -0.999      *      :
4.241 -1.000      *      :
4.398 -1.000      *      :
4.555 -1.000      *      :
4.712 -1.000      *      :
4.869 -1.000      *      :
5.027 -1.000      *      :
5.184 -1.000      *      :
5.341 -0.999      *      :
5.498 -0.999      *      :
5.655 -0.998      *      :
5.812 -0.995      *      :
5.969 -0.987      *      :
6.126 -0.946      *      :

```

and we see that the limiting waveform of nested SIN is *identically 0*, but if we keep scaling the function up, we get in the limit *a square wave*. If you've got a copy of *Mathematica* you can see it very clearly by looking at the "animation" created by this one-liner:

```
Do[ Plot[ Nest[Sin, x, n], {x, -2Pi, 2Pi}, PlotRange -> {-1, 1}, PlotLabel -> n ], {n, 1, 50} ]
```

where it's clear that what's happening is that the whole plot slowly *collapses* towards zero but *different parts collapse at different rates*, making the plot get more and more square. This is not difficult to prove by using a well-known theorem from real analysis that states that every bounded monotonic sequence approaches a finite limit.

The convergence to **0** is dreadfully slow, because for small arguments, we have with high accuracy:

$$\sin(x) \approx x - x^3/6$$

which means that when we replace x by $\sin(x)$, the only thing that gets us any closer to **0** is the term $x^3/6$, which, if x itself is small, it is a *much smaller* decrement indeed.

Grade C:

Find and output all values of $N < 10000$ such that both N is prime and

$$P(N) = S(N)$$

where $P(N)$ gives the number of primes $\leq N$ and $S(N)$ gives the sum of the factorials of the digits of N .

Your program must be optimized primarily for speed, then for program size.

The following is my original solution for the bare-bones HP-71B, a plain-vanilla 6-line (220 byte) program which doesn't require any ROMs or LEX add-ons and trades RAM usage for speed to deliver the goods rather quickly:

```
10 DESTROY ALL @ OPTION BASE 0 @ M=5000 @ INTEGER P(M) @ P(1)=1
20 FOR N=2 TO (SQR(2*M)+1)/2 @ IF P(N) THEN 40
30 FOR K=2*N*(N-1)+1 TO M STEP 2*N-1 @ P(K)=1 @ NEXT K
40 NEXT N @ T=1 @ FOR N=2 TO M @ IF P(N) THEN 60 ELSE K=2*N-1 @ S=0 @ T=T+1
50 S=S+FACT(MOD(K,10)) @ K=K DIV 10 @ IF K THEN 50 ELSE IF S=T THEN DISP 2*N-1
60 NEXT N @ DISP "OK"
```

>RUN

```
6521
OK
```

Lines 10-40 perform a fast sieving procedure to flag all odd numbers < 10000 as either prime or composite, recording the results in P , an integer array of adequate dimension.

This allows lines 40-60 to quickly scan the array, skipping all non-prime entries, and computing the $P_i(N)$ function effortlessly by simply updating the prime count each time an element is flagged as a prime. This running value of $P_i(N)$ is compared to the sum of the factorials of the digits of the current N (which is computed in a tight loop at line 50) and any solution found is then immediately output, resuming the search afterwards.

As it happens, **6521** is the only solution $< 10,000$, but there's just another one (and no more) in the range from 10,000 to infinity which some of you correctly and cleverly located, namely **5224903**.

If composite numbers were also allowed as solutions, there would be 23 solutions in all, starting at **6500** and ending at **11071599**. Among these, I find it interesting that **5854409**, **5854419**, and **5854429** happen to be both consecutive and in arithmetic progression.

This problem can be generalized to numerical bases other than 10, for example **6719**₁₀ is a solution in base 7 and **5378437**₁₀ is a solution in base 15.

Grade B:

Given a positive integer N , write a program to compute and output the value of S , defined as

$$S = \sum_{n=1}^N \frac{1}{(r_n)^2}$$

where r_n is the n -th positive root of $\tan(x) = x$, where x is expressed in radians, as usual.

Regrettably, this very interesting challenge, which has a truly surprising result, didn't seem to catch the eye of the vast majority of you for whatever reason and was left largely untouched.

PeterP made a brave attempt at dealing with it, but his final version still has a number of fatal flaws (such as the fact that it doesn't properly separate the roots from the poles and mixes them at times while computing the sum, etc), and so the numerical result isn't accurate. As is my policy in these cases, when a challenge gets essentially no acceptable responses I simply forfeit it as well, lest its appeal would be wasted.

The "surprise" factor of this challenge is the extremely unexpected result you get. Having an infinite sum of the roots of a transcendental trigonometric equation with arguments in radians, one would expect to get some weird no-name irrational, possibly transcendental number related to Pi in some way, say $\pi^3/310$ or some such value. But nothing could be further from the truth because what you actually get absolutely defies intuition, and paradoxically enough, it is fairly easy to concoct a purely formal proof "a la Euler" to prove that this unexpected result is mathematically exact and correct. The value you get for the B+ variant isn't that big a surprise, but is also most unexpected in its very unique way.

By the way, one of the main difficulties of this challenge lies in the fact that you must compute a lot of roots (possibly thousands), very quickly, and very accurately, in the sense that all roots should be accounted for with no missing entries, in their proper order, and that no poles are mistaken for roots. This requires a very precise separation between poles and roots and, as it happens, they tend to become extremely close for larger values, which enormously increases the difficulty of separating them. Besides, as you know, HP built-in solvers have a very hard time telling apart a pole from a root !

Grade A:

The limit of the sum of the reciprocals of the natural numbers 1 to N does diverge when N tends to infinite:

$$S = \sum_{n=1}^{N \rightarrow \infty} \frac{1}{n} \rightarrow \infty$$

However, if we simply leave out of the sum just those terms whose denominators include the digit 9 (such as 199, 34343900132, ...), the infinite sum then nicely converges to a finite value.

You must write a program to compute as accurately as possible this finite value of the infinite sum. You must optimize your program for both accuracy and speed.

This has been the grade of choice for many of you, with excellent results. This series converges far too slowly for direct summation, as has been extensively commented in the thread, so better algorithms are definitely required.

As some of you pointed out, there's a PDF document titled "SUMMING A CURIOUS, SLOWLY CONVERGENT SERIES" by Schmelzer and Baillie, freely downloadable from various web locations, which discusses the problem at length and gives in sufficient detail a couple of algorithms: the simpler Fischer algorithm, which can give the sum for the original Grade A problem to high accuracy in very fast times (but which can't be used for the A+ variant as it does not generalize to other digits or sequences of digits), and a more convoluted but extremely general algorithm by the authors, which can be used for the generalized sum. See the paper for the details.

I'll give now my two original A solutions, both implementing the Fischer algorithm but one of them optimized mainly for size, the other for speed.

This is my shorter original solution, a 6-line (265-byte) HP-71B program:

```

1 DESTROY ALL @ OPTION BASE 0 @ M=11 @ D=10 @ DIM B(M) @ B(0)=D @ FOR N=2 TO M
2 S=0 @ FOR K=N TO 2 STEP -1 @ S=S+FNC(N,K)*(D^(N-K+1)-D^K+1)*B(N-K) @ NEXT K
3 B(N-1)=(D*(11^N-D^N)-S)/N/(D^N-9) @ NEXT N @ H=D*LN(D)
4 FOR K=2 TO M-1 @ H=H-B(K-1)*FNZ(K)/D^K @ NEXT K @ DISP H
5 DEF FNC(N,K)=FACT(N)/FACT(K)/FACT(N-K)
6 DEF FNZ(X)=INTEGRAL(0,100,0,IVAR^(X-1)/(EXP(IVAR)-1))/GAMMA(X)

```

>RUN

22.9206766192 (6.83 seconds, Emu71 @ 2.4 Ghz, about 20 min in a real HP-71B)

and this is my longer (but 60+ times faster !) original solution, a 10-line (443-byte) solution for the HP-71B:

```

1 DESTROY ALL @ OPTION BASE 0 @ M=11 @ D=10 @ DIM B(M) @ B(0)=D @ FOR N=2 TO M
2 S=0 @ FOR K=N TO 2 STEP -1 @ S=S+FNC(N,K)*(D^(N-K+1)-D^K+1)*B(N-K) @ NEXT K
3 B(N-1)=(D*(11^N-D^N)-S)/N/(D^N-9) @ NEXT N @ H=D*LN(D)
4 FOR K=2 TO M-1 @ H=H-B(K-1)*FNZ(K)/D^K @ NEXT K @ DISP H
5 DEF FNC(N,K)=FACT(N)/FACT(K)/FACT(N-K)
6 DEF FNZ(Z) @ IF Z=2 THEN FNZ=PI*PI/6 ELSE IF Z=3 THEN FNZ=1.20205690316
7 IF Z=4 THEN FNZ=PI^4/90 ELSE IF Z=5 THEN FNZ=1.03692775514
8 IF Z=6 THEN FNZ=PI^6/945 ELSE IF Z=7 THEN FNZ=1.00834927738
9 IF Z<8 THEN END ELSE S=1 @ T=0 @ N=2
10 S=S+N^(-Z) @ N=N+1 @ IF S<>T THEN T=S @ GOTO 10 ELSE FNZ=S

```

>RUN

22.9206766192 (0.1 seconds, Emu71 @ 2.4 Ghz, about 20 seconds in a real HP-71B)

Both results are identical and correct to 12 digits, but the longer version runs 60+ times faster. The only difference between them lies in the definition of the Riemann Zeta function:

- The shorter version implements it as a user-defined function, FNZ(X), which can compute the Zeta function for real arguments, not only integers, using integration. As such, it does require the Math ROM, and it's somewhat slow. For instance:

```
>FNZ(10), FNZ(PI)
```

```
1.00099457513 1.17624173838
```

- On the other hand, the longer version implements the same user-defined function but only for integer arguments, using theoretical constant values for arguments from 2 to 7 and the power series for integer arguments greater than 7, which converges quickly enough for such arguments. This results in much faster computation times when compared to the integral version and further does not require the Math ROM, a bare bones HP-71B will do nicely.

As for the A+ variant, Egan Ford has worked very hard and very successfully to implement the quite complicated Schmelzer-Baillie's general algorithm, with excellent results, my most sincere congratulations for his outstanding contribution and thanks a lot for his interest in this challenge.

That's all. I hope you've enjoyed this S&SMC (and even learned a math thingy and/or programming trick or two), I certainly did enjoy seeing your *truly excellent posts*, as clever and insightful as always, you're the best !

And last but not least, a fond welcome to PeterP, which finally got to try and solve some of the grades like a pro. See, Peter ? It wasn't that difficult, man ! :-)

Best regards from V.

Re: S&SMC#19: My Original Solutions & Comments [LONG]

Message #73 Posted by **Thomas Klemm** on 28 June 2007, 7:41 p.m.,
in response to message #72 by Valentin Albillo

Hi Valentin

Instead of solving $\tan(x) = x$, I solved $x = \arctan(x)$ using a fixed-point iteration which converges faster as N grows. This avoids the troubles you mentioned concerning the poles at $\pi*(n + 1/2)$. Instead they are used as starting points.

Here's my HP-11C program:

```
STO I
CLx
STO 0
LBL 1
RCL I
PI
*
STO 1
LSTx
2
/
+
LBL 0
ENTER
ATAN
RCL 1
+
x#y
GTO 0
x^2
1/x
STO + 0
DSE
GTO 1
RCL 0
```

Yet the problem is that this is still running very slowly for large values of N. I must admit, that I've cheated and written a C-program to calculate values bigger than N = 1000:

N =	1	S = 0.04952768282755720119
N =	10	S = 0.09079018230840834005
N =	100	S = 0.09899682151630028929
N =	1000	S = 0.09989878003791599550
N =	10000	S = 0.099989868889474789056
N =	100000	S = 0.09999898679829559526
N =	1000000	S = 0.09999989867891767874
N =	10000000	S = 0.09999998986788264898
N =	100000000	S = 0.09999999898678817371

From this table I assume that the sum converges towards 0.1 which is in fact very astonishing.

Thanks for not letting out the cat from the bag yet.
I do appreciate your S&SMCs very much.

Kind regards
Thomas

Edited: 29 June 2007, 2:38 a.m.

Re: S&SMC#19: My Original Solutions & Comments [LONG]

Message #74 Posted by [Arnaud Amiel](#) on 29 June 2007, 3:10 a.m.,
in response to message #73 by Thomas Klemm

I had started a program and for 100 and 1000 it gave me the same answer as you got. I concluded my program is wrong. I will go back to it this weekend then.

Arnaud

Re: S&SMC#19: My Original Solutions & Comments [LONG]

Message #75 Posted by [Valentin Albillo](#) on 29 June 2007, 6:02 a.m.,
in response to message #73 by Thomas Klemm

Hi, Thomas:

Thomas posted:

"From this table I assume that the sum converges towards 0.1 which is in fact very astonishing."

First of all, thanks for your interest in this particular subchallenge, actually it was my pet one of the lot and it was sad for me to see it go into oblivion.

Your assumption is correct, the exact value of the infinite sum is indeed $1/10$. This is most unexpected, to say the least, and just goes to prove even further that math is an inexhaustible source of beauty and amazement.

I guess there are many teachers and math professionals out there who know their trigonometrics and calculus inside out, yet not one of them would guess that an infinite sum of the infinite transcendental roots of a basic transcendental trigonometric equation involving no arbitrary constants and nothing special at all would come up to be *exactly* **0.1**.

Not some weird possibly transcendental value presumably involving π , not some irrational surd, but a simple, rational value which happens to be the exact reciprocal of a small number, **10**. No wonder we have 10 fingers and 10 toes and thus our number system is (mostly) based around 10, as 10 is such an important constant that you need to add up an infinite number of quantities ultimately related to π in order to beget it. :-)

"Thanks for not letting out the cat from the bag yet. I do appreciate your S&SMCs very much."

Don't worry, I'll let the cat in the bag for a little while, just to see what you come up with, and also to see whether other contributors get interested and join us as well.

After that, if there's been enough interest, I'll publish my original program for the HP-71B, plus comments and perhaps even my formal proof of this astonishing fact, in the same style of bold formal manipulations that Euler liked to indulge in. Not rigorous in any way but enlightening nevertheless, easy-to-grasp, and ultimately a lot of fun.

Thank you very much for your appreciation and interesting contribution (for the HP-11C no less ! Way to go !), and

Best regards from V.

Re: S&SMC#19: My Original Solutions & Comments [LONG]

Message #76 Posted by [Egan Ford](#) on 6 July 2007, 1:28 a.m.,
in response to message #73 by Thomas Klemm

Quote:

Yet the problem is that this is still running very slowly for large values of N.

I thought I'd try this on my accelerated Nonpareil 15C.

N=1,000,000. ~15 hours later:



My alternative solution for B is based on an article by *Sidney Frankel*, *National Mathematics Magazine*, Vol. 11, No. 4. (Jan., 1937), pp. 177-182.

The following equation is an estimate of the n^{th} root of $\tan(x)=x$:

$$r_n = [1 + 1/a^2][n\pi + \tan^{-1}a] - 1/a \text{ where } a = (2n+1)\pi/2.$$

71B output:

```
N =      1  S = 0.04951757574 IN .03 SEC
N =     10  S = 0.09077991548 IN .05 SEC
N =    100  S = 0.09898655469 IN .12 SEC
N =   1000  S = 0.09988851321 IN 1.15 SEC
N =  10000  S = 0.09997960207 IN 11.05 SEC
N = 100000  S = 0.09998871997 IN 108.07 SEC
N = 1000000 S = 0.09998962993 IN 1075.05 SEC
N = 10000000 S = 0.09998967228 IN 10171.28 SEC
```

This does not converge to 1/10, but it's close.

71B code:

```

10 RADIANS
20 DESTROY ALL
30 INPUT "MAX=";M
40 FOR I=0 TO M-1
50 X=0
60 T=TIME
70 FOR J=1 TO 10^I
80 A=(2*J+1)*PI/2
90 B=(1+1/(A*A))*(J*PI+ATAN(A))-1/A
100 X=X+1/(B*B)
110 NEXT J
120 STD
130 N$=STR$(10^I)
140 T$=STR$(TIME-T)
145 S$=""
150 FOR K=LEN(N$) TO 9
160 S$=S$&" "
170 NEXT K
180 FIX 12
190 DISP "N = ";S$;N$;"   S = ";X;"IN ";T$;" SEC"
200 NEXT I

```

Edited: 6 July 2007, 1:44 a.m.

Re: S&SMC#19: My Original Solutions & Comments [LONG]

Message #77 Posted by [Thomas Klemm](#) on 6 July 2007, 9:15 a.m.,
in response to message #73 by Thomas Klemm

1. Assuming that x is close to one of the poles of $\tan(x)$ the difference dx can be estimated:


$$x + dx = (k + 1/2)*\pi$$

$$0 = \cot(x + dx) = \frac{1}{\tan(x + dx)} = \frac{1 - \tan(x) * \tan(dx)}{\tan(x) + \tan(dx)}$$

$$1 - \tan(x) \tan(dx) = 0$$

$$dx \sim \tan(dx) = \frac{1}{\tan(x)} = \frac{1}{x} \sim \frac{1}{(k + 1/2)*\pi}; \text{ since } dx \ll 1$$

Instead of using $q = (k + 1/2)*\pi$ as initial value for the fixed-point iteration the value $q - 1/q$ is used. Therefore lines 17-19 have been added to the program. This makes the program running about 20% faster.

(Or use  from [Tanc Function](#))

2. To estimate the remainder of the series I used the following formula I found [somewhere](#):

$$\begin{array}{c}
 N \\
 \hline
 \backslash \\
 < \\
 / \\
 \hline
 k = 1
 \end{array}
 \begin{array}{c}
 1 \\
 \hline
 x^2
 \end{array}
 \sim
 \begin{array}{c}
 2 \\
 \hline
 \frac{\pi^2}{6}
 \end{array}
 -
 \begin{array}{c}
 1 \\
 \hline
 N
 \end{array}
 +
 \begin{array}{c}
 1 \\
 \hline
 \frac{1}{2N}
 \end{array}
 -
 \begin{array}{c}
 1 \\
 \hline
 \frac{1}{6N}
 \end{array}
 +
 \begin{array}{c}
 1 \\
 \hline
 \frac{1}{30N}
 \end{array}
 -
 \begin{array}{c}
 1 \\
 \hline
 \frac{1}{42N}
 \end{array}
 +
 \begin{array}{c}
 1 \\
 \hline
 \frac{1}{30N}
 \end{array}
 + \dots$$

After a little magic I got:

$$\frac{\sum_{k=0}^{\infty} \frac{1}{(2k+1)^2} \sim \frac{1}{4N} - \frac{1}{68N^3} + \frac{1}{3016N^5} - \frac{1}{42128N^7} + \frac{1}{30512N^9} - \dots}{k=N}$$

That's what lines 32 - 56 in the improved HP-11C program are for:

```

01 STO I      16 +      31 GTO 1      46 +
02 1          17 ENTER  32 RCL 2      47 *
03 +         18 1/x    33 x^2     48 RCL 3
04 1/x       19 -      34 ENTER    49 +
05 STO 2     20 LBL 0   35 ENTER    50 RCL 2
06 CLx       21 ENTER  36 ENTER    51 *
07 STO 0     22 ATAN   37 RCL 7     52 2
08 LBL 1     23 RCL 1   38 *       53 PI
09 RCL I     24 +      39 RCL 6     54 /
10 PI        25 x#y    40 +      55 x^2
11 *         26 GTO 0   41 *       56 *
12 STO 1     27 x^2    42 RCL 5     57 RCL 0
13 LSTx      28 1/x    43 +      58 +
14 2         29 STO + 0 44 *
15 /         30 DSE    45 RCL 4
    
```

As a prerequisite the registers 3 - 7 must be loaded with the following constants:

```

4 1/x
STO 3
-48 1/x
STO 4
160 1/x
STO 5
-31 ENTER 5376 /
STO 6
127 ENTER 15360 /
STO 7
    
```

Now I get the following improved values S':

N = 1	S = 0.04952768282755720119	S' = 0.09920028974449183939
N = 10	S = 0.09079018230840834005	S' = 0.09999487093737255449
N = 100	S = 0.09899682151630028929	S' = 0.0999999335749214181
N = 1000	S = 0.09989878003791599550	S' = 0.099999999317650488
N = 10000	S = 0.09998986889474789056	S' = 0.09999999999315806
N = 100000	S = 0.09999898679829559526	S' = 0.0999999999999315
N = 1000000	S = 0.09999989867891767874	S' = 0.0999999999999998
N = 10000000	S = 0.09999998986788264898	S' = 0.1000000000000000
N = 100000000	S = 0.09999999898678817371	S' = 0.0999999999999999

Edited: 8 July 2007, 9:28 a.m. after one or more responses were posted

Re: S&SMC#19: My Original Solutions & Comments [LONG]

Message #78 Posted by [Egan Ford](#) on 6 July 2007, 1:09 p.m.,
in response to message #77 by [Thomas Klemm](#)

Brilliant! Accelerated 15C returns 0.1 with N=1000 in 79 sec:

Runtime: 0:00:01:19

0.1000000000
RAD

Date: Fri Jul 6 17:25:23 UTC 2007

Normal 15C returns 0.1 with N=1000 in 49 min:

Runtime: 0:00:49:02

0.1000000000
RAD

Date: Fri Jul 6 18:15:16 UTC 2007

EMU71 output of your solution:

N =	1	S = 0.04952768283	S' = 0.09920028974	IN .06 SEC
N =	10	S = 0.09079018231	S' = 0.09999487094	IN .01 SEC
N =	100	S = 0.09899682152	S' = 0.09999999336	IN .12 SEC
N =	1000	S = 0.09989878004	S' = 0.09999999999	IN 1.15 SEC
N =	10000	S = 0.0998986889	S' = 0.10000000000	IN 11.01 SEC

Physical 71B output:

N =	1	S = 0.04952768283	S' = 0.09920028974	IN 1.8 SEC
N =	10	S = 0.09079018231	S' = 0.09999487094	IN 4.64 SEC
N =	100	S = 0.09899682152	S' = 0.09999999336	IN 24.61 SEC
N =	1000	S = 0.09989878004	S' = 0.09999999999	IN 192.97 SEC
N =	10000	S = 0.0998986889	S' = 0.10000000000	IN 1769.42 SEC

Code:

```
: SSMC19B ;          ( FORGET SSMC19B TO CLEAN UP )
```

```
FVARIABLE MYTMP      ( CREATE SOME REGISTERS )
```

```
FVARIABLE MYSUM
```

```
FVARIABLE MYPI
```

```
FVARIABLE MYTIME
```

```
FVARIABLE N
```

```
FVARIABLE N2
```

```
FVARIABLE MYTMP2
```

```
: DOSPRIME
```

```
N2 RCL 1. F+ 1/X
```

```
2. Y^X MYTMP2 STO
```

```
127. 15360. F/
```

```
MYTMP2 RCL F*
```

```
-31. 5376. F/ F+
```

```
MYTMP2 RCL F*
```

```
160. 1/X F+
```

```
MYTMP2 RCL F*
```

```
-48. 1/X F+
```

```
MYTMP2 RCL F*
```

```
4. 1/X F+
```

```

N2 RCL 1. F+ 1/X F*
2. MYPI RCL F/ 2. Y^X F*
MYSUM RCL F+
;

: TANXROOTS
N RCL N2 STO
0. MYSUM STO      ( 0. -> MYSUM )
3.14159265358979 ( FLOAT STACK: PI )
MYPI STO          ( PI -> MYPI )
CLOCK             ( GET TIME -> X )
MYTIME STO       ( X -> MYTIME )
BEGIN             ( START LOOP N TO 1 )
  N RCL           ( N -> X )
  MYPI RCL       ( FLOAT STACK: N PI )
  F*             ( N * PI )
  MYTMP STO      ( STO X -> MYTMP )
  LASTX         ( PI )
  2. F/ F+      ( PI/2 + PI*N )
  FENTER 1/X F- ( 30% SPEED BUMP? )
  BEGIN         ( FIXED-POINT ITERATION START )
    FENTER      ( DUP )
    ATAN        ( ATAN X )
    MYTMP RCL   ( RCL MYTMP TO X )
    F+         ( X Y + )
    X=Y? -1 = UNTIL ( FIXED-POINT ITERATION END ON X=Y )
    -2. Y^X     ( 1/X^2 )
    MYSUM RCL F+ ( X MYSUM + )
    MYSUM STO   ( X -> MYSUM )
    N RCL 1. F- N STO ( DSE N )
  X=0? -1 = UNTIL
  ." N = "     ( MAKE IT PRETTY )
N2 RCL LGT 1. F+ IP
12 FTOI -
0 DO ." " LOOP
N2 RCL STD F.
." S = "
MYSUM RCL
12 FIX F.
." S' = "
DOSPRIME F.
." IN "
CLOCK MYTIME RCL F-
STD F. ." SEC" CR
;

: RUNIT
RADIANS          ( SET RADIANS )
0 DO
  I ITOF 10^X
  N STO
  TANXROOTS
LOOP
;

```

Edited: 6 July 2007, 2:30 p.m.

Re: S&SMC#19: Grade B [LONG]

Message #79 Posted by [Valentin Albillo](#) on 10 July 2007, 10:31 a.m.,
in response to message #78 by Egan Ford

Hi, Thomas and Egan:

Thank you very much to both of you for your interest in this poor, orphaned part of my challenge, you've both produced superb theory and actual code to try and solve it and your solutions are of course correct and truly excellent. I'm very obliged for your outstanding efforts.

This is my original solution for the HP-71B (7 lines, 271 bytes):

```

10 DESTROY ALL @ OPTION BASE 1 @ RADIANS @ L=100 @ R=FNROOT(4,5,TAN(FVAR)-FVAR)
20 S=1/(R*R) @ FOR I=2 TO 500 @ K=1/I @ R=FNROOT(R+PI-K,R+PI+K,TAN(FVAR)-FVAR)
30 GOSUB 60 @ NEXT I @ A=PI/2 @ B=1/PI @ C=2*PI @ D=1/4/PI+2/(3*PI3)
40 L=1000 @ FOR I=501 TO 71000 @ R=PI*I+A-B/I+1/(C*I*I)-D/(I*I*I)
50 GOSUB 60 @ NEXT I @ END
60 S=S+1/(R*R) @ IF NOT MOD(I,L) THEN DISP USING "6D,3X,Z.6D";I,S
70 RETURN

```

This basic transcendental equation (which does find its uses in Engineering) has an infinite number of positive roots. As there's the problem of proper root identification and separation, which gets more difficult as the roots grow larger since they tend to be extremely near the poles, with the subsequent danger of some root being skipped or some pole being mistaken for a root, my routine uses the Solver just for the first 500 roots, which can be properly separated by generating adequate initial guesses for each one.

After that, a simple asymptotic formula for the roots is used, which gives 12 correct decimal digits for each root and much faster than the Solver can. Thus, this formula does produce all remaining roots, from the 501th onwards. The crossing index, 500, has been chosen empirically because that's approximately the point where the asymptotic formula is accurate enough to take over the Solver. Were it not for the asymptotic formula, the Solver alone would begin to skip or misidentify roots around the 3,000th one given the currently generated initial guesses.

As written, my solution will generate and sum the first 71,000 roots (just edit the 71000 in the listing above if you want to generate and process more roots), like this:

>RUN

```

100  0.098997
200  0.099496
300  0.099663
400  0.099747
500  0.099798
1000 0.099899
2000 0.099949
3000 0.099966
4000 0.099975
5000 0.099980
6000 0.099983
7000 0.099986
... ..
65000 0.099998
66000 0.099998
67000 0.099998
68000 0.099999
69000 0.099999
70000 0.099999
71000 0.099999

```

and it does it in just 29 seconds under Emu71, some 2 hours on a physical HP-71B, with the final result being 0.099999 to six decimal figures, in perfect agreement with the theoretical value 1/10.

I'll give now a short & sweet proof I've shamelessly concocted to show that the correct theoretical sum is exactly 1/10. This proof of mine is admittedly in the spirit of some of the most celebrated Euler's ones, i.e., utterly non-rigorous, but it shows the correct solution which, once allegedly known, you can then proceed to rigorously prove either by other means or by strengthening this simple proof. Let's proceed:

- 1) Let's put the original equation in a more convenient form:

$$\tan(x) = x \rightarrow \sin(x)/\cos(x) = x$$

$$\rightarrow \sin(x) = x \cdot \cos(x)$$

2) Let's substitute $\sin(x)$ and $\cos(x)$ by their Taylor series expansions and collect all terms at the left side:

$$\begin{aligned} x - x^3/3! + x^5/5! - x^7/7! + \dots &= x \cdot (1 - x^2/2! + x^4/4! - x^6/6! + \dots) \\ &= x - x^3/2! + x^5/4! - x^7/6! + \dots \end{aligned}$$

->

$$(1/2! - 1/3!)x^3 - (1/4! - 1/5!)x^5 + (1/6! - 1/7!)x^7 - \dots = 0$$

3) Let's divide both sides by x^3 to get rid of zero roots:

$$(1/2! - 1/3!) - (1/4! - 1/5!)x^2 + (1/6! - 1/7!)x^4 - \dots = 0$$

4) and now, considering this as an "infinite-degree polynomial equation", we then have that the required sum of the (transformed) roots is, by the Vieta formulas:

$$\text{Sum} = (1/4! - 1/5!)/(1/2! - 1/3!) = 1/10$$

q.e.d

This all depends on the fact of whether the "infinite-degree polynomial equation"s roots do sufficiently coincide with the appropriate transformation of the ones of the original transcendental equation. We can get some peace of mind by testing for various degrees N , to get a taste of what happens when N goes to infinity. The following short program asks for the degree N , does construct and truncate the above polynomial to the specified degree (only in X instead of in X^2), and finds all its roots and their sum:

```
1 DESTROY ALL @ OPTION BASE 0 @ INPUT "N="; N @ DIM P(N) @ COMPLEX R(N+1),S
2 FOR I=0 TO N @ P(I)=(-1)^I*(1/FACT(2*I+2)-1/FACT(2*I+3)) @ NEXT I @ MAT R=PROOT(P)
3 S=0 @ FOR I=0 TO N-1 @ S=S+R(I) @ DISP R(I),1/SQR(R(I)) @ NEXT I @ DISP S
```

Let's test with a 40th-degree truncation:

>RUN

N = 40 [ENTER]

...

```
(0.000780, -0.000329) (33.696792, 6.813122)
(0.001299, 0.000000) (27.747075, -0.000000)
```


(0.001344, 0.000000)	(27.273576, -0.000000)
(0.001810, 0.000000)	(23.503515, -0.000000)
(0.002410, 0.000000)	(20.371987, -0.000000)
(0.003372, 0.000000)	(17.220729, -0.000000)
(0.005054, 0.000000)	(14.066195, -0.000000)
(0.008410, 0.000000)	(10.904122, -0.000000)
(0.016756, 0.000000)	(7.725252, -0.000000)
(0.049528, -0.000000)	(4.493409, 0.000000)

(0.100000, 0.000000)

As you can see, all the roots add up to exactly 0.1, and taking the inverse of their square root (as they are the inverse of the squares of the original roots, as needed for the sum), we get a very good approximation of the first roots of the original transcendental equation, namely

4.493409, 7.725252, 10.904122, 14.066194, 17.220755, 20.371303, ...

so we find it easy to believe that for higher degrees the accuracy improves and in the limit, the result holds and the sum for the roots of the polynomial does coincide with the sum of the roots of the original transcendental equation.

Thanks again and

Best regards from V.

Re: S&SMC#19: Grade B [LONG]

Message #80 Posted by [Thomas Klemm](#) on 10 July 2007, 2:08 p.m.,
in response to message #79 by Valentin Albillo

Hi Valentin

Your proof is cool, I like it!

And thanks to Egan for his posts which I read with much interest.
It was nice to see my program transformed into another language.
Now I even know some FORTH.

Kind regards
Thomas

Re: S&SMC#19: Grade B [LONG]

Message #81 Posted by [Egan Ford](#) on 10 July 2007, 3:32 p.m.,
in response to message #79 by Valentin Albillo

Do you have any comments on B+ you would like to share? Computing yields 0.0972640247325, but it would take an enormous leap to see that as $(e^*e - 7)/4$. The masses without a proof would have to search text books and the web to find the exact answer (as I had [here](#)). I ask for your thoughts because of the teaser on your web site: Boldly Going - Identifying Constants.

Thanks for all the challenges, I enjoy the mental exercise.

Re: S&SMC#19: Grade B [LONG]

Message #82 Posted by [PeterP](#) on 10 July 2007, 7:31 p.m.,

in response to message #79 by Valentin Albillo

was tempted to do this anonymously to hide in light of the much better mathematicians around (it now looks more like heresy that I even participate openly in those challenges next to you guys)

Anyway, great minds think alike - [this short paper](#) has a similar proof as Valentin and then goes on to extend it to a more general class of transcendental equations and gives some examples. Maybe this is interesting to some so I thought I post it. Please forgive it this is 'old news' to all of you already.

Cheers

Peter

Re: S&SMC#19: Grade B [LONG]

Message #83 Posted by [Egan Ford](#) on 10 July 2007, 8:34 p.m.,

in response to message #82 by PeterP

You may also enjoy R. M. Young, A Rayleigh popular problem, *American Mathematical Monthly* 93 (1986) 660-664.

62 solutions were submitted. Two are documented in the article.

Edited: 10 July 2007, 8:35 p.m.

Re: S&SMC#19: My Original Solutions & Comments [LONG]

Message #84 Posted by [PeterP](#) on 29 June 2007, 9:41 p.m.,

in response to message #72 by Valentin Albillo

Valentin,

Thanks for all your encouraging and friendly prodding, it was great fun. And indeed, it was not only a lot of fun but was going easier with time and more practice!

As for B, I fully appreciate you'd like to keep things under wraps for a later point in time. Maybe however you can give me offline at my email some tips and pointers. As I mentioned in my post, after I had programmed something and got stuck in trying to explain the result, I ended up finding a paper which explains it and indeed shows that $1/10$ is the right value. So it seems that my program is wrong yet still delivers a somewhat correct result... Your insight and guidance would be most appreciated!

All the best and thanks for providing me (and us I daresay) with lots of fun and learning!

Cheers

Peter

A message from Thomas Schmelzer to all S&SMC#19 A/A+ participants.

Message #85 Posted by [Egan Ford](#) on 29 June 2007, 9:45 p.m.,

in response to message #1 by Valentin Albillo

Thomas Schmelzer ([SUMMING A CURIOUS, SLOWLY CONVERGENT SERIES](#)) asked me to post this on his behalf:

Quote:

Hoi zame,

I am pleased you have attacked the harmonic series omitting certain digits. To implement this kind of code on an old HP calculator is certainly challenging and geeky! Well done. It seems this problem provided some fun and this was exactly the intention of the paper I wrote with Robert Baillie.

Best wishes,

Thomas

Post your replies here and I will send him the link. Thanks!

Edited: 29 June 2007, 9:52 p.m.

Re: A message from Thomas Schmelzer to all S&SMC#19 A/A+ participants.

Message #86 Posted by [Valentin Albillo](#) on 29 June 2007, 10:47 p.m.,
in response to message #85 by Egan Ford

Hi, Egan:

Thanks for posting Mr. Schmelzer's comment. This would be my reply to him:

"Dear Mr. Schmelzer:

Thank you very much for your kind and encouraging words re Mr. Ford's truly excellent implementation of your algorithm in a vintage but superb HP handheld.

I was indeed inspired by your very paper when I included this assignment as part of one of my regular 'challenges' to the HP fan community. After reading it, I thought it was a most interesting topic and a superb paper at that, clear and detailed, which prompted me to first implement both algorithms (Fischer's and yours) in HP-71B's code, which proved immensely fun, then issue the challenge to the rest of exceedingly clever programmers who regularly visit this HP forum, for they to share the fun with me, which they certainly did.

So please receive my most sincere thanks for your kindness, which I beg you to please extend to Mr. Baillie, and both of you may rest assured that if providing some fun was the exact intention of your paper; you succeeded in spades.

Best regards from Valentin Albillo"

Best regards from V.

Edited: 29 June 2007, 10:55 p.m.

Re: Short & Sweet Math Challenge #19: Surprise ! [LONG] B+ Solution

Message #87 Posted by [Egan Ford](#) on 6 July 2007, 12:29 a.m.,
in response to message #1 by Valentin Albillo

Quote:

Note: Get a B+ !

If you manage to solve the above, you can get a B+ by computing and, most specially, *identifying* the exact sum but this time using

$$(r_n)^2 + 1$$

in the denominators instead of simply $(r_n)^2$

Exact sum:

$$S = \sum_{n=1}^{\infty} \frac{1}{(r_n)^2 + 1} = \frac{e^2 - 7}{4}$$

Inspired by Thomas Klemm's amazingly short and sweet 11C program I thought I would do the obvious and add one to the denominator and see what I got:

```
N =      1   S = 0.04719044923 IN .05 SEC
N =     10   S = 0.08805677133 IN .02 SEC
N =    100   S = 0.09626084957 IN .18 SEC
N =   1000   S = 0.09716280477 IN 1.27 SEC
N =  10000   S = 0.09725389363 IN 11.99 SEC
N = 100000   S = 0.09726301153 IN 115.13 SEC
N = 1000000   S = 0.09726392341 IN 969.29 SEC
N = 10000000   S = 0.09726401460 IN 8463.29 SEC
```

The above output was generated by the *Klemm-like* 71B FORTH program at the end of this post. I selected FORTH because it is faster than BASIC (well faster than *my* BASIC) and porting RPN to FORTH is easy.

As you can see each digit of precision takes ~10x longer to run. Extra digits would have cost a day, then 10 days. Another problem is that the larger terms are too close to zero.

Like Thomas I did resort to a small C program (64-bit ints, 128-bit floats) to get a few more digits:

```
N =      1   S = 0.0 4719044922581127794
N =     10   S = 0.0 8805677132930494860
N =    100   S = 0.09 626084957021318412
N =   1000   S = 0.097 16280477399029997
N =  10000   S = 0.0972 5389362741386840
N = 100000   S = 0.09726 301153095815560
N = 1000000   S = 0.09726 392341158023534
N = 10000000   S = 0.0972640 1460054520640
N = 100000000   S = 0.09726402 371945072695
N = 1000000000   S = 0.097264024 63133223744
N = 10000000000   S = 0.097264024 71681260679
```

I have two sources I check when presented with mystery numbers. First, the book *Mathematical Constants*. Second, *Plouffe's Inverter* (<http://pi.lacim.uqam.ca/eng/>). The later scored a hit (several hits actually). The most interesting was:

$$9726402473266255 = (m414) \frac{1}{\ln(\exp(1))} \cdot \text{DuboisR2} \cdot \sqrt{5}^2$$

Odd result. You can simplify in your head and get $\text{DuboisR2} * 5$. The same *Inverter* page has a link to the constants. DuboisR2 is the **2nd Du Bois Reymond constant**. $\text{DuboisR2} = (e^2 - 7)/2 = 0.1945280494\dots$, so $\text{DuboisR2} * 5 = 0.97264024\dots$, therefore $\text{DuboisR2} * 5/10$ or $\text{DuboisR2}/2 = 0.097264024\dots$

The **2nd Du Bois Reymond constant** is defined in pages 238-239 in the book *Mathematical Constants*. An interesting read (books.google.com if you do not have this book):

Paraphrased - "Let r_1, r_2, r_3, \dots denote all positive solutions of the equation $\tan(x) = x$. Then

$$c_m = 2 * \sum_{j=1}^{\infty} \frac{1}{((r_j)^2 + 1)^{m/2}}$$

"

It is not too hard to see that $c_2/2$ equals the B+ problem. Since $c_2 = (e^2 - 7)/2$ then $c_2/2 =$

$$\frac{e^2 - 7}{4} = 0.097264024\dots$$

The solution to problem B is also mentioned in *Mathematical Constants* (page 239).

Additional **Du Bois Reymond Constants** info: <http://mathworld.wolfram.com/DuBoisReymondConstants.html>.

71B FORTH code:

```
: SSMC19BP ;          ( FORGET SSMC19BP TO CLEAN UP )

FVARIABLE MYTMP      ( CREATE SOME REGISTERS )
FVARIABLE MYSUM
FVARIABLE MYPI
FVARIABLE MYTIME
FVARIABLE N
FVARIABLE N2

: TANXROOTS
  N RCL N2 STO
  0. MYSUM STO      ( 0. -> MYSUM )
  3.14159265358979 ( FLOAT STACK: PI )
  MYPI STO          ( PI -> MYPI )
  CLOCK            ( GET TIME -> X )
  MYTIME STO       ( X -> MYTIME )
  BEGIN
    N RCL           ( N -> X )
    MYPI RCL        ( FLOAT STACK: N PI )
    F*              ( N * PI )
    MYTMP STO       ( STO X -> MYTMP )
    LASTX           ( PI )
    2. F/ F+        ( PI/2 + PI*N )
    BEGIN
      FENTER        ( DUP )
      ATAN          ( ATAN X )
      MYTMP RCL     ( RCL MYTMP TO X )
      F+            ( X Y + )
      X=Y? -1 = UNTIL ( FIXED-POINT ITERATION END ON X=Y )
      2. Y^X        ( X^2 )
      1. F+         ( B+ )
      1/X           ( 1/[X^2+1] )
      MYSUM RCL F+  ( X MYSUM + )
      MYSUM STO     ( X -> MYSUM )
      N RCL 1. F- N STO ( DSE N )
    X=0? -1 = UNTIL
    ." N = "       ( MAKE IT PRETTY )
    N2 RCL LGT 1. F+ IP
    10 FTOI -
    0 DO ." " LOOP
    N2 RCL
    STD F. ." S = "
    MYSUM RCL
    FIX 12 F.
    ." IN "
```

```
CLOCK MYTIME RCL F-  
STD F. ." SEC" CR  
;  
  
: RUNIT  
RADIANS          ( SET RADIANS )  
0 DO  
  I ITOF 10^X  
  N STO  
  TANXROOTS  
LOOP  
;
```

Edited: 6 July 2007, 2:34 a.m.

[[Return to Index](#) | [Top of Index](#)]



[Go back to the main exhibit hall](#)