[VA] SRC#002- Almost integers and other beasties

01-24-2019, 12:58 AM **Post: #21**

Valentin Albillo
Senior Member

Posts: 347
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC#002- Almost integers and other beasties

Hi again,

Dealing with some search theory, I recently came across this short expression:

$$\frac{1}{12}+\frac{\Pi^2}{6Ln^2(2)}+\frac{2}{Ln(2)}\sum_{k=1}^{\infty}\frac{(-1)^k}{k(2^k-1)}$$

which this straightforward **HP-71B** program *(a command-line expression would work as well)* quickly evaluates to the *apparently* exact *integer* value **1**:

```
1     DESTROY ALL @ V=0 @ FOR K=1 TO 40 @ V=V+(-1)^K/K/(2^K-1) @ NEXT K
2     V=V*2/LN(2)+1/12+PI^2/6/LN(2)^2 @ DISP V

>RUN
            1
```

But it's not !
See if you can get the actual, more accurate *almost-integer* value with your HP. It certainly qualifies as a most amazing almost-integer one ! 😀

V.
.

02-09-2019, 11:53 PM **Post: #22**

Gerson W. Barbosa
Senior Member

Posts: 1,135
Joined: Dec 2013

RE: [VA] SRC#002- Almost integers and other beasties

Valentin Albillo Wrote: ⇨ (01-04-2019 12:33 AM)

For completeness, I forgot to include this remarkable one which I discovered and posted here last March:

```
Sin(9*(Sin 1 + Cos 40)) = 0.9999999999999998
```30826985368...

which differs from the integer **1** by about *1e-16.*

Beautiful one! Hope you don't mind if I uglify it a bit:

Sin(9*(Sin 1 + Cos 40) + 5/(e*10^8))

Best regards,

Gerson.

EMAIL PM FIND QUOTE REPORT

02-10-2019, 12:33 AM **Post: #23**

**ttw**
Member

Posts: 164
Joined: Jun 2014

**RE: [VA] SRC#002- Almost integers and other beasties**

There are quite a few of these strewn about mathematical sites on the internet. Some have easy explanations. Some have very deep explanations. Some have no seeming reason at all.

Sin(11) = about -1
Exp(pi)-pi = about 20
Cos(pi*Cos(pi*Cos(Ln(pi+20)))) = about 1
Exp(pi*Sqrt(163)) = about 262537412640768744

EMAIL PM FIND QUOTE REPORT

02-10-2019, 12:56 AM **Post: #24**

**Valentin Albillo**
Senior Member

Posts: 347
Joined: Feb 2015
Warning Level: 0%

**RE: [VA] SRC#002- Almost integers and other beasties**

.
Hi, **Gerson**:

**Gerson W. Barbosa Wrote:** ⇨ (02-09-2019 11:53 PM)

**Valentin Albillo Wrote:** ⇨ (01-04-2019 12:33 AM)

For completeness, I forgot to include this remarkable one which I discovered and posted here last March:

```
Sin(9*(Sin 1 + Cos 40)) = 0.9999999999999998
```30826985368...

which differs from the integer **1** by about *1e-16.*

Beautiful one! **Hope you don't mind if I uglify it a bit:**

**Sin(9*(Sin 1 + Cos 40) + 5/(e*10^8))**

I don't understand ...

Also, could you accurately evaluate the expression I gave in my post #21 in this thread (the previous one to yours) and if so, what accurate result did you get ? The proximity to the exact integer 1 is *not* a coincidence, there are deep mathematical reasons for it.

Best regards and have a nice weekend.
V.
.

PM FIND EDIT QUOTE REPORT

02-10-2019, 04:41 AM **Post: #25**

**Gerson W. Barbosa**
Senior Member

Posts: 1,135
Joined: Dec 2013

**RE: [VA] SRC#002- Almost integers and other beasties**

> **Valentin Albillo Wrote:** ⇨ (02-10-2019 12:56 AM)
>
> .
> Hi, **Gerson**:
>
> > **Gerson W. Barbosa Wrote:** ⇨ (02-09-2019 11:53 PM)
> >
> > Beautiful one! **Hope you don't mind if I uglify it a bit:**
> >
> > **Sin(9*(Sin 1 + Cos 40) + 5/(e*10^8))**
>
> I don't understand ...

Hi, Valentin,

The factor of *e* doesn´t appear to fit nicely in the trigonometric expression, but it makes the result get a bit closer to 1:

Sin(9*(Sin 1 + Cos 40) + 5/(e*10^8)) = **0.99999999999999999999999997**7740056822767

> **Valentin Albillo Wrote:** ⇨ (02-10-2019 12:56 AM)
>
> Also, could you accurately evaluate the expression I gave in my post #21 in this thread (the previous one to yours) and if so, what accurate result did you get ? The proximity to the exact integer 1 is *not* a coincidence, there are deep mathematical reasons for it.
>
> Best regards and have a nice weekend.
> V.
> .

```
%%HP: T(3)A(R)F(.);
\<< 126 40 'DIGITS' STO 0 1 ROT
  FOR k 1 FNEG k FY\|^X k FDIV 2 k FY\|^X 1 FSUB FDIV FADD
```

```
  NEXT DUP FADD 2 FLN FDIV 1 12 FDIV FADD FPI FSQ 6 FDIV 2 FLN FSQ FDIV FADD ZZ\
<-\->F DROP \->STR DUP HEAD "." + SWAP TAIL +
\>>

EVAL ->

"1.000000000001237412575736110228719610648"
```

The RPL program above requires the LongFloat library.

Number of iterations = Ceil(W(10^n*ln(2))/ln(2)), where n = number of digits and W(x) is the Lambert W function.

Best regards,

Gerson.

EMAIL PM FIND QUOTE REPORT

02-10-2019, 02:48 PM (This post was last modified: 02-10-2019 02:56 PM by Gerson W. Barbosa.) **Post: #26**

Gerson W. Barbosa
Senior Member

Posts: 1,135
Joined: Dec 2013

RE: [VA] SRC#002- Almost integers and other beasties

> **Gerson W. Barbosa Wrote:** ⇨ (02-10-2019 04:41 AM)
>
> ```
> %%HP: T(3)A(R)F(.);
> \<< 126 40 'DIGITS' STO 0 1 ROT
>   FOR k 1 FNEG k FY\|^X k FDIV 2 k FY\|^X 1 FSUB FDIV FADD
>   NEXT DUP FADD 2 FLN FDIV 1 12 FDIV FADD FPI FSQ 6 FDIV 2 FLN FSQ FDIV FADD ZZ\
> <-\->F DROP \->STR DUP HEAD "." + SWAP TAIL +
> \>>
>
> EVAL ->
>
> "1.000000000001237412575736110228719610648"
> ```
>
> The RPL program above requires the LongFloat library.
>
> Number of iterations = Ceil(W(10^n*ln(2))/ln(2)), where n = number of digits and W(x) is the Lambert W function.

This is a more generic version that takes the desired number of digits, *n*, as an argument. As I am using an approximation for W(x), the estimation of the required number of iterations might not be exact for small values of n. Both programs are basically a straightforward conversion of Valentin's HP-71B program in post #21.

```
100

%%HP: T(3)A(R)F(.);
\<< RCLF SWAP -105 CF -3 CF DUP 'DIGITS' STO ALOG 2 LN * LN DUP LN - LASTARG SWAP
/ + 2 LN / CEIL 0 1 ROT
  FOR k 1 FNEG k FY\|^X k FDIV 2 k FY\|^X 1 FSUB FDIV FADD
  NEXT DUP FADD 2 FLN FDIV 12 FINV FADD FPI FSQ 6 FDIV 2 FLN FSQ FDIV FADD ZZ\<-
\->F NEG SWAP \->STR DUP SIZE ROT \=/ -51 FC? { "." } { "," } IFTE UNROT { DUP
```

```
TAIL SWAP HEAD } { "0" } IFTE UNROT + + SWAP STOF
\>>

EVAL ->

1.0000000000012374125757361102287196106466728742977320481965484438441718256405304288509138855861935 25
```

(132.59 seconds on the HP 50g)

```
 5 -> 0.99990

 6 -> 1.00000

 7 -> 1.000000

12 -> 1.00000000001

20 -> 1.0000000000012374126

40 -> 1.0000000000012374125757361102287196106466728742977

50 -> 1.0000000000012374125757361102287196106466728742977
```

EMAIL PM FIND QUOTE REPORT

02-10-2019, 05:26 PM (This post was last modified: 02-10-2019 06:19 PM by Albert Chan.) **Post: #27**

Albert Chan
Senior Member

Posts: 624
Joined: Jul 2018

RE: [VA] SRC#002- Almost integers and other beasties

> **Gerson W. Barbosa Wrote:** ⇒ (02-10-2019 02:48 PM)
> Number of iterations = Ceil(W(10^n*ln(2))/ln(2)), where n = number of digits and W(x) is the Lambert W function.
>
> ```
> 50 -> 1.0000000000012374125757361102287196106466728742977
> ```

Above 50 digits numbers are confirmed corrrect.
Can you explain how the iteration count formula is derived ?

> **Valentin Albillo Wrote:** ⇒ (01-24-2019 12:58 AM)
>
> $$\frac{1}{12}+\frac{\Pi^2}{6Ln^2(2)}+\frac{2}{Ln(2)}\sum_{k=1}^{\infty}\frac{(-1)^k}{k(2^k-1)}$$

I see that the sum gained about 1 bit precision with each iteration, so I use n / log10(2) ~ 3.322n
It is slightly over-estimated, but not by much.

Edit: the difference is the effect of 1/k, iterations ~ 3.322n - log2(3.322n)

EMAIL PM FIND QUOTE REPORT

02-10-2019, 06:06 PM (This post was last modified: 02-10-2019 06:14 PM by Gerson W. Barbosa.) **Post: #28**

Gerson W. Barbosa
Senior Member

Posts: 1,135
Joined: Dec 2013

RE: [VA] SRC#002- Almost integers and other beasties

> **Albert Chan Wrote:** ➡ (02-10-2019 05:26 PM)
>
> > **Gerson W. Barbosa Wrote:** ➡ (02-10-2019 02:48 PM)
> >
> > Number of iterations = Ceil(W(10^n*ln(2))/ln(2)), where n = number of digits and W(x) is the Lambert W function.
> >
> > ```
> > 50 -> 1.00000000000123741257573611022871961064667287429773204819654844384417182564053
> > ```
>
> Above 50 digits numbers are confirmed corrrect.
> Can you explain how the iteration count formula is derived ?

Starting with $\log_{10}(k.2^k) = n$, I solved $k.2^k = 10^n$ for k. Well, actually W|A did :-)

$k = W(10^n.\ln(2))/\ln(2)$

BTW, it's better to replace `ALOG 2 LN * LN` with `10 LN * 2 LN LN +` just in case you want to evaluate it to one thousand digits or more:

```
1000

\<< RCLF SWAP -105 CF -3 CF DUP 'DIGITS' STO 10 LN * 2 LN LN + DUP LN -
LASTARG SWAP / + 2 LN / CEIL 0 1 ROT
FOR k 1 FNEG k FY\|^X k FDIV 2 k FY\|^X 1 FSUB FDIV FADD
NEXT DUP FADD 2 FLN FDIV 12 FINV FADD FPI FSQ 6 FDIV 2 FLN FSQ FDIV FADD ZZ\<-
\->F NEG SWAP \->STR DUP SIZE ROT \=/ -51 FC? { "." } { "," } IFTE UNROT { DUP
TAIL SWAP HEAD } { "0" } IFTE UNROT + + SWAP STOF
\>>

EVAL ->

1.00000000000123741257573611022871961064667287429773204819654844384417182564053
42885091388558619352497626845334008619165837450903001904672978600537014020759086
53972210668862091672466121582555971369478336628117111805015220469582973183869567
49813586119403326983996836799698362386464361717810944715248515847063950123049027
85528947933780707497372217486300760223459895208271343612686740722308571122141720
60133366839502480369120342433228486075440964655597427100579440680205978185469463
76873631661338090760132715563114425400886965240835824220034845681146540332945848
09115605566107380898677023776867118135971086811207980254600217139884419904867460
04071504113819770701596087697700373957210018691354928394481593778392574770677787
76337799415286212226231921875049198549974749265675547171167195366657491492695699
89391692666496234240604535789799813602754866102044836132703557955522820580941853
00921892327891632974811217666530275540985323109184583425808784453698915073727444360690362088831464093685258316858397747 10
```

(363.97 seconds on the emulator)

B529h, 363 bytes

EMAIL PM FIND QUOTE REPORT

02-10-2019, 10:42 PM **Post: #29**

Gerson W. Barbosa
Senior Member

Posts: 1,135
Joined: Dec 2013

RE: [VA] SRC#002- Almost integers and other beasties

> **Albert Chan Wrote:** ➡ (02-10-2019 05:26 PM)
> Edit: the difference is the effect of 1/k, iterations ~ 3.322n - log2(3.322n)

Quite good estimation! It allows me to save 10 bytes by replacing `LN + DUP LN - LASTARG SWAP / + 2 LN` with `SWAP DUP2 / LN + SWAP SQ`

Now # 62Fh, 353 bytes.

EMAIL PM FIND QUOTE REPORT

02-13-2019, 03:49 PM (This post was last modified: 02-13-2019 03:51 PM by Albert Chan.) **Post: #30**

Albert Chan
Senior Member

Posts: 624
Joined: Jul 2018

RE: [VA] SRC#002- Almost integers and other beasties

Hi, Gerson W. Barbosa

Your code might be shortened and faster by removing (-1)^k factor.
Since it pre-calculated iterations required, summing backwards may be more accurate.
Example: With 4 digits precision, sum 4 terms,

t = |sum| = 1 - 1/(2*3) + 1/(3*7) - 1/(4*15)

Steps:
t = 0
t = 1/60 - t = 0.01667
t = 1/21 - t = 0.04762 - t = 0.03095
t = 1/6 - t = 0.1667 - t = 0.1358
t = 1 - t = 0.8642

EMAIL PM FIND QUOTE REPORT

02-15-2019, 06:29 PM **Post: #31**

Gerson W. Barbosa
Senior Member

Posts: 1,135
Joined: Dec 2013

RE: [VA] SRC#002- Almost integers and other beasties

> **Albert Chan Wrote:** ➡ (02-13-2019 03:49 PM)
> Your code might be shortened and faster by removing (-1)^k factor.
> Since it pre-calculated iterations required, summing backwards may be more accurate.

Sure I am aware of these, but initially I was not interested in optimizing the code for speed, only for size. Anyway, since the summation can be evaluated without resorting to exponentiation, as you've pointed out, there's no reason to use it, at least inside the loop:

```
enter k;
sign := -2*mod(k, 2) + 1;
```

```
sum := 0;
a := 2^k - 1;
for i = k to 1 step -1;
  sum := sum + sign/(a*i);
  sign := -sign;
  a := a div 2
next i;
display sum
```

The RPL code, however, is now slightly slower, because either it's not properly optimized yet or the LongFloat FY^X function is very efficient for integer arguments.

```
%%HP: T(3)A(R)F(.);
\<< RCLF SWAP -105 CF -3 CF DUP 1 + 'DIGITS' STO 10 LN * 2 LN SWAP DUP2 / LN +
SWAP / CEIL 2 OVER ^ 1 - OVER 2 MOD -2 * 1 + SWAP 0 4 ROLL 1
  FOR i OVER i FMULT 4 PICK FMULT FINV FADD ROT NEG ROT 2 FDIV FIP ROT -1
  STEP UNROT DROP2 DUP FADD 2 FLN FDIV 12 FINV FADD FPI FSQ 6 FDIV 2 FLN FSQ
FDIV FADD ZZ\<-\->F NEG SWAP \->STR DUP SIZE ROT \=/ -51 FC? { "." } { "," }
IFTE UNROT { DUP TAIL SWAP HEAD } { "0" } IFTE UNROT + + DUP SIZE " " REPL " "
"" SREPL DROP SWAP STOF
\>>
```

1A8h, 428.5 bytes

141.34 seconds on my HP-50g, for 100 (previously, 132.59 seconds)

```
  1 -> 0.9

  2 -> 1.0

  3 -> 0.999

  4 -> 0.9999

  5 -> 0.99998

  6 -> 1.00000

  7 -> 0.9999997

 12 -> 1.00000000000

 20 -> 1.00000000000012374125

 40 -> 1.0000000000012374125757361102287196106466

 50 -> 1.00000000000123741257573611022871961064667287429773204819654844384417182564053042885091338855861935 25
```

Regards,

Gerson.

EMAIL PM FIND QUOTE REPORT

02-15-2019, 08:50 PM **Post: #32**

Albert Chan
Senior Member

Posts: 624
Joined: Jul 2018

RE: [VA] SRC#002- Almost integers and other beasties

Gerson W. Barbosa Wrote: ⇨ (02-15-2019 06:29 PM)

```
a := 2^k - 1;
...
a := a div 2
```

Despite RPL does not see speedup, this is a great optimization !
Can the sign flipping code be removed, and possibly gain some speed ?

Code:

```
>>> sum, k = 0, 35       # for 12 digits accuracy
>>> a = (1<<k) - 1       # k bits of 1
>>> while k > 0:
...     sum = 1/(a*k) - sum
...     a, k = a>>1, k-1
...
>>> print sum
0.868876652659
```

EMAIL PM FIND QUOTE REPORT

02-16-2019, 12:48 AM (This post was last modified: 02-16-2019 03:15 AM by Gerson W. Barbosa.) **Post: #33**

Gerson W. Barbosa
Senior Member

Posts: 1,135
Joined: Dec 2013

RE: [VA] SRC#002- Almost integers and other beasties

Albert Chan Wrote: ⇨ (02-15-2019 08:50 PM)

Can the sign flipping code be removed, and possibly gain some speed ?

Code:

```
>>> sum, k = 0, 35       # for 12 digits accuracy
>>> a = (1<<k) - 1       # k bits of 1
>>> while k > 0:
...     sum = 1/(a*k) - sum
...     a, k = a>>1, k-1
...
>>> print sum
0.868876652659
```

Oh, I see!

```
%%HP: T(3)A(R)F(.);
\<< RCLF SWAP -105 CF -3 CF DUP 1 + 'DIGITS' STO 10 LN * 2 LN SWAP DUP2 / LN +
SWAP / CEIL 2 OVER ^ 1 - 0 ROT 1
  FOR i OVER i FMULT FINV SWAP FSUB SWAP 2 FDIV FIP SWAP -1
```

```
  STEP FSUB DUP FADD 2 FLN FDIV 12 FINV FADD FPI FSQ 6 FDIV 2 FLN FSQ FDIV
FADD ZZ\<-\->F NEG SWAP \->STR DUP SIZE ROT \=/ -51 FC? { "." } { "," } IFTE
UNROT { DUP TAIL SWAP HEAD } { "0" } IFTE UNROT + + DUP SIZE " " REPL " " ""
SREPL DROP SWAP STOF
\>>
```

10B6h, 393.5 bytes

Now 129.59 seconds for 100 digits.

P.S.: Replaced NIP FNEG with FSUB at loop exit. I'd imagined the former were faster, but it turns our it's actually 0.05 slower (although this is not conclusive after one measurement only).

EMAIL PM FIND QUOTE REPORT

02-23-2019, 06:58 AM **Post: #34**

ttw
Member

Posts: 164
Joined: Jun 2014

RE: [VA] SRC#002- Almost integers and other beasties

For things like (-1)^k in a list with k running over a large range, one can process by unrolling the loop twice. There needs to be either cleanup at the end or pre compute the first step for odd range.

EMAIL PM FIND QUOTE REPORT

« Next Oldest | Next Newest »

Enter Keywords Search Thread

Pages (2): « Previous 1 **2**

NEW REPLY

View a Printable Version
Send this Thread to a Friend
Subscribe to this thread

User(s) browsing this thread: Valentin Albillo*