mathematical diary

entry #4 - 17.8.20
(from imo 2016 shortlist c1) i choose a secret binary word with n letters, and tell you the set of all $\binom{n}{k}$ binary words one gets by introducing exactly k errors to my chosen word (as well as the values n, k). how could you go about finding my secret word efficiently? do notice you cannot always tell my word - if $n = 2, k = 1$ and i picked 01 you'd get $\{11, 00\}$ and might think my secret word is 10.
well, say my word had x in bit number t. then you'll count $\binom{n-1}{k}$ words with x in bit number t, and $\binom{n-1}{k-1}$ words with $\overline{x} = 1 - x$ in bit number t. since these values only agree for $n = 2k$, if this is not the case, you can find my secret word in $n\binom{n}{k} \approx n^{k+1}$ time. now what about $n = 2k$? well you could always mistake my word for the opposite one (switching zeros and ones). we now sketch how you could go about finding all possible secret words, consisting exactly of this pair. assume wlog the first bit is zero. now, take a look at bit number t. if it were 0, you'd see $\binom{2k-2}{k}$ rows with 00 in the appropriate spaces, while if it were 1, you'd see $\binom{2k-2}{k-1}$ rows with 00 in the appropriate spaces, and since these are different numbers, you could tell the value of the bit.

entry #3 - 9.8.20
I recently pondered on how computers generate entropy. suppose we have a source for a truly random bit - how should one use it to generate a uniformly random number in $\{1, 2, 3\}$, and more generally in $[N]$? well, if $N = 2^k$ is a power of two, just use RB k times. if not, we can generate a random number in $[2^{\lceil \log_2(N) \rceil}]$, and if it's out of range, redo the whole process as many times as it takes until we get a number in $[N]$. this process is geometric, meaning on average it takes no more than $2 \log n$ calls to RB. i reckon need, on average, at least $\log n$, and it seems likely to get around with just that. so how can we improve the algorithm? say $N = 6$. if we don't get a number in $[6]$, what we do get is a random bit. so why not use it, and only pay two calls to RB instead of three next time. generalising this, split $[2^{\lceil \log_2(N) \rceil}] - [N]$ into its binary form - take out the largest power of two you can, and keep going. if our process did not stop, we still get k bits for next time if we fell into a pit of 2^k consecutive numbers. now, if I didn't make any calculation errors, we do achieve our bound of $\sim \log n$. right now I don't have much more to say about this, but I'm sure I'll get back to this topic.

entry #2 - 8.8.20
USAMO 2008 problem 6 reads: at a certain mathematical conference, every pair of mathematicians are either friends or strangers. At mealtime, every participant eats in one of two large dining rooms. Each mathematician insists upon eating in a room which contains an even number of his or her friends. Prove that the number of ways that the mathematicians may be split between the two rooms is a power of two.
fascinating. how does one go about solving this? a good guess would be to bash

using linear algebra over $\mathbf{F}_2$. letting x_i denote the boolean variable specifing which room mathematician i eats in, and noting that i, j sit in the same room iff $y_{ij} \equiv x_i + x_j + 1 = 1$, we get a system of linear equations $\forall i \sum_{j \sim i} y_{ij} = 0$. one must still understand why it has a solution. letting L denote the Laplacian of the freindship graph (adjacency matrix except the diagonal contains the vertex degrees), we want $\deg \in \text{image} L$. this is a special case of the fact the diagonal of a symmetric binary matrix is in its image. [which we can show via demonstrating it is perpendicular to the kernel]. in particular, we get an algorithm for dividing the mathematicians to their satisfaction. is it optimal?

entry #1 - 4.8.20
IMO 1979 problem 3 reads $\sim$ two circles intersect as A. two points, starting simultaneously from A, trace the circles, both in the same orientation and constant angular speed [they meet at A again after each revolved once around its circle] prove the existence of a point P, such that the two points are always equidistant from P.
seems pretty interesting, but I haven't practiced Euclidean geometry in ages, so I might as well bash this using complex numbers, I figured. I found more to be true, [in great part due to a simulation on desmos] and it goes as follows. let B denote the second intersection of the two circles, C_i their centers, and $M(t)$ the midpoint of the two moving points [as a function of the time variable t] then $M(t)$ travels in a circle - the circle whose center is the midpoint of the C_i's, and which contains A and B. Further, the line $\ell(t)$ between the two moving points always contains B. The problem is then solved, as P makes the diameter of said circle with B.