

Strings of Consecutive Happy Numbers

23 Feb 2008

1 Mar 2008

15 Mar 2008

21 Mar 2008

1 Sept 2008

6 Sept 2008

10 Sept 2008

12 Sept 2008

18 Sept 2008

17 Oct 2008

10 Nov 2008

1 Jan 2009

6 Jan 2009

9 Jan 2009

22 Jan 2009

26-27 June 2009

The goal here is to show that the first string of fourteen consecutive happy numbers.

We claim the 34567901197532 digit number

$N = 7.(2098518518492 \text{ nines}).8.(32469382679037 \text{ nines}).3$ works.

Note that Dr. Grundman uses S_2 but we will simply use S in the explanation, which in our program is the procedure 'onestep'.

Also note we will use a dot as the digit concatenation operator. Thus, 111112999994 could be written 11111.2.9999.94 if we wish.

First we define our procedures.

```
> restart;
```

```
> f := n → n^2; #in case we ever want to investigate the cube of the digits, etc.
```

$f := n \rightarrow n^2$ (1)

```
> bs := 10;
```

```
    #this is the base, in case we ever want to investigate binary or ternary or any other base.
```

$bs := 10$ (2)

```
> onestep := proc(n1)
```

```
    #this is what Dr. Grundman calls  $S_2(n1)$  and what we will simply call  $S$  below.
```

```
    local ans, n, d;
```

```
    n := n1;
```

```
    ans := 0;
```

```
    while n > 0 do
```

```
        d := n mod bs;
```

```
        ans := ans + f(d);
```

```
        n := (n-d) / bs;
```

```
    end do;
```

```
    ans;
```

```
end;
```

```

onestep := proc(n1)
  local ans, n, d;
  n := n1;
  ans := 0;
  while 0 < n do d := mod(n, bs); ans := ans + f(d); n := (n - d) / bs end do;
  ans
end proc

```

(3)

```

> happy := proc(n)
  # returns -1 if not happy, and returns the number of steps to reach 1 if it is happy
  local m, j, height;
  m := n;
  height := -1;
  for j from 1 to 100 while (m > 1 and m ≠ 4) do
  m := onestep(m);
  end do;
  if m = 1 then height := j; end if;
  height;
end;

```

```

happy := proc(n)

```

(4)

```

  local m, j, height;
  m := n;
  height := -1;
  for j to 100 while 1 < m and m <> 4 do m := onestep(m) end do;
  if m = 1 then height := j end if;
  height
end proc

```

The next procedure is only needed when we want to find the smallest N with $S(N) = n$ for a given n. A separate worksheet has the details on how this is constructed. The array contains the smallest N for $1 \leq n \leq 486 = 6 \cdot 81$.

```

> lowS := [1, 11, 111, 2, 12, 112, 1112, 22, 3, 13, 113, 222, 23, 123, 1123, 4, 14, 33, 133, 24,
124, 233, 1233, 224, 5, 15, 115, 1115, 25, 125, 1125, 44, 144, 35, 135, 6, 16, 116, 1116,
26, 45, 145, 335, 226, 36, 136, 1136, 444, 7, 17, 117, 46, 27, 127, 1127, 246, 227, 37, 137,
1137, 56, 156, 1156, 8, 18, 118, 337, 28, 128, 356, 1356, 66, 38, 57, 157, 266, 238, 257,
1257, 48, 9, 19, 119, 248, 29, 129, 1129, 466, 58, 39, 139, 1139, 258, 239, 1239, 448, 49,
77, 177, 68, 168, 277, 1277, 268, 458, 59, 159, 666, 368, 259, 1259, 2666, 78, 178, 359,
468, 69, 169, 1169, 2468, 269, 378, 577, 1577, 568, 369, 1369, 88, 188, 79, 179, 288, 469,
279, 1279, 668, 388, 578, 379, 1379, 2388, 569, 1569, 488, 89, 189, 777, 1777, 289, 1289,
2777, 4668, 588, 389, 579, 1579, 2588, 2389, 2579, 4488, 489, 99, 199, 688, 1688, 299,
1299, 2688, 4588, 589, 399, 1399, 3688, 2589, 2399, 12399, 788, 499, 779, 1779, 689,
1689, 2779, 12779, 2689, 3788, 599, 1599, 5688, 3689, 2599, 888, 1888, 789, 1789, 2888,
4689, 699, 1699, 6688, 3888, 2699, 3789, 5779, 15779, 5689, 3699, 4888, 889, 1889, 799,
1799, 2889, 4699, 2799, 12799, 5888, 3889, 5789, 3799, 13799, 23889, 5699, 15699,
4889, 899, 1899, 6888, 16888, 2899, 12899, 26888, 45888, 5889, 3899, 5799, 15799,
25889, 23899, 25799, 7888, 4899, 999, 1999, 6889, 16889, 2999, 12999, 26889, 37888,
5899, 3999, 13999, 36889, 25899, 8888, 18888, 7889, 4999, 7799, 17799, 6899, 16899,
27799, 38888, 26899, 37889, 5999, 15999, 56889, 36899, 25999, 8889, 18889, 7899,

```

17899, 28889, 46899, 6999, 16999, 58888, 38889, 26999, 37899, 57799, 157799, 56899, 36999, 48889, 8899, 18899, 7999, 17999, 28899, 46999, 27999, 127999, 58889, 38899, 57899, 37999, 137999, 238899, 56999, 78888, 48899, 8999, 18999, 68889, 168889, 28999, 128999, 268889, 378888, 58899, 38999, 57999, 157999, 258899, 88888, 188888, 78889, 48999, 9999, 19999, 68899, 168899, 29999, 129999, 268899, 378889, 58999, 39999, 139999, 368899, 258999, 88889, 188889, 78899, 49999, 77999, 177999, 68999, 168999, 277999, 388889, 268999, 378899, 59999, 159999, 568899, 368999, 259999, 88899, 188899, 78999, 178999, 288899, 468999, 69999, 169999, 588889, 388899, 269999, 378999, 577999, 1577999, 568999, 369999, 488899, 88999, 188999, 79999, 179999, 288999, 469999, 279999, 1279999, 588899, 388999, 578999, 379999, 1379999, 888888, 569999, 788889, 488999, 89999, 189999, 688899, 1688899, 289999, 1289999, 2688899, 3788889, 588999, 389999, 579999, 1579999, 2588999, 888889, 1888889, 788899, 489999, 99999, 199999, 688999, 1688999, 299999, 1299999, 2688999, 3788899, 589999, 399999, 1399999, 3688999, 2589999, 888899, 1888899, 788999, 499999, 779999, 1779999, 689999, 1689999, 2779999, 3888899, 2689999, 3788999, 599999, 1599999, 5688999, 3689999, 2599999, 888999, 1888999, 789999, 1789999, 2888999, 4689999, 699999, 1699999, 5888899, 3888999, 2699999, 3789999, 5779999, 8888888, 5689999, 3699999, 4888999, 889999, 1889999, 799999, 1799999, 2889999, 4699999, 2799999, 12799999, 5888999, 3889999, 5789999, 3799999, 13799999, 8888889, 5699999, 7888899, 4889999, 899999, 1899999, 6888999, 16888999, 2899999, 12899999, 26888999, 37888899, 5889999, 3899999, 5799999, 15799999, 25889999, 8888899, 18888899, 7888999, 4899999, 9999999] :

```
> minimalN := proc(n)
  local q, r, k, ans;
  global lowS;;
  if n < 487 then ans := lowS[n];
  else
    q := iquo(n, 81, 'r');
    ans := lowS[n - (q - 5) * 81] * 10q-5 + (10q-5 - 1);
  end if;
  ans;
end;
```

```
minimalN := proc(n)
  local q, r, k, ans;
  global lowS;
  if n < 487 then
    ans := lowS[n]
  else
    q := iquo(n, 81, 'r'); ans := lowS[n - 81 * q + 405] * 10(q - 5) + 10(q - 5) - 1
  end if;
  ans
end proc
```

(5)

```
> minimalbigN := proc(n)
  local q, r, k, ans;
  global lowS;;
  if n < 487 then ans := lowS[n];
  else
    if n < 1000 then
      q := iquo(n, 81, 'r');
```

```

ans := lowS[n - (q - 5) · 81] · 10q-5 + (10q-5 - 1);
else
q := iquo(n, 81, 'r');
ans := [ lowS[n - (q - 5) · 81], q - 5];
end if; end if;
ans;
end;

```

minimalbigN := **proc**(n) (6)

```

local q, r, k, ans;
global lowS;
if n < 487 then
  ans := lowS[n]
else
  if n < 1000 then
    q := iquo(n, 81, 'r'); ans := lowS[n - 81 * q + 405] * 10(q - 5) + 10(q - 5) - 1
  else
    q := iquo(n, 81, 'r'); ans := [lowS[n - 81 * q + 405], q - 5]
  end if
end if;
ans

```

end proc

N = 7. (2098518518492 nines).8.(32469382679037 nines).3

> M1 := onestep(7) + 2098518518492 · 9² + 8² + 32469382679037 · 9²;
M1 := 279999999699962 (7)

> c := 962; M2 := $\frac{(M1 - c)}{1000}$; b := onestep(M2);
c := 962
M2 := 2799999996999
b := 899 (8)

We first verify our candidate N = 7. (2098518518492 nines).8.(32469382679037 nines).2 does indeed give 14 consecutive happy numbers. Note there are seven before the carry and seven after the carry.

> **for** d **from** 2 **to** 9 **do** print(d, happy(onestep(7) + 2098518518492 · 81 + 8² + 32469382679037 · 81 + d²)) **end do**;
2, -1
3, 4
4, 7
5, 7
6, 5
7, 5
8, 5
9, 8 (9)

```

> for d from 0 to 7 do print( d, happy(onestep(7) + 2098518518492·81 + 92 + d2) ) end do;
                                0, 7
                                1, 8
                                2, 8
                                3, 8
                                4, 6
                                5, 6
                                6, 7
                                7, -1

```

(10)

We begin by finding seven or more in a row before a carry, then seven or more in a row after a carry.

The idea is that $N = N1.d0$ where $d0$ is the final digit. Let $M1 = S(N1)$ and we divide $M1 = M2.c$ where c is the final three digits.

If $c+d0^2 < 1000$ then $S(S(N)) = S(M2) + S(c+d0^2)$ so we can merely check this to see if it is happy or not. Below we use $b=S(M2)$ and i for the various final digits we are trying.

To guarantee that $c+d0^2 < 1000$ we investigate $c < 919$ for those "before the carry" and $c < 964$ for those "after the carry." We actually only need to check up to $b=1007$.

```

> for b from 0 to 1500 do
  for c from 0 to 918 do
    fini := false;
    for i from 9 to 3 by -1 while fini = false do
      if happy(b + onestep(c + i2)) = -1 then fini := true; end if;
    end do;
    if fini = false then print(b, c, minimalN(b), happy(b + onestep(c + 22))) end if;
  end do;
end do:

```

(11)

```

> for b from 0 to 2000 do
  for c from 0 to 963 do
    fini := false;
    for i from 0 to 6 while fini = false do
      if happy(b + onestep(c + i2)) = -1 then fini := true; end if;
    end do;
    if fini = false then print(b, c, minimalN(b), happy(b + onestep(c + 72)), happy(b
      + onestep(c + 82)), happy(b + onestep(c + 92))) end if;
  end do;
end do:

```

(11)

1168, 434, 2888999999999999, 5, -1, -1
 1175, 334, 3789999999999999, 5, -1, -1
 1180, 234, 4888999999999999, 5, -1, 4
 1183, 134, 7999999999999999, 5, -1, -1
 1184, 34, 1799999999999999, 5, -1, 7
 1651, 915, 18889999999999999999, 4, -1, -1
 1654, 934, 28889999999999999999, 5, 5, -1
 1668, 815, 18899999999999999999, 4, -1, -1
 1671, 834, 28899999999999999999, 5, 5, -1
 1683, 715, 48899999999999999999, 4, -1, -1
 1686, 734, 68889999999999999999, 5, 5, -1
 1696, 615, 2588999999999999999999, 4, -1, -1
 1699, 634, 7888999999999999999999, 5, 5, -1
 1707, 515, 2688999999999999999999, 4, -1, -1
 1710, 534, 3999999999999999999999, 5, 5, 4
 1716, 415, 7889999999999999999999, 4, -1, -1
 1719, 434, 1779999999999999999999, 5, 5, -1
 1723, 315, 388889999999999999999999, 4, -1, -1
 1726, 334, 5999999999999999999999, 5, 5, -1
 1728, 215, 568899999999999999999999, 4, -1, -1
 1731, 115, 888999999999999999999999, 4, -1, -1
 1731, 234, 888999999999999999999999, 5, 5, -1
 1732, 15, 188899999999999999999999, 4, -1, -1
 1734, 134, 178999999999999999999999, 5, 5, -1
 1735, 34, 288899999999999999999999, 5, 5, -1

(12)

We still need to check all the ones that have carries in the M1 level, that is, $c > 918$ before the carry or $c > 963$ after the carry.

To analyze this, consider the $M1 = S(N1)$ and $M1 = M2.c$ where c is a three digit number.

If $c > 918$ then we need to consider how the carry affects the M2 portion. Let $M2 = M3.d2.99...99$ where the end of M2 has exactly k nines with the digit $d2 < 9$. Then $S(M2) = S(M3) + d2^2 + k*9^2$, so $S(M3) = S(M2) - d2^2 - k*81$. Now the $c+9^2$ will give a carry $10.e.f$ where e and f are the last two digits of $c+9^2$. Thus, after this carry, we have $M5 = M3.(d2+1).00...00.0.e.f$ so $S(M5) = S(M3) + (d2+1)^2 + S(e.f) = S(M2) - d2^2 + (d2+1)^2 - k*81 + S(e.f) = b+2*d2+1-k*81+S(e.f)$. This is what we need to check to see if it is happy.

When $c > 935$ we also need to worry about the carry with $d0=8$, when $c > 950$ also worry about the carry when $d0=7^2$, etc.

The b below corresponds to the $S(M2)$ as above.

```

> for b from 0 to 1500 do
  for c from 919 to 935 do #so these have the last digit 9 carry but not any others.
    fini := false;
    for i from 3 to 8 while fini = false do

```

```

if happy( $b + \text{onestep}(c + i^2)$ ) = -1 then fini := true; end if;
end do;
if fini = false then #now we check for last digit 9
  for k from 0 to floor( $\frac{b}{81}$ ) do
    for d2 from 0 to 8 do
      if happy( $b + 2 \cdot d2 + 1 - k \cdot 81 + \text{onestep}((c + 9^2) \bmod 100)$ ) > 0 then print( $b, c, k, d2,$ 
        happy( $b + \text{onestep}(c + 2^2)$ )) end if;
    end do;
  end do;
end if;
end do;
end do;

```

```

1227, 927, 0, 4, -1
1227, 927, 1, 0, -1
1227, 927, 1, 2, -1
1227, 927, 1, 5, -1
1227, 927, 4, 1, -1
1227, 927, 5, 6, -1
1227, 927, 5, 7, -1
1227, 927, 6, 0, -1
1227, 927, 6, 6, -1
1227, 927, 6, 7, -1
1227, 927, 7, 7, -1
1227, 927, 8, 0, -1
1227, 927, 8, 6, -1
1227, 927, 9, 0, -1
1227, 927, 9, 1, -1
1227, 927, 10, 4, -1
1227, 927, 10, 7, -1
1227, 927, 11, 4, -1
1227, 927, 12, 0, -1
1227, 927, 12, 3, -1
1227, 927, 13, 0, -1
1227, 927, 15, 1, -1
1227, 927, 15, 7, -1

```

(13)

```

> for b from 0 to 1500 do
  for c from 936 to 950 do #so these have the last digit 9 and 8 carry but not any others.
    fini := false;
    for i from 3 to 7 while fini = false do
      if happy( $b + \text{onestep}(c + i^2)$ ) = -1 then fini := true; end if;
    end do;
    if fini = false then #now we check for last digit 9 and 8
      for k from 0 to floor( $\frac{b}{81}$ ) do

```

```

for d2 from 0 to 8 do
  fini2 := false;
  for i from 8 to 9 while fini2 = false do
    if happy(b + 2·d2 + 1 - k·81 + onestep((c + i2) mod 100)) = -1 then fini2 := true; end
    if;
  end do;
  if fini2 = false then print(b, c, k, d2, happy(b + onestep(c + 22))) end if;
end do;
end do;
end if;
end do;
end do;

```

538, 944, 6, 6, -1

1302, 949, 14, 7, -1

(14)

```

> for b from 0 to 1500 do
  for c from 951 to 963 do #so these have the last digit 9,8,7 carry but not any others.
    fini := false;
    for i from 3 to 6 while fini = false do
      if happy(b + onestep(c + i2)) = -1 then fini := true; end if;
    end do;
    if fini = false then #now we check for last digit 9,8,7
      for k from 0 to floor( $\frac{b}{81}$ ) do
        for d2 from 0 to 8 do
          fini2 := false;
          for i from 7 to 9 while fini2 = false do
            if happy(b + 2·d2 + 1 - k·81 + onestep((c + i2) mod 100)) = -1 then fini2 := true; end
            if;
          end do;
          if fini2 = false then print(b, c, k, d2, happy(b + onestep(c + 22))) end if;
        end do;
      end do;
    end if;
  end do;
end do;

```

899, 962, 3, 6, -1

1381, 962, 9, 8, -1

(15)

```

> for b from 0 to 1500 do
  for c from 964 to 974 do #so these have the last digit 9,8,7,6 carry but not any others.
    fini := false;
    for i from 3 to 5 while fini = false do
      if happy(b + onestep(c + i2)) = -1 then fini := true; end if;
    end do;
    if fini = false then #now we check for last digit 9,8,7,6
      for k from 0 to floor( $\frac{b}{81}$ ) do
        for d2 from 0 to 8 do
          fini2 := false;
          for i from 6 to 9 while fini2 = false do

```

```

if happy( $b + 2 \cdot d2 + 1 - k \cdot 81 + \text{onestep}((c + i^2) \bmod 100)$ ) = -1 then fini2 := true; end
if;
end do;
if fini2 = false then print( $b, c, k, d2, \text{happy}(b + \text{onestep}(c + 2^2))$ ) end if;
end do;
end do;
end if;
end do;
end do;

```

```

1123, 974, 10, 2, -1
1373, 964, 3, 5, -1
1421, 967, 0, 8, -1
1433, 967, 0, 2, -1

```

(16)

```

> for b from 0 to 1500 do
  for c from 975 to 983 do #so these have the last digit 9,8,7,6,5 carry but not any others.
    fini := false;
    for i from 3 to 4 while fini = false do
      if happy( $b + \text{onestep}(c + i^2)$ ) = -1 then fini := true; end if;
      end do;
      if fini = false then #now we check for last digit 9,8,7,6,5
        for k from 0 to floor( $\frac{b}{81}$ ) do
          for d2 from 0 to 8 do
            fini2 := false;
            for i from 5 to 9 while fini2 = false do
              if happy( $b + 2 \cdot d2 + 1 - k \cdot 81 + \text{onestep}((c + i^2) \bmod 100)$ ) = -1 then fini2 := true; end
              if;
              end do;
            end do;
          end do;
          if fini2 = false then print( $b, c, k, d2, \text{happy}(b + \text{onestep}(c + 2^2))$ ) end if;
          end do;
        end do;
      end do;
    end do;
  end do;

```

```

439, 980, 2, 8, -1
772, 980, 9, 5, -1
1096, 978, 0, 4, -1
1159, 978, 0, 4, -1
1249, 980, 12, 8, -1
1333, 978, 3, 7, -1
1349, 980, 16, 0, -1
1485, 978, 4, 3, -1
1499, 980, 18, 6, -1

```

(17)

```

> for b from 0 to 1500 do
  for c from 984 to 990 do #so these have the last digit 9,8,7,6,5,4 carry but not any others.
    fini := false;
    for i from 3 to 3 while fini = false do

```

```

if happy( $b + \text{onestep}(c + i^2)$ ) = -1 then fini := true; end if;
end do;
if fini = false then #now we check for last digit 9,8,7,6,5,4
  for k from 0 to floor( $\frac{b}{81}$ ) do
    for d2 from 0 to 8 do
      fini2 := false;
      for i from 4 to 9 while fini2 = false do
        if happy( $b + 2 \cdot d2 + 1 - k \cdot 81 + \text{onestep}((c + i^2) \bmod 100)$ ) = -1 then fini2 := true; end
          if;
        end do;
      if fini2 = false then print( $b, c, k, d2, \text{happy}(b + \text{onestep}(c + 2^2))$ ) end if;
      end do;
    end do;
  end if;
end do;

```

```

> for b from 0 to 1500 do
  for c from 991 to 999 do #so these have the last digit 9,8,7,6,5,4,3 carry but not any others.
    #fini := false;
    #for i from 3 to 4 while fini = false do
      #if happy( $b + \text{onestep}(c + i^2)$ ) = -1 then fini := true; end if;
      #end do;
      #if fini = false then #now we check for last digit 9,8,7,6,5,4,3
        for k from 0 to floor( $\frac{b}{81}$ ) do
          for d2 from 0 to 8 do
            fini2 := false;
            for i from 3 to 9 while fini2 = false do
              if happy( $b + 2 \cdot d2 + 1 - k \cdot 81 + \text{onestep}((c + i^2) \bmod 100)$ ) = -1 then fini2 := true; end
                if;
              end do;
            if fini2 = false then print( $b, c, k, d2, \text{happy}(b + \text{onestep}(c + 2^2))$ ) end if;
            end do;
          end do;
        end if;
      end do;
    end do;
  end do;

```

Now we worry about possible carries in the M1 level when we calculated the 0,1,2,3,4,5,6 final digits. (six in a row after the carry.)

The problem is if $c > 963$ then the digit 5 could cause a carry into the M2 portion of M1.

```

> for b from 0 to 1500 do
  for c from 964 to 974 do #so these have the last digit 6 carry but not any others.
    fini := false;
    for i from 0 to 5 while fini = false do
      if happy( $b + \text{onestep}(c + i^2)$ ) = -1 then fini := true; end if;
    end do;

```

```

if fini = false then #now we check for last digit 5
  for k from 0 to floor( $\frac{b}{81}$ ) do
    for d2 from 0 to 8 do
      fini2 := false;
      for i from 6 to 6 while fini2 = false do
        if happy( $b + 2 \cdot d2 + 1 - k \cdot 81 + \text{onestep}((c + i^2) \bmod 100)$ ) = -1 then fini2 := true; end if;
      end do;
      if fini2 = false then print(b, c, k, d2, happy( $b + \text{onestep}(c + 2^2)$ )) end if;
    end do;
  end do;
  print(fini);

```

true

(18)

```

> for b from 0 to 1500 do
  for c from 975 to 983 do #so these have the last digit 5 carry but not any others.
    fini := false;
    for i from 0 to 4 while fini = false do
      if happy( $b + \text{onestep}(c + i^2)$ ) = -1 then fini := true; end if;
    end do;
    if fini = false then #now we check for last digit 5
      for k from 0 to floor( $\frac{b}{81}$ ) do
        for d2 from 0 to 8 do
          fini2 := false;
          for i from 5 to 6 while fini2 = false do
            if happy( $b + 2 \cdot d2 + 1 - k \cdot 81 + \text{onestep}((c + i^2) \bmod 100)$ ) = -1 then fini2 := true; end if;
          end do;
          if fini2 = false then print(b, c, k, d2, happy( $b + \text{onestep}(c + 2^2)$ )) end if;
        end do;
      end do;
      print(fini);

```

783, 982, 0, 7, 7

true

(19)

```

> for b from 0 to 1500 do
  for c from 984 to 990 do #so these have the last digit 5 and 4 carry but not any others.
    fini := false;
    for i from 0 to 3 while fini = false do
      if happy( $b + \text{onestep}(c + i^2)$ ) = -1 then fini := true; end if;
    end do;
    if fini = false then #now we check for last digit 5,4

```

```

for  $k$  from 0 to floor( $\frac{b}{81}$ ) do
  for  $d2$  from 0 to 8 do
     $fini2 := false$ ;
    for  $i$  from 4 to 6 while  $fini2 = false$  do
      if  $happy(b + 2 \cdot d2 + 1 - k \cdot 81 + onestep((c + i^2) \bmod 100)) = -1$  then  $fini2 := true$ ; end if;
    end do;
    if  $fini2 = false$  then  $print(b, c, k, d2, happy(b + onestep(c + 2^2)))$  end if;
  end do;
end do;
end do;
end do;
end do;
 $print(fin_i)$ ;

```

1107, 985, 0, 6, 5

1107, 985, 11, 4, 5

true

(20)

```

> for  $b$  from 0 to 1500 do
  for  $c$  from 991 to 995 do #so these have the last digit 5,4,3 carry but not any others.
     $fin_i := false$ ;
    for  $i$  from 0 to 2 while  $fin_i = false$  do
      if  $happy(b + onestep(c + i^2)) = -1$  then  $fin_i := true$ ; end if;
    end do;
    if  $fin_i = false$  then #now we check for last digit 5,4,3
      for  $k$  from 0 to floor( $\frac{b}{81}$ ) do
        for  $d2$  from 0 to 8 do
           $fini2 := false$ ;
          for  $i$  from 3 to 6 while  $fini2 = false$  do
            if  $happy(b + 2 \cdot d2 + 1 - k \cdot 81 + onestep((c + i^2) \bmod 100)) = -1$  then  $fini2 := true$ ; end if;
          end do;
          if  $fini2 = false$  then  $print(b, c, k, d2, happy(b + onestep(c + 2^2)))$  end if;
        end do;
      end do;
    end do;
  end do;
 $print(fin_i)$ ;

```

1022, 991, 0, 8, 4

true

(21)

```

> for  $b$  from 0 to 1500 do
  for  $c$  from 996 to 998 do #so these have the last digit 5,4,3,2 carry but not any others.
     $fin_i := false$ ;
    for  $i$  from 0 to 1 while  $fin_i = false$  do
      if  $happy(b + onestep(c + i^2)) = -1$  then  $fin_i := true$ ; end if;
    end do;
    if  $fin_i = false$  then #now we check for last digit 5,4,3,2

```

```

for  $k$  from 0 to floor( $\frac{b}{81}$ ) do
for  $d2$  from 0 to 8 do
   $fini2 := false$ ;
for  $i$  from 2 to 6 while  $fini2 = false$  do
if  $happy(b + 2 \cdot d2 + 1 - k \cdot 81 + onestep((c + i^2) \bmod 100)) = -1$  then  $fini2 := true$ ; end
if;
end do;
if  $fini2 = false$  then  $print(b, c, k, d2, happy(b + onestep(c + 2^2)))$  end if;
end do;
end do;
end if;
end do;
end do;
 $print(fin_i)$ ;

```

true

(22)

```

> for  $b$  from 0 to 1500 do
  for  $c$  from 999 to 999 do #so these have the last digit 5,4,3,2,1 carry but not any others.
     $fin_i := false$ ;
    for  $i$  from 0 to 0 while  $fin_i = false$  do
      if  $happy(b + onestep(c + i^2)) = -1$  then  $fin_i := true$ ; end if;
    end do;
    if  $fin_i = false$  then #now we check for last digit 5
      for  $k$  from 0 to floor( $\frac{b}{81}$ ) do
        for  $d2$  from 0 to 8 do
           $fini2 := false$ ;
          for  $i$  from 1 to 6 while  $fini2 = false$  do
            if  $happy(b + 2 \cdot d2 + 1 - k \cdot 81 + onestep((c + i^2) \bmod 100)) = -1$  then  $fini2 := true$ ; end
            if;
          end do;
          if  $fini2 = false$  then  $print(b, c, k, d2, happy(b + onestep(c + 2^2)))$  end if;
        end do;
        end do;
        end if;
        end do;
        end do;
         $print(fin_i)$ ;

```

true

(23)

Since the only b value with $b < 908$ with seven in a row after the carry is $b = 783$, and since the b value before the carry must exceed this, the value $b = 899$ is the only possibility before the carry.

We first verify if the string extends to eight in a row; we see here it does not.

```

> for  $b$  from 0 to 1500 do
  for  $c$  from 975 to 983 do #so these have the last digit 5 carry but not any others.
     $fin_i := false$ ;
    for  $i$  from 0 to 4 while  $fin_i = false$  do
      if  $happy(b + onestep(c + i^2)) = -1$  then  $fin_i := true$ ; end if;
    end do;

```

```

if  $fini = false$  then #now we check for last digit 5
  for  $k$  from 0 to  $\text{floor}\left(\frac{b}{81}\right)$  do
    for  $d3$  from 0 to 8 do
       $fini2 := false$ ;
      for  $i$  from 5 to 6 while  $fini2 = false$  do
        if  $\text{happy}(b + 2 \cdot d3 + 1 - k \cdot 81 + \text{onestep}((c + i^2) \bmod 100)) = -1$  then  $fini2 := true$ ; end
        if;
      end do;
      if  $fini2 = false$  then  $\text{print}(b, c, k, d3, \text{happy}(b + 2 \cdot d3 + 1 - k \cdot 81 + \text{onestep}((c + 7^2) \bmod 100)))$  end if;
    end do;
  end do;
  end do;
  end do;
  end do;
  end do;
   $\text{print}(fini)$ ;

```

783, 982, 0, 7, -1
true (24)

> $\text{minimalN}(899 - 6^2)$;

279999999999 (25)

We look for the digit preceding the k nines that will give the smallest N . The following shows that $d=8$ is best.

> $a := 2799999996999962$;

```

for  $d$  from 0 to 8 do  $\text{print}(d, \text{minimalbigN}(a - d^2))$  end do;

```

```

   $a := 2799999996999962$ 
  0, [789999, 34567901197525]
  1, [1888999, 34567901197525]
  2, [3689999, 34567901197525]
  3, [2689999, 34567901197525]
  4, [499999, 34567901197525]
  5, [3788899, 34567901197525]
  6, [8888899, 34567901197524]
  7, [899999, 34567901197524]
  8, [799999, 34567901197524]

```

(26)

> $\text{minimalN}(783 - 7^2)$

129999999999 (27)

so the value of $S(P+1)$ "after the carry" must be 1299999999997982, in particular, the last four digits of this must be 7982.

We do not need to test each value of k , since after the carry we know the last four digits have to be $c=7982$ and without the three last digits it must have $b=S(P2)=783$. In order for the last four digits to be 7982, we need $a - (d+1)^2 + d^2 - k \cdot 9^2 \bmod 10000 = 7982$, thus, $k=9037 \bmod 1000$.

> $a := 2799999996999962$; $d := 8$;

```
a := 2799999996999962
d := 8 (28)
```

```
> (a + 2·d + 1 - 7982) mod 10000
81
9037 (29)
```

```
> a := 2799999996999962;
ktop := iquo(a, 81);
a3 := (ktop - 9037) mod 10000;
kstart := ktop - a3;
a := 2799999996999962
ktop := 34567901197530
a3 := 8493
kstart := 34567901189037 (30)
```

```
> a := 2799999996999962;
d := 8; # d is the digit that precedes the k digits of nine at the end of N1.
ktop := iquo(a, 81);
a3 := (ktop - 9037) mod 10000;
for k from 3(ktop - a3) to 0 by -10000 do # k is the number of nines at the end of N1.
q := a + 2·d + 1 - k·81;
if onestep(q - 982) = 783 then print(k, q, onestep(q - 982)) end if;
end do;
```

```
a := 2799999996999962
d := 8
ktop := 34567901197530
a3 := 8493
32469382679037, 16997999997982, 783
```

```
Warning, computation interrupted
32464560929037, "done" (31)
```

```
> 34567901197524 + 5; %- 32469382679037;
34567901197529
2098518518492 (32)
```

```
> 1 + 2098518518492 + 1 + 32469382679037 + 1
34567901197532 (33)
```

We verify that this does indeed give 14 in a row.

$N = 7 \cdot (2098518518492 \text{ nines}) \cdot 8 \cdot (32469382679037 \text{ nines}) \cdot 2$

```
> a1 := 72 + 2098518518492·81 + 82 + 32469382679037·81;
for j from 2 to 9 do print(j, onestep(a1 + j2), happy(a1 + j2)) end do;
a1 := 2799999996999962
2, 1052, -1
3, 1030, 4
4, 1093, 7
5, 1093, 7
6, 1125, 5
7, 671, 5
```

8, 709, 5

9, 694, 8

(34)

> $a1 := 7^2 + 2098518518492 \cdot 81 + 9^2;$

for j **from** 0 **to** 7 **do** $print(j, onestep(a1 + j^2), happy(a1 + j^2))$ **end do;**

$a1 := 169979999997982$

0, 932, 7

1, 937, 8

2, 964, 8

3, 946, 8

4, 1009, 6

5, 847, 6

6, 863, 7

7, 808, -1

(35)