July 10, 2007 at 15:17

1. Intro. I want to count the number of strings that
a) use the letters 0, 1, and 01,
b) contain each of the digits 0 and 1 exactly n times, and
c) whose consecutive letters share a digit.

2. The backtrack solution keeps track of how many 0s and how many 1s are yet to be used and what was the previous letter. For convenience the 0 will be encoded as 2^0, the 1 as 2^1, and the 01 as $2^0 + 2^1$.

```
#define max 37
#include <stdio.h>
#include <string.h>
  typedef long long i64;
  i64 cache[4][max + 1][max + 1];
  i64 solve(int previous, int zero_left, int one_left)
  {
    i64 &r = cache[previous][zero_left][one_left];
    if (r ≠ −1) return r;
    if (¬zero_left ∧ ¬one_left) return r = 1;
    r = 0;
    if (zero_left ∧ (previous & 1)) r += solve(1, zero_left − 1, one_left);
    if (one_left ∧ (previous & 2)) r += solve(2, zero_left, one_left − 1);
    if (zero_left ∧ one_left) r += solve(3, zero_left − 1, one_left − 1);
    return r;
  }
  int main( )
  {
    int n;
    scanf("%d", &n);
    if (n > max) {
      fprintf(stderr, "Sorry,␣but␣%d␣is␣too␣big␣for␣me.\n", n);
      return 1;
    }
    memset(cache, −1, sizeof (cache));
    for (int i = 1; i ≤ n; ++i) printf("n␣%d␣cnt␣%lld\n", i, solve(3, i, i));
  }
```

3. The numbers tend to grow quite quickly. They overflow a 64 bit signed integer for $n \geq 37$. The data suggests that the grow is $O(2^{1.8n})$. The sequence does not appear in the Online Encyclopedia of Integer Sequences.

4. Index.

cache: 2.
fprintf: 2.
i: 2.
i64: 2.
main: 2.
max: 2.
memset: 2.
n: 2.
one_left: 2.
previous: 2.
printf: 2.
r: 2.
scanf: 2.
solve: 2.
stderr: 2.
zero_left: 2.

STRINGS1

	Section	Page
Intro	1	1
Index	4	2