Strings of Consecutive Happy Numbers
23 Feb 2008
1 Mar 2008

The goal here is to show that the first string of six consecutive happy numbers begins with 78999999999999959999999996 which has
S(78999999999999959999999996) = 1739. Amazingly, this string continues to give seven in a row.

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.
```

$$f := n \rightarrow n^2 \quad \textbf{(1)}$$

```
> bs := 10;
      #this is the base, in case we ever want to investigate binary or ternary or any other base.
```

$$bs := 10 \quad \textbf{(2)}$$

```
> 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)                                                              (3)
    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
```

```
> 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 <= n <= 486 = 6*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, 999999] :

```
> 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]·10^(q - 5) + (10^(q - 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)

We first verify that 789999999999999599999999996 does indeed give seven happy numbers in a row.

```
> N := 789999999999999599999999996 :
  for j from -1 to 7 do
  print(N + j, happy(N + j));
  end do:
```

789999999999999599999999995, -1
789999999999999599999999996, 8
789999999999999599999999997, 8
789999999999999599999999998, 7
789999999999999599999999999, 5
789999999999999960000000000, 8
789999999999999960000000001, 4
789999999999999960000000002, 5
789999999999999960000000003, -1

(6)

```
> 25·81
```

2025

(7)

We now check to see if there is a smaller string of 6 consecutive happy numbers with the first one having the last two digits less than 94 (so no carry to the third digit.) Since our known solution has 25 digits, we need only check numbers N with S(N) < 25*81 = 2025.
We split N into N1*10^2 + b where b is a two digit number. Then S(N) = S(N1) + S(b). Let a = S(N1). So we need only check if a + S(b) is happy.

```
> for a from 1 to 2025 do
  c := 0;
  for b from 0 to 94 do
  if happy(a + onestep(b)) > 0 then c := c + 1; else c := 0; end if;
  if c > 4 then print(a, b, c) end if;
  end do; end do:
```

48, 92, 5
948, 92, 5
1130, 92, 5
1803, 70, 5
1835, 30, 5 **(8)**

Note that we did not find six in a row. Thus, if there are six in a row, there must be a carry to the third digit..

We look for all strings of 3 or more consecutive happy numbers where the final one ends with 99.

```
> A := [ ];
  for a from 2 to 2025 do
   if happy(a + onestep(99)) > 0 and happy(a + onestep(98)) > 0 and happy(a
      + onestep(97)) > 0
  then
  print(a, happy(a + onestep(96)), happy(a + onestep(95))) ;
   A := [op(A), a];
  end if;
  end do:
```

A := [ ]
487, -1, -1
493, -1, -1
1758, 7, -1 **(9)**

```
> A;
```

[487, 493, 1758] **(10)**

```
>
```

We will now find strings of three or more consecutive happy numbers starting with the 00 digits.

```
> for a from 1 to 2025 do
  if happy(a) ≥ 0 and happy(a + onestep(1)) ≥ 0 and happy(a + onestep(2)) ≥ 0 then
  print(a, happy(a + onestep(3)), happy(a + onestep(4)));
  end if;
  end do;
```

129, -1, -1
1029, -1, -1
1121, -1, -1
1184, -1, -1
1211, -1, -1

$$1299, -1, -1$$
$$1474, -1, -1$$
$$1574, -1, -1$$
$$1744, -1, 4$$
$$1754, -1, -1$$
$$1814, -1, -1$$
$$1929, -1, -1 \quad \textbf{(11)}$$

So we see there are never more than three in a row after the carry from 99, so there must be at least three in a row before the 00, namely, at least the 97, 98, and 99 ending digits must yield happy numbers.

We will now see how a sequence ending in 99 can be extended.

We illustrate our ideas with the a=1758 case.

Set N = N1.99 where S(N1) = 1758. Set N1 = N2.d.9...9 where d is a digit not equal 9, and there are exactly k digits of nine ending N1.
Clearly, k*81 cannot exceed 1758 so k is at most 21.
Then N+1 = N2.(d+1).0...0.00.
We have S(N1) = S(N2)+d^2+k*81 = 1758.
We have S(N+1) = S(N2) +(d+1)^2 = S(N1) +2d+1 - k*81 = 1758 + 2d+1-k*81

We will check all possible k values from 0 to 21 and all possible digits from 0 to 8 to see if the S(N+1) and S(N+2) are happy.

Since a= 1758 has four in a row, we need to calculate if the 00 and 01 give happy numbers. But for a= 487 or 493 values, we need three in a row, that is, the N+1 ending in 00, N+2 ending in 01 and N+3 ending in 02 all happy.

As we see here, only 1758 yields three consecutive happy results.

```
> a := 487;
  ktop := iquo(a, 81);
   for k from 0 to ktop do  # k is the number of nines at the end of N1.
  for d from 0 to 8 do   # d is the digit that precedes the k digits of nine.
  if happy(a + 2·d + 1 − k·81) > 0 and happy(a + 2·d + 1 − k·81 + 1) > 0 and happy(a
      + 2·d + 1 − k·81 + 4) > 0 then print(a, k, d, happy(a + 2·d + 1 − k·81 + 9), a − d^2
      − k·81) end if;
  end do; end do:
```

$$a := 487$$
$$ktop := 6 \quad \textbf{(12)}$$

```
> a := 487;
  ktop := iquo(a, 81);
   for k from 0 to ktop do  # k is the number of nines at the end of N1.
  for d from 0 to 8 do   # d is the digit that precedes the k digits of nine.
  if happy(a + 2·d + 1 − k·81) > 0 and happy(a + 2·d + 1 − k·81 + 1) > 0 and happy(a
      + 2·d + 1 − k·81 + 4) > 0 then print(a, k, d, happy(a + 2·d + 1 − k·81 + 9), a − d^2
      − k·81) end if;
```

```
end do; end do:
```

$$a := 487$$
$$ktop := 6 \quad \textbf{(13)}$$

```
> a := 1758;
ktop := iquo(a, 81);
 for k from 0 to ktop do  # k is the number of nines at the end of N1.
for d from 0 to 8 do   # d is the digit that precedes the k digits of nine.
if happy(a + 2·d + 1 − k·81) > 0 and happy(a + 2·d + 1 − k·81 + 1) > 0 then print(a,
    k, d, happy(a + 2·d + 1 − k·81 + onestep(2)), happy(a + 2·d + 1 − k·81
    + onestep(3)), a − d^2 − k·81) end if;
end do; end do:
```

$$a := 1758$$
$$ktop := 21$$
$$1758, 0, 6, -1, -1, 1722$$
$$1758, 4, 6, -1, -1, 1398$$
$$1758, 8, 5, 4, -1, 1085$$
$$1758, 14, 6, -1, -1, 588$$
$$1758, 18, 0, -1, 3, 300 \quad \textbf{(14)}$$

```
> minimalN(1722); evalf(log10(%) + 0);
```

$$2779999999999999999999$$
$$22.44404479 \quad \textbf{(15)}$$

```
> minimalN(1398); evalf(log10(%) + 4);
```

$$2779999999999999999$$
$$22.44404480 \quad \textbf{(16)}$$

```
> minimalN(1085); evalf(log10(%) + 8);
```

$$78999999999999$$
$$21.89762709 \quad \textbf{(17)}$$

```
> minimalN(588); evalf(log10(%) + 14);
```

$$277999999$$
$$22.44404480 \quad \textbf{(18)}$$

```
> minimalN(300); evalf(log10(%) + 18);
```

$$57899$$
$$22.76267106 \quad \textbf{(19)}$$

```
>
```

Our goal is now to find the smallest of these possibilities.
We see that when k=8, d=5, and S(N2) = 1085, we have one less digit than any other possibility, so it will give the smallest, regardless of the digit d.
Here N2 = 78999999999999.
The number N is the concatenation N2.d.9...9.96 which is 78999999999999.5.99999999.96.

Amazingly, this best case for six actually gives seven in a row.