[VA] SRC #012c - Then and Now: Sum

28th November, 2022, 00:20

Post: #1

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

[VA] SRC #012c - Then and Now: Sum

Hi, all,

After the nice solutions posted for *Problem 2* and the *2,500 views* mark exceeded (plus no less than *three* other related threads created by **Albert Chan**, **Thomas Klemm** and **J-F Garnier**,) now's the time for the next part of my ***SRC #012 - Then and Now***, where I'm showing that vintage *HP* calcs which were able problem-solvers back **THEN** in the 80's can **NOW** solve recent tricky problems intended to be tackled with fast modern computers, never mind slow ancient pocket calcs.

In the next weeks I'm proposing **six** *increasingly harder* such problems for you to try and solve using your vintage *HP* calcs while abiding by the ***mandatory rules*** summarized here:

> You must use **VINTAGE HP CALCS** (physical/virtual,) coding in either **RPN** (inc. *mcode*), **RPL** (variants existing at the time, inc. *SysRPL*) or **HP-71B** languages (inc. *BASIC, FORTH, Assembler*), so **NO** *XCAS, MATHEMATICA, MAPLE, EXCEL, C/C++/C#, PYTHON,* etc., **NO LENGTHY MATH** SESSIONS and **NO CODE PANELS**.
>
> On the plus side, you may use any official/popular modules, pacs or libraries available at the time, such as the **Math Pac**, **HP-IL** and **JPC ROM**s for the **HP-71B**, the **Advantage Module**, **PPC ROM** and *Extended Memory* for the **HP-41**, and assorted *libraries* for the ***RPL*** models, to name a few.

That said, I'm done with holding back so here you are, a new problem which deals with the ***sum*** of an infinite series, namely:

> As an aside, you may remember the *RPN* program featured in my ***HP Article VA001 - Long Live the HP-11C***, which used an efficient algorithm to quickly and accurately compute the sum of *alternating* series (say *1 - 1/2 + 1/3 - 1/4 + ...*), even oscillating (say *1 - 1 + 1 - 1 + ...*) or divergent ones (say *1 - 2 + 3 - 4 + ...*)

Problem 3: Sum

Write a sumptuous program to summarily sum this unassuming yet serious series:

$$S = \sum_{n=1}^{\infty} (f(n))^{-1}$$

where ***f(n)*** is defined thus: if *n < 3* then ***f(n)*** = *n* else ***f(n)*** = *n * f(d(n))*, where *d(n)* = number of binary digits of *n*.

An example will make it clear; to compute ***f(10)*** we proceed as follows:

We need *f(10)*, which is *10 * f(d(10)) = 10 * f(**4**)* {as *10* = 1010_2 which has ***4*** binary digits}
and now we need *f(4)*, which is *4 * f(d(4)) = 4 * f(**3**)* {as *4* = 100_2 which has ***3*** binary digits)
and now we need *f(3)*, which is *3 * f(d(3)) = 3 * f(**2**)* {as *3* = 11_2 which has ***2*** binary digits)
and now we need *f(2)*, which is ***2*** by definition

and backtracking we have *f(3)* = 3*2 = ***6*** and then *f(4)* = 4*6 = ***24*** and finally ***f(10)*** = 10*24 = ***240***.

Your program should have *no inputs* and must output the sum and automatically end. You should strive for *10-12 correct digits* and the faster the running time the better.

Some useful advice is to try and find the correct balance between the program doing all the work with no help from you *(i.e. sheer brute force, which might take substantial running time,)* or else using some insight to help speed up the process. Your choice.

If I see interest, in a week or so I'll post my own *original solution* for the **HP-71B**, which is a *6-line* program that does the job. In the meantime, let's see your very own clever solutions **AND** *remember the above rules*, please.

V.

PM WWW FIND EDIT QUOTE REPORT

28th November, 2022, 20:19 **Post: #2**

FLISZT
Junior Member

Posts: 43
Joined: Nov 2022

RE: [VA] SRC #012c - Then and Now: Sum

Hi everyone,

This is my very first post on this forum.
I've been lurking around for years, at least 12 if I take into account the old forum.
This "Sum" problem has seemed not too mathematical to me. It inspired me and finally decided me to register.

So, last night I wrote an RPL program on one of my two hp-50g. Then I checked to see if it worked on my trusty hp-28s - released in the 1980s, the 28 series is the only RPL calculator that follows the rules.

To my surprise, the program did not work: I didn't know - or had forgotten - that arithmetic operations on local variables were not yet implemented on these RPL calculators.
After the few necessary corrections, the size of the program increased from 154 to 161 bytes.
The program uses two local variables.

During the day, while rethinking my program, I realized that some instructions were unnecessary and the final size of the program is therefore reduced to 148.5 bytes.

Here are some results to check if my program is OK (?):

f(0) = 0
f(1) = 1
f(2) = 2
f(3) = 6
… which seem obvious at this point!

f(11) = 264
f(12) = 288
f(100 = 4200

… and f(10) = 240, of course!

EMAIL PM FIND QUOTE REPORT

29th November, 2022, 02:00 **Post: #3**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012c - Then and Now: Sum

.
Hi, **FLISZT**,

> **FLISZT Wrote:** (28th November, 2022 20:19)
> This is my very first post on this forum.
> I've been lurking around for years, at least 12 if I take into account the old forum.

Welcome to the forum and congratulations for stopping being a lurker, hope you'll enjoy the place, very nice & knowledgeable people here (mostly).

> **Quote:**
> This "Sum" problem has seemed not too mathematical to me. It inspired me and finally decided me to register.

Well, thanks for your interest in this thread I recently created and I'm very glad it inspired you to join us, the more we are the merrier. If you like challenges, you might want to have a look at the Challenges section of my HP calcs site (it also

features SHARP Pocket Computers).

> **Quote:**
> So, last night I wrote an RPL program on one of my two hp-50g. Then I checked to see if it worked on my trusty hp-28s [...] To my surprise, the program did not work [...] During the day, while rethinking my program, I realized that some instructions were unnecessary and the final size of the program is therefore reduced to 148.5 bytes.

Good. Please post the code when you feel you're ready to share it with us all.

> **Quote:**
> Here are some results to check if my program is OK (?):
>
> f(0) = 0, f(1) = 1, f(2) = 2, f(3) = 6 … which seem obvious at this point!
>
> f(11) = 264, f(12) = 288, f(100 = 4200 … and f(10) = 240, of course!

Of course. All of them are correct.

Best regards.
V.

PM WWW FIND EDIT QUOTE REPORT

29th November, 2022, 03:41 (This post was last modified: 29th November, 2022 22:16 by FLISZT.) **Post: #4**

FLISZT
Junior Member

Posts: 43
Joined: Nov 2022

RE: [VA] SRC #012c - Then and Now: Sum

Hi Valentin !

Thank you for welcoming me.

Yes I'm ready.

With the 9th Symphony of Bruckner broadcasted at the radio / Eugen Jochum / Bavarian Radio Orchestra (1954) and now the Concerto for Piano & Orch #3 / Robert Casadesus, New York Philh, Dimitri Mitropoulos (live, 1957)… how could it not be the case?!

Here is my attempt:

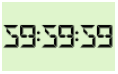
```
@ duplication of the value n; m is initialized at 1 (the neutral element of the multiplication…)
<<
DUP 1 -> b m
@ flag 1 is set; BIN mode selected
<<
1 SF
BIN
@ as long as flag 1 is set
WHILE 1 FS?
REPEAT
b 2 > IF
THEN
@ start counting the number of binary digits:
@ conversion of b into a binary object
b R->B
@ conversion of this object into a string (10 ⇒ "# 1010b")
->STR
SIZE
@ size of the string minus 3 characters
@ 2 of them are the "prefix" of the binary object / 1 is the suffix: "b"
3 -
@ n * f(d(n))
DUP
'b' STO
m *
'm' STO
ELSE
@ flag 1 is cleared (signal of ending the "WHILE" loop)
```

```
1 CF
END
END
@ The last calculated value of m (or 1… ) is multiplied
@ by the value of n put in the stack at the very beginning
m *
>>
>>
```

Edit: typo
Edit2 : no code panel!

EMAIL PM FIND QUOTE REPORT

29th November, 2022, 10:52 (This post was last modified: 29th November, 2022 14:49 by Werner.) **Post: #5**



Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

Well, I have a result..

2.08637766501

I still have to double-check my code and my reasoning ;-)
Execution is pretty fast (a blink of an eye on Free42, will run it on my 42S when I'm reasonably certain it is correct)

Cheers, Werner

EMAIL PM FIND QUOTE REPORT

29th November, 2022, 16:52 (This post was last modified: 29th November, 2022 17:49 by J-F Garnier.) **Post: #6**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

> **Werner Wrote:** (29th November, 2022 10:52)
> I still have to double-check my code and my reasoning ;-)

At least you have a reasoning :-)
Clearly brute force is not the way to go, but I can't figure out a short-cut.
But knowing that you got a solution so quickly is already a clue.

J-F

EMAIL PM WWW FIND QUOTE REPORT

29th November, 2022, 18:31 **Post: #7**

John Keith
Senior Member

Posts: 883
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

I wouldn't call f(n) an "unassuming series", it's really quite a complex and interesting series. Being recursive it can take some time to compute for larger numbers but there are some shortcuts we can use to speed up the computation.

First of all, d(n), the number of bits needed to represent n, is a very simple sequence that can be computed rapidly with a simple loop. Secondly, there is an underlying fractal pattern in f(n) and to compute k terms one need only compute approximately 2 * LOG2(k) terms recursively, and the rest can be filled in by a simple loop needing only 1 addition per term.

This is still basically brute force and I don't have a complete program ready yet but at least the germ of an idea.

EMAIL PM FIND QUOTE REPORT

pinkman The contents of this message are hidden because pinkman is on your ignore list. Show this Post

30th November, 2022, 08:36 **Post: #9**

ThomasF
Member

Posts: 117
Joined: Sep 2016

RE: [VA] SRC #012c - Then and Now: Sum

> **Werner Wrote:** (29th November, 2022 10:52)
> Well, I have a result..
>
> 2.08637766501

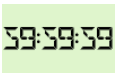
Hmm ... I also thought I had a result. I did some brute force and found a way to simplify, but maybe I'm headed somewhere else on a bad track ...
The track I followed led me closer to 1.96643... and should be possible to fit in a RPN version.

But, I just have to check some more to see if I'm on the wrong path this time

Cheers,
Thomas

EMAIL PM FIND QUOTE REPORT

30th November, 2022, 11:06 **Post: #10**



Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

Ran my program on my 42S, and produced the exact same result in 55 seconds.
Werner

EMAIL PM FIND QUOTE REPORT

30th November, 2022, 11:15 (This post was last modified: 30th November, 2022 12:29 by J-F Garnier.) **Post: #11**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

Since we are several to have tried the brute force approach, here are my results that show that it's a no go.

I rewrote 1/f(n) to 1/(n.f(d)) and used the fact that all the n values between 2^d and 2^(d+1)-1 have the same number of binary digits d+1 so the sum of these values can be factored with the same f(d+1).
Translated into code (sorry for the change of notation):

```
J1=2^(K-1) @ J2=2*J1-1
S=0 @ FOR J=J1 TO J2 @ S=S+1/J @ NEXT J
S1=S1+S/F(K)
```

I also noticed that the intermediate value S converges to LOG(2), so to have an idea of how the series behaves I replace the partial sum S with LOG(2) for K>=15, i.e. for the f(n) with n>2^15.
I went up to K=4000, i.e. numbers with 4000 binary digits (> 10^1200) , and the sum has still not converged.
I can just say that the sum is larger than 1.95, which is compatible with both Werner and Thomas prelim results.

Any confirmation of my analysis is welcome!

```
10 ! SRC12C
20 L=4000
30 DIM F(L)
40 ! calculate the first L F(I)
50 F(1)=1 @ F(2)=2
60 FOR I=3 TO L
70 D=INT(LOG(I)/LOG(2))+1
90 F(I)=I*F(D)
100 NEXT I
120 !
130 S1=1/F(1)+1/F(2)+1/F(3)
140 FOR K=3 TO L
142 S=LOG(2) ! approx value used for K>=15
145 IF K>=15 THEN 210
150 J1=2^(K-1) @ J2=2*J1-1
170 S=0 @ FOR J=J1 TO J2 @ S=S+1/J @ NEXT J
210 S1=S1+S/F(K)
220 IF K<15 OR MOD(K,500)=0 THEN DISP K;S;S1
230 NEXT K
```

```
235 END

>RUN
K, partial sum S, sum S1
3 .759523809524 1.79325396826
4 .725371850372 1.82347779536
5 .709016202207 1.84711166877
6 .701020708269 1.86658446622
7 .697068688885 1.88318133976
8 .695104120223 1.88680167372
9 .694124696722 1.89001521398
10 .693635700225 1.89290536273
11 .693391380792 1.89553184523
12 .693269265777 1.89793903018
13 .69320821942 1.9001608514
14 .693177699089 1.90222388027
500 .69314718056 1.95019974933 (S estimated from here)
1000 .69314718056 1.95220716995
1500 .69314718056 1.95327727197
2000 .69314718056 1.95403237901
2500 .69314718056 1.95457439294
3000 .69314718056 1.9550131171
3500 .69314718056 1.95538406366
4000 .69314718056 1.95570539887
```

J-F

EMAIL PM WWW FIND QUOTE REPORT

30th November, 2022, 11:55 (This post was last modified: 30th November, 2022 12:59 by ThomasF.) **Post: #12**

ThomasF
Member

Posts: 117
Joined: Sep 2016

RE: [VA] SRC #012c - Then and Now: Sum

> **J-F Garnier Wrote:** (30th November, 2022 11:15)
> Any confirmation of my analysis is welcome!

Hi J-F,

I can't confirm, but that was the same track I was using.

Realizing that f(d) is constant between 2^d and 2^(d+1)-1, and if moved out of the summation we are left with a harmonic series, i.e we could just sum over something like this: 1/f(d) * (H(2^(d+1)-1) - H(2^d-1)) with d=1..K, and testing that hypothesis converges to something close to 1.96643...

At least we get the similar result - is it the right track ... ? Maybe

Edit: Corrected the formula - got one reciprocal too many ...

Cheers,
Thomas

EMAIL PM FIND QUOTE REPORT

30th November, 2022, 12:44 (This post was last modified: 1st December, 2022 18:00 by Werner.) **Post: #13**

Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

[updated a few error in indexes. Results and code remain the same]
Time for my reasoning and code ;-)

sum of 1/f(x) with f(x) = n*f(d(x)) with d(x) = number of bits to represent x in binary. f(1)=1 and f(2)=2, by definition.

To get the number of bits used by an integer I use:

```
LBL D
CLA
BINM
ARCL ST X
```

```
CLX
ALENG
EXITALL
RTN
```

which, on a 42S, gets the correct result over the range 1..2^36-1, which is far more than we'll need.

then, calling F with 1 x XEQ F will calculate f(x):

```
LBL F
3
X>Y?
GTO 00
Rv
STOx ST Y
XEQ D
GTO F
LBL 00
Rv
x
RTN
```

Let's write out a few values of f(x)

1 1
2 2
3 6
4 24 = 4*6
5 30 = 5*6
6 36 = 6*6
7 42 = 7*6
8 192 = 8*24
9 216 = 9*24
10 240 = 10*24
11 264 = 11*24
..

so the sum becomes

S = 1 + 1/2 + 1/6 + 1/24 + 1/30 + 1/36 + 1/42 + 1/192 + 1/216 + 1/240 + 1/264 + 1/288 + 1/312 + 1/336 + 1/360 + 1/480 + ..

Probably everyone will just have tried to sum up these numbers.. getting nowhere.
The numbers go down very slowly.. for instance, the millionth term is 1/6e8 - which in this case of course does not mean we have reached an accuracy of 1/6e8, as it is not an alternating series.
However, we can combine terms. Take the sequence

1/24 + 1/30 + 1/36 + 1/42 = 1/(4*6) + 1/(5*6) + 1/6*6) + 1/(7*6)

because all numbers 2^n till 2^(n+1)-1 are written with the same number of bits so

S = 1 + 1/2 + 1/6 + (1/4 + 1/5 + 1/6 + 1/7)/6 + (1/8 + 1/9 + .. 1/15)/24 + (1/16 + .. + 1/31)/30 + ..

with H(n) the nth Harmonic number = 1 + 1/2 + 1/3 + .. + 1/n

S = 1 + 1/2 + 1/6 + (H(7)-H(3))/F(3) + (H(15)-H(7))/F(4) + ..

so we can sum up (H(n-1)-H(n/2-1))/F(C) with n=2^C, but we are still stuck with basically the same convergence rate, or lack thereof.
However, for very large n, H(n-1)-H(n/2-1) = ln(2) to working precision. On Free42, this happens for n=2^111, on a real 42S it is n=2^38

So, we sum up the calculated (H(n-1)-H(n/2-1))/F(C), n=2^C, for C=3..K, where K+1 is such that H(n-1)-H(n/2-1) = ln(2), then we add the remainder multiplied by ln(2):

S = 1 + 1/2 + 1/6 + (H(7)-H(3))/F(3) + (H(15)-H(7))/F(4) + .. + (H(2^K-1)-H(2^(K-1)-1))/F(K)
+ LN(2)*(1/F(K+1) + 1/F(K+2) + ...)

if we add in LN(2)*(1/F(1) + 1/(F2) + .. 1/F(K) we get S again on the right side:

S = 5/3 + (H(7)-H(3))/(F(3) + (H(15)-H(7))/F(4) + (H(2^K-1)-H(2^(K-1)-1))/F(K)

- LN(2)*(1/F(1) + 1/F(2) + 1/F(3) + 1/F(4) + ..1/F(K))
+ LN(2)*S

or

S = (5/3 - 1.5*LN(2) + (H(7)-H(3)-LN(2))/F(3) + (H(15)-H(7)-LN(2))/F(4) + .. (H(2^K-1)-H(2^(K-1)-1)-LN(2))/F(K))/(1-LN(2));

To calculate the H(n):
- for small n we take the definition: H(n) = 1/n + 1/(n-1) + 1/(n-2) + .. 1/2 + 1
- larger n we take the approximation H(n) = ln(x + 1/(24*x + 3.7/x)) + gamma, x=n+0.5 and gamma = 0.577.. (thanks Albert!)
Since we only need H(n-1)-H(n/2-1), we don't need the Euler-Mascheroni constant.
For the 42S, n=128 results in a difference of 3e-14.
- H(2^K-1)-H(2^(K-1)-1)-LN(2) calculated with the approximation formula is LN(A)-LN(B)-LN(2) or LN(1 + (A-2B)/2B) as A and 2B grow ever closer
The factor 5/3 - 3/2*LN(2) I calculate as (10 - LN(512))/6 to avoid cancellation

I don't hardcode the value of K, but test when H(2^C-1)-H(2^(C-1)-1) = LN(2)

Running this on Free42 yields 2.0863776650059887.., on the 42S 2.08637766501 in 55 seconds.

```
00 { 187-Byte Prgm }
01▸LBL "VA3"
02 3
03 STO "C"
04 CLX
05 STO "S"
06▸LBL 10
07 RCL "C"
08 XEQ H
09 X=0?
10 GTO 00
11 1
12 RCL "C"
13 XEQ F
14 ÷
15 STO+ "S"
16 ISG "C"
17 X<>Y
18 GTO 10
19▸LBL 00
20 10
21 512
22 LN
23 -
24 6
25 ÷
26 RCL+ "S"
27 1
28 2
29 LN
30 -
31 ÷
32 RTN
33▸LBL D
34 CLA
35 BINM
36 ARCL ST X
37 CLX
38 ALENG
39 EXITALL
40 RTN
41▸LBL F
42 3
43 X>Y?
44 GTO 00
45 R↓
46 STO× ST Y
47 XEQ D
48 GTO F
```

```
49▸LBL 00
50 R↓
51 ×
52 RTN
53▸LBL H @ H(2^N-1) - H(2^(N-1)-1) - LN(2), input N
54 2
55 X<>Y
56 Y^X
57 128 @ 1E3 on Free42
58 X<Y?
59 GTO 00
60 CLX
61 2
62 -
63 0.05
64 %
65 +
66 1
67 + @ n-1,n/2-1
68 0
69▸LBL 02 @ 1/(n-1) + 1/(n-2) + .. + 1/(n/2)
70 RCL ST Y
71 IP
72 1/X
73 +
74 DSE ST Y
75 GTO 02
76 2
77 LN
78 -
79 RTN
80▸LBL 00
81 R↓
82 ENTER
83 XEQ 14
84 X<>Y
85 2
86 ÷
87 XEQ 14
88 STO+ ST X @
89 STO- ST Y
90 ÷
91 LN1+X
92 RTN
93▸LBL 14
94 0.5
95 -
96 3.7
97 RCL÷ ST Y
98 24
99 RCL× ST Z
100 +
101 1/X
102 +
103 END
```

Cheers, Werner

EMAIL PM FIND QUOTE REPORT

30th November, 2022, 16:16 (This post was last modified: 30th November, 2022 16:22 by PeterP.) **Post: #14**

PeterP
Member

Posts: 172
Joined: Jul 2015

RE: [VA] SRC #012c - Then and Now: Sum

Nicely done, Werner!

I got to the point where I had the sum on the left and the ln(2) times the partial sum on the right but I could not figure out the trick of adding the sum starting from 1 - K-1 back on the right side. Neat!

I used the in(ln2(x))+1 formula for the number of digits and a different approximation for Hn and but it has similar limits for accuracy. With a few tweaks your amazing code would fit the 41 as well, which I was shooting for.

PM FIND QUOTE REPORT

30th November, 2022, 18:06 **Post: #15**

Fernando del Rey
Junior Member

Posts: 19
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

Wow! I've been working on-and-off on this new problem, and can't get beyond the useless brute force approach.

I'm not being able to find a good trick to speed up convergence. I've tried to transform the series into an equivalent series with faster convergence, but so far I did not get anywhere.

I have not given up, however, I fear that this time I won't be among the winners circle

EMAIL PM FIND QUOTE REPORT

30th November, 2022, 21:01 **Post: #16**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012c - Then and Now: Sum

.
Hi, **FLISZT**, **Werner**, **J-F Garnier**, **John Keith** and **Fernando del Rey**,

A few assorted comments to what all of you said in your posts:

> **FLISZT Wrote:**
> Yes **I'm ready**. With the 9th Symphony of Bruckner broadcasted at the radio / Eugen Jochum / Bavarian Radio Orchestra (1954) and now the Concerto for Piano & Orch #3 / Robert Casadesus, New York Philh, Dimitri Mitropoulos (live, 1957)... **how could it not be the case?**!

Hehe, another classical music lover, like me. There was a time many years ago when I listened to classical music exclusively, nothing else appealed to me; it lasted for a few years, then I diversified again. By the way, I guess that your **FLISZT** alias honors Franz Liszt, amirite ?

> **FLISZT Wrote:**
> Here is my attempt:

Very nice, thank you for going ahead and posting it here. But that is just the *definition* of the recursive ***f(n***), you still need to provide the code which computes the ***sum*** ... ... and *btw*, thanks for editing that code panel out.

> **Werner Wrote:**
> Well, I have a result.. [...] I still have to double-check my code and my reasoning ;-) Execution is pretty fast (a blink of an eye on Free42, will run it on my 42S **when I'm reasonably certain** it is correct)

Ok, please post your code when you see fit but do try to *avoid spoiling* the fun for other people working on their own solutions right now. Oh, I see you've posted it already ...

> **Werner Wrote:**
> Time for **my reasoning and code ;-)** [...] To calculate the H(n) [...] **(thanks Albert!)**

Thanks a lot for your very detailed reasoning and nice **HP-42S** *RPN* code, **Werner**, your teaming with **Albert** (**Chan**, I suppose, who avoids posting here for whatever reason) has indeed delivered the goods: *code*, *result*, *timing*.

The only thing remaining is to ascertain that your result is *correct*, which will be most likely the case if someone else reproduces it using a wholly *different* program. We'll see ...

> **J-F Garnier Wrote:**
> Clearly brute force is not the way to go, but **I can't figure out a short-cut**.

Stick to it, please, I'm sure you'll eventually manage.

> **John Keith Wrote:**
> I wouldn't call **f(n)** an *"**unassuming** series"*, [...]

(tongue in cheek) *"Write a **sum**ptuous program to **sum**marily **sum** this unas**sum**ing yet serious series:"*

> **John Keith Wrote:**
> it's really quite a complex and interesting series. Being recursive **it can take some time to compute for larger numbers** [...]

Not really, unless the numbers are *humongously* large. For instance, *f(1,000,000,000)* needs only the initial call and 4 recursive calls, if I counted'em right.

> **John Keith Wrote:**
> This is still basically brute force and I don't have **a complete program** ready yet but at least **the germ of an idea**.

Good. *Do* persevere and post your completed program when ready, please. Remember: post *code (no panels)*, *sample run*, *result*, *timing* and *comments* if at all possible.

> **Fernando del Rey Wrote:**
> I've been working on-and-off on this new problem, and can't get beyond the useless brute force approach. [...] I've tried to transform the series into an equivalent series with faster convergence, but **so far I did not get anywhere. I have not given up**, however[...]

Don't despair, **Fernando**, it's just a matter of *perseverance* and lo and behold, the correct *"trick"* comes to mind in a flash. There's still plenty of time for you to produce a correct solution before I post my original one.

Well, thanks to all of you for your interest, you still have a number of days before I post my *6-line original solution*.

Best regards.
V.

PM WWW FIND EDIT QUOTE REPORT

30th November, 2022, 23:21 **Post: #17**

John Keith
Senior Member

Posts: 883
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

> **Valentin Albillo Wrote:** (30th November, 2022 21:01)
> .
>
> > **John Keith Wrote:**
> > it's really quite a complex and interesting series. Being recursive **it can take some time to compute for larger numbers** [...]
>
> **Not really**, unless the numbers are *humongously* large. For instance, *f(1,000,000,000)* needs only the initial call and 4 recursive calls, if I counted'em right.
>
> > **John Keith Wrote:**
> > This is still basically brute force and I don't have **a complete program** ready yet but at least **the germ of an idea**.
>
> Good. *Do* persevere and post your completed program when ready, please. Remember: post *code (no panels)*, *sample run*, *result*, *timing* and *comments* if at all possible.
>
> Best regards.
> **V.**

Unfortunately my program was just a brute force one and as many have found, brute force is a dead end for this challenge. After summing 2^17 terms and seeing no convergence, it was obvious that I was getting nowhere. I was never

anywhere near an analytical solution such as Werner's. Also as you mentioned, the recursion did not turn out to be a problem. It was an interesting and enjoyable challenge, though. Maybe next time...

EMAIL PM FIND QUOTE REPORT

1st December, 2022, 05:29 **Post: #18**

FLISZT
Junior Member

Posts: 43
Joined: Nov 2022

RE: [VA] SRC #012c - Then and Now: Sum

Hi All!

> **Valentin Albillo Wrote:** (30th November, 2022 21:01)
> Hehe, another classical music lover, like me. There was a time many years ago when I listened to classical music exclusively, nothing else appealed to me; it lasted for a few years, then I diversified again. By the way, I guess that your FLISZT alias honors Franz Liszt, amirite ?

Yes, rather a "mélomane" in general and a classical music lover in particular. I have two old souvenirs with classical music but out of topic. I have listened to different kinds of music. But unlike you, for the last few years I have been listening almost exclusively to classical music (symphonies, symphonic poems... and pieces created for the piano.)
Indeed, I like Liszt, but I could have choosen an other name like "Saint-Saëns" which sounds very "français" (so many composers created masterpieces) or "Franz Lisp" (I read that this language did exist!) to be more connected to the forum.

> **Valentin Albillo Wrote:** (30th November, 2022 21:01)
> But that is just the definition of the recursive f(n)

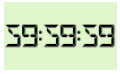
Yes, I realized my mistake later... and so I understood why I was the 1st to publish a code when there are so many math and programming gurus in this forum!
A bit like thinking to be ahead in a F1 Grand-Prix with an old 2CV... but already being almost one lap late, 1 min after departure. 😀😀
When I "draw" a calculator, it's basically only to check my accounts or to try this kind of "game". So I'm going to search with the "help" of my old and few skills in math.

> **Valentin Albillo Wrote:** (30th November, 2022 21:01)
> .. and *btw*, thanks for editing that code panel out.

Don't mention it!
About the code panels, I will say that it's a pity that you have to scroll as soon as the code is a bit long.
On the other hand, the layout is much more adapted to a structured language like RPL. Allowing small code panels (with no scrolling) could be a solution... or not. Just a thought.

EMAIL PM FIND QUOTE REPORT

1st December, 2022, 09:10 **Post: #19**



Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

> **Valentin Albillo Wrote:** (30th November, 2022 21:01)
> Ok, please post your code when you see fit but do try to *avoid spoiling* the fun for other people working on their own solutions right now. Oh, I see you've posted it already ...

Apologies for jumping the gun! But I did wait for a full day..

> **Quote:**
> Thanks a lot for your very detailed reasoning and nice **HP-42S** *RPN* code, **Werner**, your teaming with **Albert** has indeed delivered the goods: *code*, *result*, *timing*.

I didn't team up with Albert, I just used his formula for H(n), which is shorter and more accurate than the one you find elsewhere, and which he seems to be able to produce off the cuff. (Though in this case, as he pointed out to me in the meantime, using the original formula with a few more terms would've been better as it contains LN(2), which would then be cancelled out)

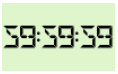
> **Quote:**
> The only thing remaining is to ascertain that your result is *correct*, which will be most likely the case if someone else reproduces it using a wholly *different* program. We'll see ... 😀

Yes, well. The trick I used only works if the sum is convergent to begin with, something I haven't been able to prove.

Cheers, Werner

EMAIL PM FIND QUOTE REPORT

1st December, 2022, 10:27 (This post was last modified: 1st December, 2022 10:33 by Werner.) **Post: #20**



Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

Using the original asymptotic series H(n) = ln(n) + gamma + 1/(2*n) - 1/(12*n^2) + 1/(120*n^4) - ... results in:
H(n-1)-H(n/2-1) - ln(2) = 1/(2n) + 1/(4n^2) - 1/(8n^4) + 1/(4n^6) - 17/(16n^8) + .. (thanks^2, Albert!).
This formula is so accurate that my stopping criterion needed to be changed to 1+x = 1 ;-) But now, I need the definition only for n<32 instead of 128, and the running time on a real 42S went down to 36 seconds.

```
00 { 180-Byte Prgm }
01▸LBL "VA3"
02 3
03 STO "C"
04 CLX
05 STO "S"
06▸LBL 10
07 RCL "C"
08 XEQ H
09 1
10 ENTER
11 RCL+ ST Z
12 X=Y?
13 GTO 00
14 R↓ @ X contains 1
15 RCL "C"
16 XEQ F
17 ÷
18 STO+ "S"
19 ISG "C"
20 X<>Y
21 GTO 10
22▸LBL 00
23 10
24 512
25 LN
26 -
27 6
28 ÷
29 RCL+ "S"
30 1
31 2
32 LN
33 -
34 ÷
35 RTN
36▸LBL D
37 CLA
38 BINM
39 ARCL ST X
40 CLX
41 ALENG
42 EXITALL
43 RTN
44▸LBL F
45 3
46 X>Y?
47 GTO 00
48 R↓
49 STO× ST Y
50 XEQ D
51 GTO F
52▸LBL 00
53 R↓
```

```
54 ×
55 RTN
56▸LBL H @ H(2^N-1) - H(2^(N-1)-1) - LN(2), input N
57 2
58 X<>Y
59 Y^X
60 32 @ 128 on Free42 gives 18 digits of accuracy
61 X≤Y?
62 GTO 00
63 CLX
64 2
65 -
66 0.05
67 %
68 +
69 1
70 + @ n-1,n/2-1
71 0
72▸LBL 02 @ 1/(n-1) + 1/(n-2) + .. + 1/(n/2)
73 RCL ST Y
74 IP
75 1/X
76 +
77 DSE ST Y
78 GTO 02
79 2
80 LN
81 -
82 RTN
83▸LBL 00 @ H(n-1) - H(n/2-1) - ln(2) ~= 1/(2n) + 1/(4n^2) - 1/(8n^4) + 1/(4n^6) - 17/(16n^8)
84 R↓
85 STO+ ST X
86 X^2
87 LASTX
88 1/X
89 272
90 RCL÷ ST Z
91 16
92 -
93 R^
94 ÷
95 2
96 +
97 R^
98 ÷
99 1
100 -
101 R^
102 ÷
103 -
104 END
```

Cheers, Werner

EMAIL PM FIND QUOTE REPORT

1st December, 2022, 11:42 (This post was last modified: 1st December, 2022 23:12 by J-F Garnier.) **Post: #21**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

> **Valentin Albillo Wrote:** (30th November, 2022 21:01)
>
> > **J-F Garnier Wrote:**
> >
> > Clearly brute force is not the way to go, but **I can't figure out a short-cut**.
>
> Stick to it, please, I'm sure you'll eventually manage.

The missing point for me was that there are efficient approximations of the Harmonic numbers, and that we can easily find the point when the partial sum can be reduced to log(2).
Now I think I have an idea to calculate the sum in a different (but maybe equivalent) way than Werner, and my

understanding is that it will also prove that the sum is convergent.

Edit: **Result 2.08637766501 confirmed** I will post my HP-71B program and comments soon...

J-F

EMAIL PM WWW FIND QUOTE REPORT

2nd December, 2022, 09:26 **Post: #22**

Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

I found another, similar way, but I fear it's the same as J-F's, so I will hold out till he posted his solution ;-)
Werner

EMAIL PM FIND QUOTE REPORT

3rd December, 2022, 03:28 **Post: #23**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012c - Then and Now: Sum

Hi, all,

A **last round** of comments before I post *my original solution* next **Sunday** night:

> **John Keith Wrote:**
> Unfortunately my program was just a brute force one and as many have found, **brute force is a dead end** for this challenge.

Indeed.

> **John Keith Wrote:**
> After **summing 2^17 terms** and seeing no convergence, it was obvious that I was getting nowhere.

It would need summing much more terms than that to get just *one* roughly correct digit.

> **Quote:**
> I was never anywhere near an analytical solution such as Werner's. Also as you mentioned, the recursion did not turn out to be a problem. **It was an interesting and enjoyable challenge**, though. **Maybe next time**...

Glad you liked it. The analytics aren't that difficult at all, it's just getting the correct idea. Thank you very much for your interest and for doing your best to solve it. Maybe next time, though *Problems 4, 5* and *6* are allegedly more difficult *(though still solvable using an **HP-71B**) ... or not !*

> **FLISZT Wrote:**
> About the **code panels**, I will say that it's a pity that you have to scroll as soon as the code is a bit long.

For me, the problem with **CODE** panels is that when I create a *PDF* with the thread (which I always do to upload it to my site), the code within them gets *truncated* and so becomes useless. Thanks for your participation in this thread, Bruno.

> **Werner Wrote:**
> Apologies for jumping the gun! But **I did wait for a full day**..

Most commendable. Well, someone had to be first, it might as well be you.

> **Werner Wrote:**
> Yes, well. The trick I used only works **if the sum is convergent** to begin with, something **I haven't been able to prove**.

You can trust me, it ***is*** a *convergent* series and it's not that hard to prove, surely **Albert** *(Chan ?)* can provide you with

one proof or seven, else I'll obligue.

> **Werner Wrote:**
> Using the original asymptotic series [...] **(thanks^2, Albert!)** [...] now, I need the definition only for n<32 instead of 128, and the running time on a *real 42S* went down to ***36*** *seconds*.

Excellent run time indeed, you should've posted *"thanks^**3**"* to **Albert** instead, and when he provides you with the convergence proof you can up it to *"thanks^**4**"*

> **J-F Garnier Wrote:**
> Now **I think I have an idea to calculate the sum** in a different (but maybe equivalent) way than Werner, and my understanding is that **it will also prove that the sum is convergent**.

Good. See ? *Perseveration* was the key and eventually you did manage, as I said you would.

> **J-F Garnier Wrote:**
> Edit: Result **2.08637766501 confirmed**. I will post my HP-71B program and comments soon...

Perfect. I can confirm that the result is indeed correct to 12 digits. Eagerly waiting for you to post your **HP-71B** program and comments. You have till *next **Sunday** night*.

> **Werner Wrote:**
> I found **another**, similar way, but **I fear it's the same as J-F's**, so I will hold out till he posted his solution ;-)

Most considerate of you, **J-F** will surely be most pleased.

V.

PM WWW FIND EDIT QUOTE REPORT

3rd December, 2022, 10:54 (This post was last modified: 3rd December, 2022 15:18 by J-F Garnier.) **Post: #24**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

Here is my "solution", based on Werner's reasoning, so most credit due to him:-)

I restarted from the step below, changing the K+1 limit (the point from where we use ln(2) as an approximation of the partial sum) to K. It's a just a notation:

S = 1 + 1/2 + 1/6 + (H(7)-H(3))/F(3) + (H(15)-H(7))/F(4) + .. + (H(2^(K-1)-1)-H(2^(K-2)-1))/F(K-1)
+ LN(2)*(1/F(K) + 1/F(K+1) + ...)

Here we can do to the last part S3 = (1/F(K) + 1/F(K+1) + ...) what we already did for the first part of the sum,
i.e. writing F(K) = K.F(d) with d=number of binary digits of K.
If we choose K=2^(M-1), we have F(K) = K.F(M)

S3 = (H(2^M-1)-H(2^(M-1)-1)/F(M) + ... + (H(2^(K-1)-1)-H(2^(K-2)-1))/F(K-1)
+ LN(2)*(1/F(K) + 1/F(K+1) + ...)

And here we recognize that we can do the same thing again on the last (1/F(K) + 1/F(K+1) + ...) part, and again and again.
Also the quantity S2= (H(2^M-1)-H(2^(M-1)-1)/F(M) + ... + (H(2^(K-1)-1)-H(2^(K-2)-1))/F(K-1)
has already been computed as part of the beginning of the sum.

So we end with:
S = 1 + 1/2 + 1/6 + (H(7)-H(3))/F(3) + .. + (H(2^(M-1)-1)-H(2^(M-2)-1))/F(M-1) + S2
+ LN(2) * (S2 + LN(2) * (S2 + LN(2) * (S2 ...

The limit Sx of the quantity (S2 + LN(2) * (S2 + LN(2) * (S2 + LN(2) * S2 ...) is finite and is such as S2 + LN(2) * Sx = Sx
so Sx = S2 / (1 - LN(2))

To compute the whole sum S:
compute S1 = 1 + 1/2 + 1/6 + (H(7)-H(3))/F(3) + .. + (H(2^(M-1)-1)-H(2^(M-2)-1))/F(M-1)
compute S2 = (H(2^M-1)-H(2^(M-1)-1)/F(M) + ... + (H(2^(K-1)-1)-H(2^(K-2)-1))/F(K-1)
compute S = S1 + S2 / (1 - LN(2))

M is chosen such as 2^K=2^(2^(M-1)) >> 1E12 to ensure the partial sum H(2^(K-1)-1)-H(2^(K-2)-1) is accurately represented by ln(2) :
I used M=7 so K=2^6=64.

```
10 ! SRC12C3
20  ! HP-71 / HP-75 version
30  L=63
40  DIM F(63)
50  !
60  L2=LOG(2) ! used several times
70  ! calculate the F(I)
80  F(1)=1 @ F(2)=2
90  FOR I=3 TO L @ D=INT(LOG(I)/L2)+1 @ F(I)=I*F(D) @ NEXT I
100 !
110 ! compute the sum S0 for K=1..2
120 S0=1/F(1)+1/F(2)+1/F(3)
130 ! compute the sum S1 for K=3..6
140 S1=0
150 FOR K=3 TO 6
160   J1=2^(K-1) @ J2=2*J1-1
170   X=0 @ FOR J=J1 TO J2 @ X=X+1/J @ NEXT J
180   S1=S1+X/F(K)
190 NEXT K
210 ! now compute the sum S2 for K=7..40 using the H approx
230 S2=0
240 FOR K=K TO 40
250   N=2^(K+1) @ N2=N*N
260   X=((((-272/N2+16)/N2-2)/N2+1)/N+1)/N+L2
270   S2=S2+X/F(K)
280 NEXT K
290 ! and complete the sum S2 up to K=L using the LOG(2) approx
310 FOR K=K TO L @ S2=S2+L2/F(K) @ NEXT K
320 !
330 ! now compute the final sum S
340 S=S2/(1-LOG(2))+S1+S0
350 DISP S
```

Result = 2.08637766501
HP-75D: 9.3s
HP-71B: 13.2s

J-F

EMAIL PM WWW FIND QUOTE REPORT

3rd December, 2022, 14:15 **Post: #25**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012c - Then and Now: Sum

Sorry about math sessions. I will keep it short.

Let F(n) = n * F(bits of n), except that F(2)=2, F(1)=1
Let G(n) = sum of n-bits integer reciprocal, except that G(1)=5/4

Let sum = index from 1 to K-1, SUM = index K to infinity

S = sum(1/F) + SUM(1/F) // definition
S = sum(G/F) + SUM(G/F) // sum(G/F) accelerated convergence, but still not fast enough

S = sum(G/F) + LN2 * SUM(G/LN2/F)

Since G/LN2 ≥ 1, no matter how big K is, we have:

S ≥ sum(G/F) + LN2 * SUM(1/F)
S ≥ sum(G/F) + LN2 * (S - sum(1/F))

S*(1-LN2) ≥ sum((G-LN2)/F)

No need to hard code conditions for index K, or for G converged to LN2.
Just sum RHS until convergence. It will converge, very quickly.

(G-LN2) part shrink at O(1/2^n), which already can converge without F

EMAIL PM FIND QUOTE REPORT

3rd December, 2022, 14:59 **Post: #26**

PeterP
Member

Posts: 172
Joined: Jul 2015

RE: [VA] SRC #012c - Then and Now: Sum

Ok, I was able to get it into the 41 with the below code.
As mentioned earlier I had gotten to the equation with ln(2) on the right side but needed Werner's trick (Thank you!) for getting it solved.
Based on HP41 accuracy, I have split the calculation into three segments:
1) Straight Forward calculation of f(x)
2) Using the approximation H(n) ~ ln(n) + 1/(2n) - 1/(12n^2) + 1/(120n^4)
3) Using the approximation that H(n) - H(n-1) ~ Ln(2)

On the hp 41, the approximation 2) is accurate within the precision of the HP41 from n = 2^5 onwards.

On the hp41, the approximation 3) is accurate within the precision of he HP41 from n = 2^31 onwards.

So I am calculating a straight sum from n = 1 till 2^5-1, then switch to the approximation sum 2) from then onwards until 2^31, after which I use ln(2).

I looked at the approximations and I do believe that they prove that the series converges.

I dont know how to do the nice math font here, so my appologies for the below.

S = sum Valentin asked us to calculate.
f(n) = function that valentin gave us
g(x) = H(2^x-1) - H(2^(x-1)-1)
with H(x) being the Harmonic Series up to x
ap(x) = ln(x) + 1/(2x) - 1/(12x^2) + 1/(120x^4)

S = sum(n = 1 to 2^m-1) of f(n)^-1 + sum(x = m+1 to infinity) of f(x)^-1*g(x)
replacing g(x) with the approximation we note that the approximation is always larger than g(x) as we stop with an addition.

S<=sum(n = 1 to 2^m-1) of f(n)^-1 + sum(x = m+1 to infinity) of f(x)^-1* (ap(2^x-1)-ap(2^(x-1)-1)

We then replace the ap() on the right side with ln(2) after a certain cut off point, noting that Ln(2) is also larger than ap()

S<= sum(n = 1 to 2^m-1) of f(n)^-1 + sum(x = m+1 to p-1) of f(x)^-1* (ap(2^x-1)-ap(2^(x-1)-1) + ln(2) * sum(y=p to infinity) of f(y)^-1

with an infinite sum on the right side, this inequality can only be true if S converges.

Or at least this is how my thinking went.

Runtime on the i41cx emulator is a few seconds, result it produces is 2,088075017. Which means that my assumptions about the correct cross-over points are not correct and I might be able to squeeze out a slightly better result by choosing later cross over points but I am not entirely sure how/why, as the calc can't differentiate between ln(2) and the approximation and the approximation and H(2x-1) - H(x-1) anymore at my current cut off points.

However, my flight has landed and I had given up on this when reading it but then had nothing to do on my flight over (europe to US) and made some progress, then got Werners tip, and was able to finish it on the way back. So I feel pretty happy, and thankful.

Here is the code.
Lbl F calculates Valentin's function
Lbl G calculates the approximation of H

Lbl VA12c
CLA

```
CLX
STO 10
STO 11
STO 12
2
STO 02
31
STO 00
LBL 00
RCL 00
XEQ F
ST+10
DSE00
GTO 00
31.005
STO 00
LBL 01
RCL 00
INT
XEQ G
STO 11
RCL 00
INT
XEQ F
RCL 11
*
ST+ 12
DSE 00
GTO 01
RCL 02
ln
SIGN
LastX
-
1/x
RCL 12
*
RCL 10
+
BEEP
STOP
LBL F
SIGN
STO M
X<>L
LBL 10
3
x>y?
GTO 12
x<>y
ST*M
LN
RCL 02
LN
/
INT
INCX
GTO 10
LBL 12
X<>Y
RCL M
*
1/x
RTN
LBL G
RCL 02
X<>Y
Y^X
DECX
STO N
RCL 02
```

```
LASTx
DECX
Y^X
DECX
STO O
LN
LASTx
RCL 02
*
1/x
+
RCLO
X^2
12
*
1/x
-
RCL O
X^2
x^2
120
*
1/x
+
RCL N
LN
LASTx
RCL 02
*
1/x
+
RCL N
X^2
12
*
1/x
-
RCL N
X^2
X^2
120
*
1/x
+
X<>Y
-
RTN
```

PM FIND QUOTE REPORT

3rd December, 2022, 18:39 **Post: #27**

ijabbott
Senior Member

Posts: 1,228
Joined: Jul 2015

RE: [VA] SRC #012c - Then and Now: Sum

> **Valentin Albillo Wrote:** (3rd December, 2022 03:28)
>
> > **FLISZT Wrote:**
> >
> > About the **code panels**, I will say that it's a pity that you have to scroll as soon as the code is a bit long.
>
> For me, the problem with **CODE** panels is that when I create a *PDF* with the thread (which I always do to upload it to my site), the code within them gets *truncated* and so becomes useless. Thanks for your participation in this thread, Bruno.

Just on that point, the "printable" version of threads does expand the CODE panels, and so does the Lite (Archive) view. They would make your PDF files smaller too! It would be nice if the forum had an option to show the entire thread on one page as that would be even easier to convert to PDF.

EMAIL PM FIND QUOTE REPORT

3rd December, 2022, 19:30 **Post: #28**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012c - Then and Now: Sum

> **Albert Chan Wrote:** (3rd December, 2022 14:15)
> Let F(n) = n * F(bits of n), except that F(2)=2, F(1)=1
> Let G(n) = sum of n-bits integer reciprocal, except that G(1)=5/4

Implementation details, I do not define F(1) or G(1) for simplicity.
Loops sum Z=(G-LN2)/F, from index of 2, until convergence.

(G(1)-LN2)/F(1) = (5/4-LN2)/1 = 1/4 + (1-LN2)

```
10 DESTROY ALL @ L2=LN(2) @ SETTIME 0
20 DEF FNB(N)=IP(LN(N+.5)/L2)+1 ! BITS OF INTEGER N
30 DEF FNF(N) @ F=2 @ WHILE N>2 @ F=F*N @ N=FNB(N) @ END WHILE @ FNF=F @ END DEF
40 DEF FNG(N) @ G=0 @ N=2^N-1 @ Y=N*(N-1) ! SUM IN PAIRS
50 FOR X=N+N-1 TO N STEP -4 @ G=G+X/Y @ Y=Y-X-X+4 @ NEXT X @ FNG=G @ END DEF
60 DEF FNZ(N) @ IF N<5 THEN Z=FNG(N)-L2 @ GOTO 80
70 Z=.5^(N+1) @ Z2=Z*Z @ Z=(((-272*Z2+16)*Z2-2)*Z2+1)*Z2+Z
80 FNZ=Z/FNF(N) @ END DEF
100 S=1/4 @ I=1 @ REPEAT @ I=I+1 @ P=S @ S=S+FNZ(I) @ UNTIL P=S
110 DISP S/(1-L2)+1,I,TIME

>run
 2.086377665     31      0.1
```

Emu/DOS WinXP ≈ 200X --> HP71B runtime about 20 seconds.

EMAIL PM FIND QUOTE REPORT

3rd December, 2022, 21:16 (This post was last modified: 3rd December, 2022 21:26 by Valentin Albillo.) **Post: #29**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012c - Then and Now: Sum

.
Hi, **ijabbot**,

> **Quote:**
> Just on that point, **the "printable" version of threads does expand the CODE panels**, and so does the Lite (Archive) view. They would make your PDF files smaller too!

Yes, I know that the *Printable* version does expand the CODE panels but last time I checked, the *MathJax* (or whatever the name is) mathematical expressions do not appear as properly formated textbook expressions but as the *MathJax* text, which of course defeats the purpose entirely and looks horrible.

In short, when converted to *PDF*:

- *"normal"* version gives you nice math expressions but truncated, useless *CODE* panels.

- *"printable"* version does expand the code panels but gives you text *MathJax* instead of properly formated (graphical) math expressions.

If you know some way to have ***both*** (i.e. nice math expression *and* untruncated *CODE* panels) I'd be very obligued. 🙂

> **Quote:**
> It would be nice if the forum had an option to **show the entire thread on one page** as that would be even easier to convert to PDF.

Yep, it would be ideal, at least for me. The **old forum** did just that, the whole thread in a single page, no matter how many posts in it, which was heaven for conversion to *PDF*.

I've increased the number of posts per page to the maximum, **50**, so that if the thread results in 49 replies posted or less then it'll fit in just one page, i.e., one *PDF* file. Alas, my previous *SRC* had *86* posts in all so two pages, two *PDF*.

If **Mr. Hicks** would increase the limit of posts per page to **100** instead of **50** the situation would improve greatly.

Thanks for trying to help, have a nice weekend.
Regards.
V.
Edit: Oh, and "Lite version" does this:
Example of *Lite* version with math expressions, images, and *CODE* panels.
.

PM WWW FIND EDIT QUOTE REPORT

3rd December, 2022, 21:41 **Post: #30**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

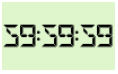
> **Albert Chan Wrote:** (3rd December, 2022 19:30)
>
> 10 DESTROY ALL @ L2=LN(2) @ SETTIME 0
> 20 DEF FNB(N)=IP(LN(N+.5)/L2)+1 ! BITS OF INTEGER N
> 30 DEF FNF(N) @ F=2 @ WHILE N>2 @ F=F*N @ N=FNB(N) @ END WHILE @ FNF=F @ END DEF
> 40 DEF FNG(N) @ G=0 @ N=2^N-1 @ Y=N*(N-1) ! SUM IN PAIRS
> 50 FOR X=N+N-1 TO N STEP -4 @ G=G+X/Y @ Y=Y-X-X+4 @ NEXT X @ FNG=G @ END DEF
> 60 DEF FNZ(N) @ IF N<5 THEN Z=FNG(N)-L2 @ GOTO 80
> 70 Z=.5^(N+1) @ Z2=Z*Z @ Z=(((-272*Z2+16)*Z2-2)*Z2+1)*Z2+Z
> 80 FNZ=Z/FNF(N) @ END DEF
> 100 S=**1/4** @ I=1 @ REPEAT @ I=I+1 @ P=S @ S=S+FNZ(I) @ UNTIL P=S
> 110 DISP S/(1-L2)**+1**,I,TIME
>
> >run
> 2.086377665 31 0.1
>
> Emu/DOS WinXP ≈ 200X --> HP71B runtime about 20 seconds.

Runs in 19.9s on a real HP-71B :-)

J-F

EMAIL PM WWW FIND QUOTE REPORT

4th December, 2022, 10:49 **Post: #31**



Werner
Senior Member

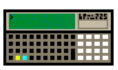
Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

Questions!
- Why does Albert's code run so much slower than J-F's, while doing 31 loops vs 40?
- Why is a 71B still so much faster than a 42S for similar programs, having even a slower CPU? (My code is down to 31 secs, and is more like Albert's than like J-F's, doing 33 loops).
Cheers, Werner

EMAIL PM FIND QUOTE REPORT

4th December, 2022, 11:50 (This post was last modified: 4th December, 2022 12:41 by J-F Garnier.) **Post: #32**



J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

> **Werner Wrote:** (4th December, 2022 10:49)
>
> Questions!
> - Why does Albert's code run so much slower than J-F's, while doing 31 loops vs 40?
> - Why is a 71B still so much faster than a 42S for similar programs, having even a slower CPU? (My code is down to 31 secs, and is more like Albert's than like J-F's, doing 33 loops).
> Cheers, Werner

Probably a bit OT question, but especially interesting for me!

- I don't fully understand Albert's code, so I'm not sure if his iterative algorithm can be compared to mine/yours.
But there may be at least two reasons that contribute to slower speed:
use of a user function to get f(n) whereas they are precalculated for me,
and FNx user functions overhead.

- for the relative speed of the 42S and 71B, the 42S indeed is not a fast machine due to the System RPL overhead.
The 32S in pure assembly code (as the 71B with similar clock speed, so comparable), is actually faster than the 42S, this is one of the reasons I prefer it to the 42S. The 32SII is unfortunately not as fast as the 32S, for reasons that are really OT here.
From here, relative execution times (the shorter, the better):
- 4:22 HP-32S Keystroke / RPN
- 5:03 HP-32SII Keystroke / RPN / Ver.0
- 12:12 HP-42S Keystroke / RPN / Ver.C

J-F

EMAIL PM WWW FIND QUOTE REPORT

4th December, 2022, 11:54 (This post was last modified: 4th December, 2022 12:42 by Albert Chan.) **Post: #33**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012c - Then and Now: Sum

> **Werner Wrote:** (4th December, 2022 10:49)
> - Why does Albert's code run so much slower than J-F's, while doing 31 loops vs 40?

```
> 20 DIM F(63) @ F(2)=2 @ F(3)=6 @ B=4
> 30 FOR X=3 TO 6 @ FOR Y=B TO B+B-1 @ F(Y)=Y*F(X) @ NEXT Y @ B=B+B @ NEXT X
> 80 FNZ=Z/F(N) @ END DEF
>RUN
 2.086377665      31      0.05
```

Above patch removed FNB(N) and FNF(N), and build list of F(N) instead.

F(N=63) is enough, since (G-LN2) shrink at O(1/2^n)
For 12 decimal digits, we need at most N=12/LOG10(2) ≈ 40
Z = (G-LN2)/F, with F growing faster than primes, O(n*ln(n)), it needed even less than that.

Cached F doubled program speed (translated to HP71B runtime of about 10s)

EMAIL PM FIND QUOTE REPORT

4th December, 2022, 12:16 **Post: #34**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012c - Then and Now: Sum

Hi, J-F Garnier

> **J-F Garnier Wrote:** (3rd December, 2022 10:54)
> To compute the whole sum S:
> compute S1 = 1 + 1/2 + 1/6 + (H(7)-H(3))/F(3) + .. + (H(2^(M-1)-1)-H(2^(M-2)-1))/F(M-1)
> compute S2 = (H(2^M-1)-H(2^(M-1)-1)/F(M) + ... + (H(2^(K-1)-1)-H(2^(K-2)-1))/F(K-1)
> compute S = S1 + S2 / (1 - LN(2))

> **Albert Chan Wrote:** (3rd December, 2022 14:15)
> Let sum = index from 1 to K-1, SUM = index K to infinity
>
> S = sum(1/F) + SUM(1/F) // definition
> S = sum(G/F) + SUM(G/F) // sum(G/F) accelerated convergence, but still not fast enough
> ...
> S*(1-LN2) ≥ sum((G-LN2)/F)

Both are exactly the same, except mine start from 1, yours start from M ≠ 1
If we re-define sum = index from M to K-1, we have:

(S-S1)*(1-LN2) ≥ sum((G-LN2)/F) = S2

EMAIL PM FIND QUOTE REPORT

4th December, 2022, 13:58 **Post: #35**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

> **Albert Chan Wrote:** (4th December, 2022 12:16)
> Both are exactly the same, except mine start from 1, yours start from M ≠ 1
> If we re-define sum = index from M to K-1, we have:
>
> (S-S1)*(1-LN2) ≥ sum((G-LN2)/F) = S2

> **Albert Chan Wrote:** (3rd December, 2022 14:15)
> S = sum(G/F) + LN2 * SUM(G/LN2/F)
>
> Since G/LN2 ≥ 1, no matter how big K is, we have:
>
> S ≥ sum(G/F) + LN2 * SUM(1/F)
> S ≥ sum(G/F) + LN2 * (S - sum(1/F))
>
> S*(1-LN2) ≥ sum((G-LN2)/F)
>
> No need to hard code conditions for index K, or for G converged to LN2.
> Just sum RHS until convergence. It will converge, very quickly.

What I don't understand in your reasoning is how you moved from:
S*(1-LN2) ≥ sum((G-LN2)/F)
to
S*(1-LN2) = sum((G-LN2)/F) when the sum has converged.

J-F
(not a mathematician)

EMAIL PM WWW FIND QUOTE REPORT

4th December, 2022, 15:19 **Post: #36**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012c - Then and Now: Sum

> **J-F Garnier Wrote:** (4th December, 2022 13:58)
> What I don't understand in your reasoning is how you moved from:
> S*(1-LN2) ≥ sum((G-LN2)/F)
> to
> S*(1-LN2) = sum((G-LN2)/F) when the sum has converged.

sum index is from 1 to K-1, but we never specify what K is.
If RHS converge, it meant K can be as big as we wanted (literally, K = infinity)

Another way to look at this:

Last dropped term contributed ≤ 1/2 ULP (definition of "converged")
Because of (G-LN2) shrink rate of O(1/2^n), next dropped term ≤ 1/4 ULP, next ≤ 1/8 ULP, ...

Adding effect of growing F, maximum sum of all missing terms will never contribute 1 ULP.
Numerically, we can turn inequality into equality.

EMAIL PM FIND QUOTE REPORT

4th December, 2022, 22:44 **Post: #37**

PeterP
Member

Posts: 172
Joined: Jul 2015

RE: [VA] SRC #012c - Then and Now: Sum

Interesting improvement to the HP41 result:

At least on the emulator, the accuracy of ln is not good enough to calculate the number of digits in the straight forward fashion of ln(x)/ln(2). For example, for 16, this results in 3.9999999999, rather than 4.

Adding an ulp to it before taking the int fixes the problem, after which my code delivers the expected 2.086377665.

PM FIND QUOTE REPORT

4th December, 2022, 23:34 (This post was last modified: 7th December, 2022 01:21 by Valentin Albillo.) **Post: #38**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012c - Then and Now: Sum

Hi, all,

Well, a full week has elapsed since I posted my *OP*, which has already passed the *~1,700 views* mark (as expected, when the difficulty increases the number of posts and/or views decreases,) and I've got a number of solutions and/or comments, namely by ***Werner***, ***J-F Garnier***, ***FLISZT***, ***ThomasF***, ***PeterP***, ***Fernando del Rey***, ***John Keith*** and ***Albert Chan***. Thank you very much to all of you for your interest and valuable contributions.

Now I'll provide my detailed ***original solution*** to this ***Problem 3*** but first a couple' comments:

> ***1)*** As I said in my *OP*, I quote:
>
> > *"Some useful advice is to try and find the correct balance between the program doing all the work with no help from you [...] or else **using some insight** to help speed up the process."*
>
> and in this case using sheer brute force is utterly *hopeless*. As will be seen below, naïvely adding up terms of the series goes nowhere; matter of fact, after summing more than *2. 10^{90} terms* we get not even *one* roughly correct digit, never mind 10-12, so this time the programmer has indeed to use ***some insight*** to succeed.
>
> ***2)*** I expected everyone to first try adding up terms from the series, but after seeing that doing so led nowhere I also expected you would then try some *acceleration methods* and/or *extrapolation methods* but no one actually did (myself included) !

That said, this is my detailed *sleuthing process* and resulting *original solution:*

My sleuthing process

First of all, I realized that ***d(n)*** = *number of binary digits of n* doesn't need any conversions to base 2 and binary-digit counting because we simply have ***d(n)* = IP(LOG2(N)+1)**, which requires an *accurate* **LOG2** function (like the one provided by the **HP-71B *Math ROM***), lest the **IP** could be *off* by one. Naïvely using **LN(N)/LN(2)** will occasionally *fail*, even in ***Free42*** *Decimal*, despite its ***34****-digit* accuracy.

Knowing that, I then wrote a bit of sleuthing *HP-71B* code to generate the first 16 terms of the series and then to add up to *100*, *1,000*, *10,000* and *100,000* terms:

```
10 DEF FNF(N) @ IF N<3 THEN FNF=N ELSE FNF=N*FNF(IP(LOG2(N)+1))
20 END DEF
30 DESTROY ALL @ FOR N=1 TO 16 @ DISP FNF(N); @ NEXT N @ DISP
40 FOR K=2 TO 5 @ N=10^K @ S=0 @ FOR I=1 TO N @ S=S+1/FNF(I) @ NEXT I @ DISP N;S @ NEXT K

>RUN

    1 2 6 24 30 36 42 192 216 240 264 288 312 336 360 480

    100       1.87752
    1000      1.89281
    10000     1.90075
    100000    1.90642
```

which told me that the series looked like some kind of *"weakened"* ***harmonic series***, turned convergent but with such a slow convergence rate that finding its value by adding terms was bound to be hopeless, as the sum didn't seem to converge fast enough even when adding up an *exponentially increasing* number of terms.

Now, computing a few ***d(n)*** values, the series looks like this *(**Sum 1**)*:

$$S = \frac{1}{f(1)} + \frac{1}{f(2)} + \frac{1}{f(3)} + \frac{1}{4\,f(3)} + \frac{1}{5\,f(3)} + \frac{1}{6\,f(3)} + \frac{1}{7\,f(3)} + \frac{1}{8\,f(4)} + \frac{1}{9\,f(4)} \cdots$$

and we notice that the value of ***d(n)*** stays the same between powers of 2, as for *n = 4* to *7* we have *d(n) = 3*, then for *n = 8* to *15* we have *d(n) = 4*, etc. Also, the denominators have factors *4*, *5*, *6*, *7*, ..., which reminds me of the **harmonic numbers**:

$$H(k) = 1 + \tfrac{1}{2} + \tfrac{1}{3} + \tfrac{1}{4} + \cdots + \tfrac{1}{k}$$

and using them the series can now be expressed like this *(**Sum 2**)*:

$$S = 1 + \frac{1}{2} + \frac{1}{6} + \frac{1}{f(3)}\Big(H(7) - H(3)\Big) + \frac{1}{f(4)}\Big(H(15) - H(7)\Big) + \cdots$$

$$= \frac{5}{3} + \sum_{n=3}^{\infty} \frac{H(2^n - 1) - H(2^{n-1} - 1)}{f(n)}$$

so we need a quick way to compute ***H(k)***, which for small values of ***k*** can be done by just summing ***k*** terms of its definition, and for large values of ***k*** can be done using its ***asymptotic series expansion***:

$$H(k) \sim \ln(k) + \gamma + \frac{1}{2k} - \frac{1}{12k^2} + \frac{1}{120k^4} + \cdots$$

which has an error less than ***1/(252 k^6)***, so using it for values of ***k*** exceeding **IROUND((1/252E-12)$^{1/6}$)** = ***40***, will be enough to get *12-digit* accuracy. Let's check it:

```
>FNH(40)+1/41;FNH(41)

     4.30293328284     4.30293328284
```

Now, using ***n*** terms of ***Sum 2*** (say, ***100*** terms) is equivalent to using ***2^n - 1*** *terms* of ***Sum 1*** (i.e. $2^{100} - 1$ = ***1.27 . 10^{30}*** *terms !!*,) thus greatly increasing our chances to achieve a decent enough rate of convergence.

To check if it suffices, I quickly wrote this other raw concoction:

```
10 DESTROY ALL @ FOR M=100 TO 300 STEP 100 @ S=5/3
20 FOR N=3 TO M @ S=S+(FNH(2^N-1)-FNH(2^(N-1)-1))/FNF(N)) @ NEXT N
30 DISP USING "3D,2X,D.4D";M;S @ NEXT M
40 DEF FNH(N) @ IF N>40 THEN 60
50 T=0 @ FOR J=1 TO N @ T=T+1/J @ NEXT J @ FNH=T @ END
60 FNH=LN(N)+.577215664902+1/(2*N)-1/(12*N*N)+1/(120*N^4) @ END DEF
70 DEF FNF(N) @ IF N<3 THEN FNF=N ELSE FNF=N*FNF(IP(LOG2(N)+1))

>RUN
    terms     sum
    ----------------
     100     1.9416
     200     1.9472
     300     1.9486
```

and though ***300*** *terms* of this ***Sum 2*** are equivalent to $2^{300} - 1$ = ***2.04 . 10^{90}*** *terms* of the original ***Sum 1***, it's plainly clear that the convergence rate is *still* too low, despite the fact that we're adding up *exponentially large* chunks of the series.

Now, if we look carefully at the numerator's expression ***H(2^n − 1) − H(2^{n-1} − 1)*** within the summation, we find that it *converges* to ***ln(2)*** as ***n*** tends to *infinity* ...

```
10 DESTROY ALL @ K=LN(2) @ FOR N=5 TO 25 STEP 5
20 DISP USING "2D,2(2X,Z.7D)";N;FNH(2^N-1)-FNH(2^(N-1)-1);RES-K @ NEXT N
30 DEF FNH(N) @ IF N>40 THEN FNH=1/120/N^4-1/12/N/N+1/2/N+.577215664902+LN(N) @ END
40 T=0 @ FOR J=1 TO N @ T=T+1/J @ NEXT J @ FNH=T

>RUN
       N H(..)-H(..)  Diff-ln(2)
      -------------------------
       5  0.7090162   0.0158690
```

```
10   0.6936357    0.0004885
15   0.6931624    0.0000153
20   0.6931477    0.0000005
25   0.6931472    0.0000000
```

... so each numerator tends to the *constant* ***ln(2)***, divided by consecutive values of ***f(n)***, and thus after a while it converges no faster than ***Sum 1***, where the constant is ***1*** instead of ***ln(2)***.

Since the cause of the slow convergence of ***Sum 2*** is that the numerators tend to a *constant* (and thus the convergence to *zero* is provided *only* by the slowly-increasing *denominators* ***f(n)***,) we can try and force the numerators to tend to *zero* (instead of *ln(2)*) by *subtracting* precisely that very constant *ln(2)* from each, which is accomplished by subtracting from ***Sum 2*** the product of the original ***Sum 1*** times ***ln(2)***, like this:

$$S - \ln(2)S = \frac{5}{3} + \sum_{n=3}^{\infty} \frac{H(2^n-1) - H(2^{n-1}-1)}{f(n)} - \ln(2)\left(\frac{3}{2} + \sum_{n=3}^{\infty} \frac{1}{f(n)}\right)$$

$$= \frac{5}{3} - \frac{3\ln(2)}{2} + \sum_{n=3}^{\infty} \frac{H(2^n-1) - H(2^{n-1}-1) - \ln(2)}{f(n)}$$

and now all we need to do is to *isolate* ***S***, which is accomplished by just dividing both sides by ***1 - ln(2)***, obtaining this expression (***Sum 3***) for the sum of the series:

$$S = \left(\frac{1}{1-\ln(2)}\right)\left(\frac{5}{3} - \frac{3\ln(2)}{2} + \sum_{n=3}^{\infty} \frac{H(2^n-1) - H(2^{n-1}-1) - \ln(2)}{f(n)}\right)$$

and as we have

$$H(2^n-1) - H(2^{n-1}-1) - \ln(2) = 2^{-n-1} + O(2^{-2n-2})$$

we just need to sum 2^{-M-1} **= *1E-12*** $\rightarrow$ **M = IP(-LOG2(1E-12)-1)** = ***38 terms*** to achieve *12-digit* accuracy (this value could be hardcoded as a *"magic constant"* but I've opted for calculating it in the initialization.)

My original solution

At long last, my ***original solution*** is thus this *273-byte* ***6****-liner*: *(**LOG2** is from the **Math ROM**)*

```
1  DESTROY ALL @ K=LN(2) @ M=IP(-LOG2(1E-12)-1) @ S=5/3-3*K/2
2  FOR N=3 TO M @ S=S+(FNH(2^N-1)-FNH(2^(N-1)-1)-K)/FNF(N) @ NEXT N
3  STD @ DISP "Sum, #terms:";S/(1-K);M
4  DEF FNH(N) @ IF N>40 THEN FNH=1/120/N^4-1/12/N/N+1/2/N+.577215664902+LN(N) @ END
5  T=0 @ FOR J=1 TO N @ T=T+1/J @ NEXT J @ FNH=T
6  DEF FNF(N) @ IF N<3 THEN FNF=N ELSE FNF=N*FNF(IP(LOG2(N)+1))
```

Line 1 does some initialization, in particular it computes the number of terms to sum.
Line 2 is a loop which adds up the previously computed number of terms.
Line 3 simply displays the final result and the number of terms used.
Lines 4 and ***5*** are the definition of ***H(n)***, the *n-th Harmonic number*.
Line 6 is the *recursive* definition of ***f(n)***.

Let's run it:

```
>RUN
     Sum, #terms: 2.08637766502   38
```

For timing, execute instead:

```
>SETTIME 0 @ CALL @ TIME
```

which takes ***0.27"*** on my emulator and ***34"*** on a *physical* **HP-71B**.

The correct sum is *2.08637766500599+*, which rounds to ***2.08637766501***, so we've got *12 correct digits (save 1 ulp)* using just *38 terms* of ***Sum 3***.

Well, that'll be it for now, I hope you enjoyed it and even learned a little from it. I'll post next ***Problem 4*** after *Christmas*, so as to avoid having to compete with *Xmas* for your attention. 😃

V.
Edit: corrected a very subtle typo.

PM WWW FIND EDIT QUOTE REPORT

5th December, 2022, 16:06 (This post was last modified: 5th December, 2022 19:09 by J-F Garnier.) **Post: #39**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

> **Valentin Albillo Wrote:** (4th December, 2022 23:34)
> I hope you enjoyed it and even learned a little from it.

Thanks for this problem, Valentin!

Yes, I really enjoyed it and leaned a lot about the harmonic series.
I only had the *souvenir* that the 1/1+1/2+1/3... series is divergent and behaves about as the log function, and even didn't know the Hn notation for the harmonic numbers.

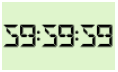
It was really interesting to see different approaches and reasoning, including yours, and the corresponding different programs, all with similar results and performances.

Also, at least for me, the contributions of the other members were very valuable to propose a solution, so it is somehow a collective work.

J-F

EMAIL PM WWW FIND QUOTE REPORT

5th December, 2022, 19:05 **Post: #40**



Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

.. and a last, ultimate contribution.
improvements:
- new stopping criterion S+H/F = S - stops on the 42S when C=33 instead of 38 ;-)
- choice between H(n) definition and approximation is made in a machine-independent way, so Free42 now calculates the result to (near?) 34-digit precision ;-) and the exact same program can be used in the 42S. Had to rewrite H(n) so that the definition code used 2 counters so that large n could be handled on Free42
- started the summation from C=2, as in Albert's code. Made it ever so slightly shorter - but I added the 1/4 at the end, not at the beginning, gaining 1 digit of accuracy ;-)
- shortened F(n)

on the 42S, max C=33 and S=2.08637766501 in 36 seconds
Free42, C=105 and S=2.086377665005988716089755856734133
and yes, I can shave off a few more bytes here and there..
eg 1 ENTER LN1+X i.o. 1 2 LN
but that would be less readable. These are not 'mini challenges', after all ;-)

```
00 { 167-Byte Prgm }
01▸LBL "VA3"
02 2
03 STO "C"
04 CLX
05 STO "S"

06▸LBL 10 @ main loop
07 RCL "C"
08 XEQ H
09 1
10 RCL "C"
11 XEQ F
12 ÷
13 RCL+ "S"
14 ENTER
15 X<> "S"
16 X≠Y?
17 ISG "C"
18 X≠Y? @ aff
19 GTO 10
```

```
20 4 @ wrap-up
21 1/X
22 RCL+ "S"
23 1
24 2
25 LN
26 -
27 ÷
28 1
29 +
30 RTN

31▸LBL D @ binary digits of an integer
32 CLA
33 BINM
34 ARCL ST X
35 CLX
36 ALENG
37 EXITALL
38 RTN

39▸LBL 04
40 R↓
41 XEQ D
42▸LBL F @ F(n), call with 1 n
43 STO× ST Y
44 3
45 X≤Y?
46 GTO 04
47 R↓
48 R↓
49 RTN

50▸LBL H @ H(2^N-1) - H(2^(N-1)-1) - LN(2), input N
51 2
52 X<>Y
53 Y^X
54 RCL ST X @ if 1/2^n + (2^n)^-10 = 1/2^n then use approximate formula
55 -9
56 Y^X
57 1
58 STO+ ST Y
59 -
60 X=0?
61 GTO 00
62 R↓
63 50
64 %
65 0
66▸LBL 02 @ 1/(n-1) + 1/(n-2) + .. + 1/(n/2)
67 DSE ST Z
68 RCL ST Z
69 1/X
70 +
71 DSE ST Y
72 GTO 02
73 2
74 LN
75 -
76 RTN
77▸LBL 00 @ H(n-1)-H(n/2-1)-ln(2) ~= 1/(2n) + 1/(4n^2) - 1/(8n^4) + 1/(4n^6) - 17/(16n^8)
78 R↓
79 STO+ ST X
80 X^2
81 LASTX
82 1/X
83 272
84 RCL÷ ST Z
85 16
86 -
87 R^
```

```
88 ÷
89 2
90 +
91 R^
92 ÷
93 1
94 -
95 R^
96 ÷
97 -
98 END
```

Cheers, Werner

EMAIL PM FIND QUOTE REPORT

5th December, 2022, 21:16 **Post: #41**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012c - Then and Now: Sum

> **Werner Wrote:** (5th December, 2022 19:05)
> - started the summation from C=2, as in Albert's code. Made it ever so slightly shorter - but I added the 1/4 at the end, not at the beginning, gaining 1 digit of accuracy ;-)

Nice work! Numbers dead-on target!

Since we know S is around 2, we might as well go for S-2
For returning S-2, patch with 3 steps on the right.

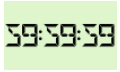
```
20 4 @ wrap-up
21 1/X
22 RCL+ "S"
23 1
24 2
25 LN
26 -
27 ÷                --> 27 -
28 1                --> 28 LASTX
29 +                --> 29 ÷
30 RTN
```

> **Quote:**
> Free42, C=105 and S=2.086377665005988716089755856734133

We now have: S-2=0.0863776650059887160897558567341326**4** // error 13 ULP (should be 77)

EMAIL PM FIND QUOTE REPORT

6th December, 2022, 09:53 **Post: #42**



Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

to get the ..77 as well, in my code:
- sum from C=105 to 2 (small to large)
- change switch criterion for H(n) to 0.01 + n = 0.01 (we need two more digits)
- then, with S = sum(C=105 to 2 step -1, (H(n)-ln(2))/f(n)), calculate (LN(2)-0.75 + S)/(1-LN(2))
(no need, I think, to paste that code anew - it is slightly off-topic)

Cheers, Werner

EMAIL PM FIND QUOTE REPORT

8th December, 2022, 20:55 (This post was last modified: 8th December, 2022 22:36 by Albert Chan.) **Post: #43**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012c - Then and Now: Sum

I was curious what it take if we do brute force, no tricks, S = sum(1/F)

G(b)/F(b) = sum(1/F(k), k = 2^(b-1) .. 2^b - 1)

S ≈ 2.086, for 12-digits accuracy we solve G(b)/F(b) = 1E-11 (1 ULP), for b

G(b)/F(b) ≈ ln(2)/(b*log2(b)) ≈ 1E-11 --> b ≈ 2.2E9 (*)

It take more terms for convergence, even though all other terms below 1 ULP.

sum(1/F) terms needed > 2^b ≈ googol ^ 6622660

(*)
We assume F grow about same rate as primes, to be conservative.
We know F grow faster, because sum of reciprocal primes diverges.

Update: using actual recursive definition of f(b)

ln(2)/f(b) ≈ 1E-11 --> b ≈ 85573726

sum(1/F) terms needed > 2^b ≈ googol ^ 257603

Edit: google should be googol (10^100), corrected.

EMAIL PM FIND QUOTE REPORT

8th December, 2022, 22:17 **Post: #44**

PeterP
Member

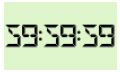
Posts: 172
Joined: Jul 2015

RE: [VA] SRC #012c - Then and Now: Sum

you probably mean a googol, not a google, right?
https://en.wikipedia.org/wiki/Googol

PM FIND QUOTE REPORT

9th December, 2022, 12:08 **Post: #45**



Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012c - Then and Now: Sum

> **Albert Chan Wrote:** (8th December, 2022 20:55)
> I was curious what it take if we do brute force, no tricks, S = sum(1/F)
>
> G(b)/F(b) = sum(1/F(k), k = 2^(b-1) .. 2^b - 1)
>
> S ≈ 2.086, for 12-digits accuracy we solve G(b)/F(b) = 1E-11 (1 ULP), for b
>
> G(b)/F(b) ≈ ln(2)/(b*log2(b)) ≈ 1E-11 --> b ≈ 2.2E9 (*)
>
> It take more terms for convergence, even though all other terms below 1 ULP.
>
> sum(1/F) terms needed > 2^b ≈ googol ^ 6622660
>
> (*)
> We assume F grow about same rate as primes, to be conservative.
> We know F grow faster, because sum of reciprocal primes diverges.
>
> **Update:** using actual recursive definition of f(b)
>
> ln(2)/f(b) ≈ 1E-11 --> b ≈ 85573726
>
> sum(1/F) terms needed > 2^b ≈ googol ^ 257603

Edit: google should be googol (10^100), corrected.

It is far worse than that, I believe. Unless I made a gross error - which is increasingly likely:
Let's try and compute Sk = 1/fk + 1/fk+1 + ... for large k

take k=2^(128-1) so b=128 and k=1.7e38

Sk = ln2*(1/fb + .. 1/fk-1 + 1/fk + 1/fk+1 + ..)
Sk*(1-ln2) = ln2*(1/fb + .. + 1/fk-1)

b=128 = 2^(8-1) so c=8
Sk*(1-ln2) = ln2*ln2*(1/fc + 1/fb-1)

and finally

Sk*(1-ln2) = ln2*ln2*(g4/f4 + g5/f5 + g6/f6 + g7/f7)

for k=2^(128-1), Sk =2.1e-03
for l=2^k-1, Sl = ln2*Sk = 1.5e-03
etc.
so even if the googol^large power term is just below 1e-11, the remainder still adds up to a sizeable fraction..

Cheers, Werner

EMAIL PM FIND QUOTE REPORT

9th December, 2022, 13:27 (This post was last modified: 9th December, 2022 20:13 by Albert Chan.) **Post: #46**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012c - Then and Now: Sum

Knowing s ≈ 2.086377665, lets try something easily calculated

sum(1/f, k=16 .. inf) = s - sum(1/f, k=1 .. 15) ≈ 0.262900
sum(1/f, k=2^16 .. inf) ≈ sum(g/f, k=16 .. inf) ≈ 0.262900 * ln2 (*)

Continued doing this, how many (* ln2) we need to make RHS ≈ 1E-11 (1 ULP) ?

number of ln2's = ln(1E-11/0.262900) / ln(ln2) ≈ 65.46

We will need tetration notation to handle term size this enormous!

16 = 2^2^2 = $^3 2$
2^16 = 2^2^2^2 = $^4 2$

sum(1/f) terms need for 12-digits accuracy (error < 1ULP) ≈ $^{69} 2$
I have no words to describe how big this is ...

(*) the estimate had an off-by-1 error. To make it all into equality:

sum(1/f, k=2^16 .. inf) = sum(g/f, k=**17** .. inf) = (sum(1/f, k=16 .. inf) - 1/f(16)) * K
where ln2 < K < g(17) ≈ ln2 + 1/2^18

We over-estimated the sum, but under-estimated K.
Overall, we probably over-estimated terms needed.

(0.262900 * ln2) ≈ 0.182228
(0.262900 - 1/480) * (ln2 + 1/2^19) ≈ 0.180785

Udpate: off-by-1 error effect after "first ln2" can be ignored.

number of ln2's = ln(1E-11/0.180785) / ln(ln2) + 1 ≈ 65.44

In terms of tetration exponent, off-by-1 error can be ignored.

EMAIL PM FIND QUOTE REPORT

11th December, 2022, 17:59 **Post: #47**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012c - Then and Now: Sum

Hi, all,

As I see there's still some activity in this thread, I'm providing an ***Epilogue*** of sorts, including additional comments about my ***Problem 3*** and the featured series. Let's start ...

Epilogue

Most of my many *Problems* and *Challenges* large and small usually feature some mathematical aspects that can be *intriguing*, *curious* and even *weird* at times, and this ***Problem 3*** is no exception.

The surprising factor here is that the series whose sum I asked people to compute looks quite simple, defined in a single line of text, yet it converges *excruciatingly slowly* to say the least, requiring a ***hyperexponential*** number of terms to get even *one* correct digit (let alone *10-12* digits,) so most people's natural instinct to sum a (somewhat large) number of terms to get some value leads *absolutely nowhere* and thus other more refined techniques are sorely needed, as seen for instance in *my solution* above.

Now, slowly convergent and divergent series are not unusual and sometimes they resemble each other so much that it's not hard but certainly not trivial to ascertain whether a given series converges or diverges. For instance, all the following series are *divergent* ...

$$\sum_{n=2}^{\infty}\frac{1}{n\log n},\ \sum_{n=3}^{\infty}\frac{1}{n\log n\log\log n},\ \sum_{n=16}^{\infty}\frac{1}{n\log n\log\log n\log\log\log n},\ \cdots$$

... while all the following very similar ones are *convergent* for any **ε > 0** ...

$$\sum_{n=2}^{\infty}\frac{1}{n(\log n)^{1+\varepsilon}},\ \sum_{n=3}^{\infty}\frac{1}{n\log n(\log\log n)^{1+\varepsilon}},\ \sum_{n=16}^{\infty}\frac{1}{n\log n\log\log n(\log\log\log n)^{1+\varepsilon}},\cdots$$

... and as you can see the *divergent* ones correspond to the particular case **ε = 0**.

And what about *our* series ? Well, as it happens its terms tend to zero *more slowly* than the terms of any of the *convergent* series above but *faster* than the terms of any of the *divergent* series, almost a "middle" case so to say. But as there's no series that lies *exactly* on the (also nonexistent) *"dividing line"* between convergence and divergence), which is it ? Does this series converge or does it diverge ?

Fortunately for the soundness of my ***Problem 3***, it *does converge* and matter of fact its sum is just a trifle over *2*, namely *2.08637766500599+* but, as stated above, it would require summing a *hyperexponential* number of terms to get *12* correct digits, never mind *15* or more.

However, if multiprecision is available, using the formula ***Sum 3*** obtained in *my solution* above (and using a multiprecision version of ***H(n)***) we can get greater accuracy by just increasing the number of terms summed up. For instance, using ***50*** *terms* it will deliver the *15 correct digits* shown earlier, and so on and so forth.

On a final note, the fact that I stated in the series' definition that *"**d(n)** = number of **binary** digits of **n**"* is *crucial* for the convergence of the series. Had I stated *"ternary" (base **3**)* digits instead of *"binary" (base **2**)*, the resulting series would've been ***divergent***.

> **Note**: if ***d(n)*** is stated in terms of the logarithm base ***a***, the series does *converge* for ***a < e*** and *diverges* otherwise. Thus, for *binary* digits the base is *2 < e* and the series does *converge*, while for *ternary* digits the base is *3 > e* and the resulting series would *diverge*.

See you in ***Problem 4*** next *January*.
V.