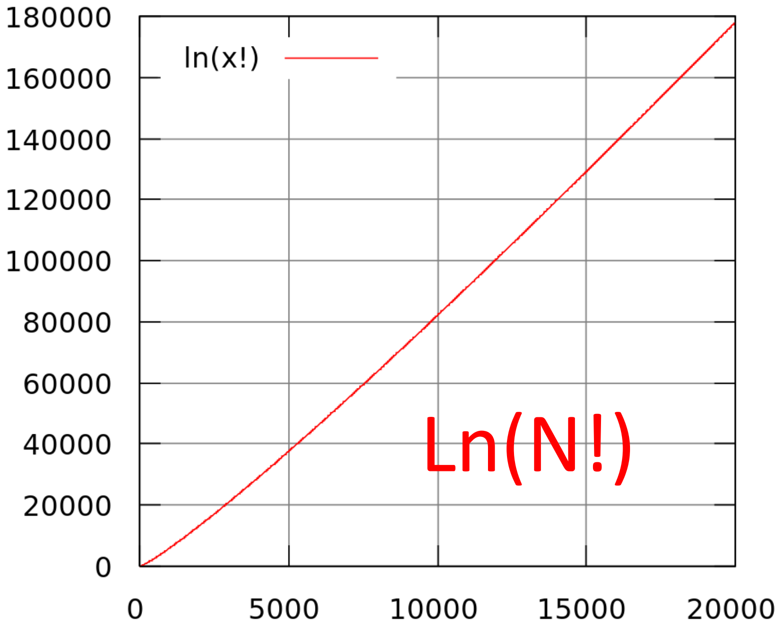
N!
Exact Solution

SVFIG Zoom

Dec. 18, 2021

A. Vidovic



Ln(N!)

The Problem

- Compute 100! exactly.

The Outer Wrapper

```
\ Compute and display N!

 : factorial ( n -- )
   dup init-factbuf
   compute-factorial
   print-factorial
 ;
```

The Setup

```
\ Setup buffer allowing for minus & 1.

 : init-factbuf ( n -- )
  0 factbuf c!  dup 0 <=
   if drop 1  then
   dup 100 <
   if  append-digits-to-factbuf
   else  dup 140 <
   if 100 - append-digits-to-factbuf
      1 append-digits-to-factbuf
     else  drop cr ." Overflow error"
     then   then  ;
```

The Outer Loop

```
\ Computes into the buffer

: compute-factorial ( n -- )
   dup 1 >
   if
        1 do i fact-iteration  loop
   else drop
  then  ;
```

The Inner Loop

```
\ One iteration using the factbuf

 : fact-iteration
   0 swap  factbuf dup c@
   swap 1+ dup  rot + swap
   do  swap over  i c@  fact-atom
       i c! swap
   loop
   drop  dup 0>
   if      append-digits-to-factbuf
     else  drop then    ;
```

For One Digit

```
 : fact-atom
( carry0 multipl digit -- carry1 result )
\ This is decimal arithmetic with carry.
    * +   dup 10 /
    swap over 10 * -  ;
```

Append One Digit

```
 \ append one or two digits to the factbuf
: append-digits-to-factbuf
   factbuf c@ 1+ swap    dup 10 <
  if  over store-digit
      factbuf c!
  else dup 10 /
       swap over 10 * -
       rot dup 1+ -rot  store-digit
       dup factbuf c!   store-digit
  then ;
```

Display The Output

```
: print-factorial
   factbuf c@   dup 0 <=
   if   drop cr ." Nothing to display."
     else
       factbuf  swap over +
       do   i c@ 48 + emit -1 +loop
       cr
   then    ;
```

The Outer Wrapper

```
\ Compute and display N!

 : factorial ( n -- )
   dup init-factbuf
   compute-factorial
   print-fact
 ;
```

The Result

```
cr cr  100 factorial

9332621544394415268169923885626670
0490715968264381621468592963895217
5999932299156089414639761565182862
53697920827223758251185210916864
000000000000000000000000

 ok
```