DAWN PUTNAM SESSION

Putnam 2002 — Papers A and B

Putnam 120 Archive

Contents

Putnam 2002 — Paper A **5**
2002 A1 5
2002 A2 7
2002 A3 9
2002 A4 12
2002 A5 14
2002 A6 17

Putnam 2002 — Paper B **21**
2002 B1 21
2002 B2 23
2002 B3 25
2002 B4 28
2002 B5 31
2002 B6 33

Putnam 2002 — Paper A

2002 A1

2002 A1. Let k be a fixed positive integer. The n-th derivative of $\frac{1}{x^k-1}$ has the form $\frac{P_n(x)}{(x^k-1)^{n+1}}$ where $P_n(x)$ is a polynomial. Find $P_n(1)$.

Fields: calculus, recursion **Position:** A1 — the opener **Answer:** $P_n(1) = (-k)^n n!$, obtained by differentiating the defining ansatz itself rather than the function, then evaluating the resulting two-term recursion at $x = 1$.

Layer 1 THE WHAT — surface analysis

The natural attempt. The instinct on seeing "the n-th derivative has this form" is to go and compute it: differentiate $f(x) = 1/(x^k - 1)$ directly, once, twice, three times, using the quotient rule, and read P_1, P_2, P_3 off the numerators as they appear. This works, in the sense that it produces correct polynomials — $P_1(x) = -kx^{k-1}$, and $P_2(x)$ a longer expression involving x^{2k-2} and x^{k-2} terms with binomial-looking coefficients. But each differentiation of a quotient with a growing-degree numerator and a $(n+1)$-th power denominator produces more terms than the last, and by $n = 3$ the algebra is already unpleasant enough that the student is doing it to find a pattern in the coefficients *as polynomials*, when the question only wants a single number: the value at $x = 1$.

Worse, $x = 1$ is precisely a root of $x^k - 1$, so there is no way to shortcut by plugging $x = 1$ into f itself — $f(1)$ is undefined, and so is every $\frac{P_n(x)}{(x^k-1)^{n+1}}$ if evaluated blindly, since the fraction as a whole represents a derivative that genuinely blows up there. The natural attempt either presses on computing full polynomials $P_n(x)$ symbolically until a closed form in n and k becomes visible by inspection (workable for $k = 1, 2$ by hand, painful beyond that), or tries to justify evaluating at $x = 1$ via a limit / L'Hôpital argument applied to the already-computed quotient — which requires knowing $P_n(x)$ explicitly near $x = 1$ anyway, so it buys nothing.

The model chain. Differentiate the *defining relation* itself, as an identity in x, rather than differentiating f from scratch each time. Writing $y = x^k - 1$ (so $y' = kx^{k-1}$) and $f^{(n)}(x) = P_n(x)/y^{n+1}$, differentiate both sides:

$$f^{(n+1)}(x) = \frac{P_n'(x)y^{n+1} - P_n(x)(n+1)y^n y'}{y^{2n+2}} = \frac{P_n'(x)y - (n+1)y'P_n(x)}{y^{n+2}}.$$

Comparing with $f^{(n+1)}(x) = P_{n+1}(x)/y^{n+2}$ gives the recursion

$$P_{n+1}(x) = (x^k - 1)P_n'(x) - (n+1)kx^{k-1}P_n(x),$$

valid for all x, as a polynomial identity — it was derived from an identity, not from a single numerical evaluation, so it is legitimate to now substitute $x = 1$ into it. At $x = 1$, $x^k - 1 = 0$, so the first term vanishes entirely, and $x^{k-1} = 1$, leaving

$$P_{n+1}(1) = -(n+1)k\,P_n(1).$$

The base case is $n = 0$: $f = 1/y = P_0(x)/y^1$, so $P_0(x) = 1$ and $P_0(1) = 1$. Unrolling the one-term recursion, $P_n(1) = (-k)^n n!$.

Layer 2 THE HOW — the bifurcation

Bifurcating point. Both approaches accept that they need a recursion or pattern connecting P_{n+1} to P_n; they split on *what gets differentiated*. The natural attempt differentiates $f(x)$ directly, over and over, producing the full polynomial $P_n(x)$ as a side effect each time. The model chain differentiates the ansatz $f^{(n)}(x) = P_n(x)/y^{n+1}$ as an equation, producing a recursion for P_{n+1} in terms of P_n that holds for every x — and only afterwards specialises that equation to $x = 1$.

> *I don't need the whole polynomial $P_n(x)$ — I only need its value at the single point $x = 1$. If I differentiate the equation $f^{(n)} = P_n/y^{n+1}$ itself, rather than differentiating f directly, I get a recursion that's valid for all x, and plugging $x = 1$ into a recursion is completely different from plugging $x = 1$ into f: the recursion is a statement about polynomials, which are defined everywhere, even though f itself is not defined at $x = 1$.*

> *The reason evaluating at $x = 1$ is not just legal but actually helpful is that $x^k - 1$ appears as an explicit multiplicative factor in the recursion for P_{n+1}. At $x = 1$ that factor is exactly zero — the same quantity whose vanishing makes $f(1)$ undefined is the quantity that kills the complicated term in the recursion and collapses two terms into one.*

Layer 3 THE WHY

Why does the singularity of f at $x = 1$ turn into a simplification rather than a problem, once the recursion is set up correctly? Because $P_{n+1}(x) = (x^k - 1)P_n'(x) - (n+1)kx^{k-1}P_n(x)$ is an identity between polynomials, and $x^k - 1$ is a polynomial too — it is well-defined and equal to zero at $x = 1$, in contrast to $1/(x^k - 1)$, which is undefined there. The object that blows up is the *quotient* $P_n/(x^k - 1)^{n+1}$; the numerator recursion, obtained before dividing, has no such singularity anywhere. This is the general shape of the trick: whenever a quantity is defined as a ratio and you are asked for a value at a point where the denominator vanishes, do not evaluate the ratio — differentiate or manipulate the identity that the numerator satisfies *before* the division happens, because the point that is singular for the ratio is very often an ordinary (or even especially clean) point for the numerator's own recursion.

There is a second thing worth extracting: the recursion $P_{n+1}(1) = -(n+1)kP_n(1)$ is a first-order linear recursion with a coefficient that itself depends on n (through the factor $(n+1)$), which is exactly the shape that unrolls to a factorial times a geometric term. Recognising $-(n+1)k$ as "this term contributes a factor of $n+1$ and a factor of $-k$ at each step" is what makes $P_n(1) = (-k)^n n!$ visible immediately, without solving a differential equation or guessing-and-checking a closed form against computed values of P_1, P_2, P_3.

Layer $+\alpha$ BACKTESTING

1. The first useful cue is that the question asks for a value *at a root of the denominator*: $x = 1$ is a root of $x^k - 1$. On seeing this, the thing to write down is: I cannot evaluate the function or the quotient there, but I might be able to evaluate a recursion for the numerator there, and the vanishing denominator may well simplify that recursion rather than break it.
2. That cue suggests differentiating the defining ansatz $f^{(n)} = P_n/y^{n+1}$ as an identity in x to get a recursion for P_{n+1} in terms of P_n, P_n', y, and y' — rather than differentiating f from first principles at each step.
3. Hesitation occurs right after computing $P_1(x) = -kx^{k-1}$ by direct quotient-rule differentiation, where the pull is to keep computing P_2, P_3 explicitly to spot a pattern in the full polynomials. This is reinforced away by noticing that only the value at $x = 1$ is wanted, which is a strong

signal that the polynomial structure away from $x = 1$ is irrelevant noise for this particular question.

4. The trap is treating $x = 1$ as an ordinary evaluation point of f or of the already-computed quotient $P_n(x)/(x^k - 1)^{n+1}$ — for instance, trying to resolve the $0/0$-looking expression by L'Hôpital's rule applied to the quotient itself, which requires exactly the explicit polynomial the shortcut was meant to avoid computing.

5. Recognise this family faster by treating "value of a numerator polynomial in a quotient-derivative identity, evaluated at a root of the denominator" as a standing signal to differentiate the ansatz, not the function, and to substitute the special point into the resulting polynomial recursion, never into the original quotient.

6. Reusable rule: when $f^{(n)}(x) = P_n(x)/g(x)^{n+1}$ and a value of P_n is wanted at a root of g, differentiate the ansatz itself to obtain $P_{n+1} = gP_n' - (n+1)g'P_n$, then substitute the root directly into this recursion — the factor of g will vanish and typically collapse the recursion to a single surviving term, turning a singularity of f into the very thing that makes the recursion solvable in closed form.

▷ **Cue:** a value of a numerator polynomial in a quotient-derivative identity $f^{(n)} = P_n/g^{n+1}$ is wanted at a root of g $\longrightarrow$ **Move:** differentiate the defining ansatz itself to get the recursion $P_{n+1} = gP_n' - (n+1)g'P_n$, then evaluate that recursion (not the original quotient) at the root, letting the g-term vanish

▷ **Cue:** a first-order recursion has a coefficient that depends linearly on the step index n $\longrightarrow$ **Move:** unroll it directly rather than guessing a closed form from computed terms; a coefficient like $-(n+1)k$ signals a product telescoping to $n!$ times a geometric factor

2002 A2

2002 A2. Given any five points on a sphere, show that some four of them must lie on a closed hemisphere.

Fields: geometry, pigeonhole **Position:** A2 **Answer:** Some four of the five points lie on a common closed hemisphere; proved by routing a great circle through two of the five points and pigeonholing the remaining three into the two resulting closed hemispheres.

Layer 1 THE WHAT — surface analysis

The natural attempt. A natural first move is to treat the five points symmetrically and reach for an averaging argument. A great circle divides the sphere into two closed hemispheres; pick a "random" great circle and ask how many of the five points fall in a given hemisphere. Generically each point lies strictly inside one hemisphere or the other with probability $1/2$ (points landing exactly on the dividing circle are a measure-zero event for a random circle), so the expected number of the five points in a random hemisphere is $5 \cdot \frac{1}{2} = 2.5$. Since the expectation exceeds 2, some choice of hemisphere contains at least 3 of the five points.

This is a genuine result, and it is tempting to feel finished — but re-reading the problem, it asks for *four* points on a common hemisphere, not three. The averaging argument, however it is refined (trying a smarter distribution over great circles, weighting the choice, and so on), tops out at guaranteeing $\lceil 5/2 \rceil = 3$ points in some hemisphere, because it treats all five points as interchangeable and never uses any relationship between specific points to do better. No amount of tuning a symmetric, generic argument closes the gap from 3 to 4: the method is one point short of what is needed, and it is short for a structural reason, not a computational one — it never uses the fact that hemispheres are *closed*, i.e., that points lying exactly on the boundary circle belong to both hemispheres at once.

The model chain. Choose two of the five points, A and B, and draw a great circle through both of them (any two points on a sphere lie on at least one great circle; if antipodal, on infinitely

many, and any one will do). This great circle is the shared boundary of two closed hemispheres, H_1 and H_2, and because it is a *closed* hemisphere condition, both A and B — lying exactly on the boundary — belong to both H_1 and H_2 simultaneously. They are not "up for grabs" between the two hemispheres; they are already members of each.

The remaining three points must each lie in H_1 or in H_2 (or on the boundary, which only helps further). By the pigeonhole principle, since 3 points are distributed among only 2 hemispheres, at least two of the three lie in the same closed hemisphere, say H_1. Then H_1 contains those two points together with A and B — four points on one closed hemisphere, as required.

Layer 2 THE HOW — the bifurcation

Bifurcating point. Both approaches use a dividing great circle and a counting argument to force points together. They split on how the great circle is chosen. The natural attempt chooses the great circle generically (effectively at random, symmetric in all five points) and counts how the whole set of five distributes across it. The model chain chooses the great circle *through two of the given points on purpose*, sacrificing symmetry so that those two points are banked as boundary members of both hemispheres before any counting begins, and only then pigeonholes the three points left over.

> *If I force the great circle itself to pass through two of my points, those two points belong to both closed hemispheres for free — I don't need to win them by counting at all. That means pigeonhole only has to do work on the remaining three points, and pigeonholing three points into two boxes guarantees two of them land in the same box, which is exactly the extra point I was missing when I tried to treat all five points symmetrically.*

Layer 3 THE WHY

Why does routing the great circle through two given points actually help, rather than just being a different way to draw the same line? Because the problem says *closed* hemisphere, and "closed" is doing real work: it means the boundary great circle itself is included in both of the hemispheres it separates. A point lying exactly on that circle is therefore a member of both closed sets at once, an asymmetric bonus unavailable to points that fall strictly on one side. If the problem instead asked about *open* hemispheres, a point on the great circle would belong to neither, and this construction would gain nothing — the two chosen points would simply be points to be placed, exactly like the other three, and the whole argument collapses back to the weaker averaging bound. The word "closed" in the statement is not a technicality to be noted in passing; it is the single fact that makes "spend two points for free by putting them on the boundary" a legal move.

The counting step is then just the ordinary pigeonhole principle, but it is worth being precise about why it needs only three points and two boxes: pigeonhole guarantees a box with at least $\lceil 3/2 \rceil = 2$ items whenever 3 items are placed in 2 boxes. Two boxes is the count that survives specifically because two points were removed from the pool in advance — with all five points still to be placed, pigeonhole into two boxes would only guarantee $\lceil 5/2 \rceil = 3$, exactly the weaker bound the natural attempt reached. The two moves — banking two points on the boundary, then pigeonholing the rest — are not independent tricks; the first is what shrinks the pigeonhole problem enough for the second to deliver the extra point.

Layer $+\alpha$ BACKTESTING

1. The first useful cue is the word "closed" attached to "hemisphere." On seeing it, write down immediately: points lying exactly on the dividing great circle belong to *both* closed hemispheres, so they can be banked for free rather than fought over by a counting argument.

2. That cue suggests pigeonhole combined with a deliberately non-generic choice of the dividing great circle — one routed through specific given points rather than chosen at random or symmetrically.

3. Hesitation occurs right after the averaging argument yields "some hemisphere contains at least 3 points" and the problem demands 4. The pull is to keep refining the same symmetric, random-circle argument (a smarter probability distribution, a cleverer weighting) rather than abandoning symmetry outright. This is reinforced away by asking directly: does the argument so far use the fact that the hemisphere is closed? If not, it is leaving a resource on the table.

4. The trap is treating all five points interchangeably throughout, because it feels more natural and more "general" — but a fully symmetric argument can only ever prove a fully symmetric bound, and $\lceil 5/2 \rceil = 3$ is the ceiling of what symmetry alone can deliver here, one short of the required 4.

5. Recognise this family faster by treating any appearance of "closed" (region, hemisphere, half-space, interval) in a pigeonhole-flavoured geometry problem as a specific invitation to route the dividing structure through given data points, converting them into boundary members of every piece that touches the boundary.

6. Reusable rule: in a covering or partition pigeonhole problem where the pieces are stated to be closed, deliberately anchor the partition's boundary through some of the given points — this removes them from the pool that must be pigeonholed, shrinking the pigeonhole problem to the remaining points and often supplying exactly the extra count that a fully symmetric, generic argument cannot reach.

▷ **Cue:** a covering or partition problem states the pieces are *closed* and asks to guarantee a certain number of given points land together $\longrightarrow$ **Move:** route the dividing structure (great circle, hyperplane, interval endpoint) through some of the given points so they count as members of every piece touching the boundary, for free, then pigeonhole only the points left over

▷ **Cue:** a symmetric averaging or random-partition argument yields a bound exactly one short of what is required $\longrightarrow$ **Move:** abandon symmetry deliberately — anchor the partition through specific given points instead of choosing it generically, since a fully symmetric argument can only ever prove a fully symmetric (and here, insufficient) bound

2002 A3

2002 A3. Let $n \geq 2$ be an integer and T_n be the number of non-empty subsets S of $\{1, 2, 3, \ldots, n\}$ with the property that the average of the elements of S is an integer. Prove that $T_n - n$ is always even.

Fields: combinatorics, involutions **Position:** A3 **Answer:** $T_n - n$ is even: the n singleton sets $\{1\}, \ldots, \{n\}$ trivially qualify, and every other qualifying subset pairs off under the fixed-point-free involution "insert or delete the mean," so the remainder of T_n is even.

Layer 1 THE WHAT — surface analysis

The natural attempt. The claim is about parity, but the eye is drawn to the definition of T_n as a *count*, and the natural instinct is to try to compute it, or at least a formula for it, directly. Fix a subset size k and ask how many k-element subsets of $\{1, \ldots, n\}$ have sum divisible by k — this is a divisibility condition on a sum over a k-subset, exactly the kind of thing a roots-of-unity filter handles: write the indicator of "$k \mid \mathrm{sum}(S)$" as $\frac{1}{k}\sum_{j=0}^{k-1} \omega^{j \cdot \mathrm{sum}(S)}$ for ω a primitive k-th root of unity, sum this over all k-subsets S via the elementary symmetric structure of $\prod_{i=1}^{n}(1 + \omega^{ji}x)$, and extract the coefficient of x^k.

This machine genuinely works, size by size, and in principle summing the resulting expression over $k = 1, \ldots, n$ produces T_n. But it is a large amount of machinery — generating functions in two variables, roots of unity, careful bookkeeping of which k pairs with which root — deployed to answer a question that only asks about the parity of $T_n - n$, not the value of T_n

itself. The divisor k changes with the subset size, so there is no single, uniform root of unity that handles every size at once; the computation has to be redone, structurally, for each k, and even after all of that work the parity of the resulting (probably unwieldy) closed form still has to be extracted separately. This is solving a strictly harder problem — exact enumeration — than the one that was actually asked.

The model chain. Separate out the n singleton subsets $\{1\}, \{2\}, \ldots, \{n\}$: each trivially has integer average (the element itself), so all n of them qualify, and they are exactly the n being subtracted in the statement $T_n - n$. It remains to show that the count of *all other* qualifying subsets is even.

Define a map φ on the qualifying subsets other than the singletons. Given such an S, with integer average $m = \frac{1}{|S|}\sum_{s \in S} s$:

- if $m \notin S$, set $\varphi(S) = S \cup \{m\}$;
- if $m \in S$ (so $|S| > 1$, since S is not a singleton), set $\varphi(S) = S \setminus \{m\}$.

In either case $\varphi(S)$ still has average exactly m: adding a copy of the mean to a set does not change the mean, since $\frac{|S|m+m}{|S|+1} = m$, and removing one instance of the mean does not change it either, since $\frac{|S|m-m}{|S|-1} = m$ (valid because $|S| > 1$ guarantees $S \setminus \{m\}$ is non-empty). So $\varphi(S)$ is again a qualifying subset.

The map φ is an involution: applying it twice returns S, because whether m belongs to the current set flips exactly once each time φ is applied. It is also fixed-point-free and never produces a singleton or the empty set from a non-singleton input, and it never maps two different non-singleton qualifying sets to the same singleton or to each other's singleton, because $\varphi(S) = S$ is impossible (S always changes by exactly one element) and $\varphi(S)$ can only be a singleton $\{m\}$ if $S = \{m\} \cup \{m\}$, which is not a valid input since S itself is required to have more than one element to be in this pool at all — so φ restricts to a fixed-point-free involution entirely within the non-singleton qualifying subsets. It therefore partitions them into pairs, so their number is even. Hence $T_n - n$, the count of non-singleton qualifying subsets, is even.

Layer 2 THE HOW — the bifurcation

Bifurcating point. Both approaches recognise that the condition "average is an integer" is the whole content of the problem. They split on what is done with that recognition: the natural attempt tries to *enumerate* the qualifying sets exactly, size by size, via a divisibility filter. The model chain never counts them at all — the phrase "$T_n - n$ is always even" is a parity claim, not a value, and the model chain instead constructs an explicit pairing (involution) on the qualifying sets, sized so that its only unpaired elements are precisely the n singletons already subtracted off.

> *The question only asks about parity, not the exact count of T_n — so instead of computing T_n I should look for a way to pair up almost all the qualifying subsets. A set that pairs up cleanly with another distinct set contributes nothing to the parity of the total; I only need to pin down exactly which sets are left over, unpaired, and the statement $T_n - n$ is telling me in advance that there should be exactly n of them.*

> *If S has integer average m, then adding or removing the value m itself doesn't change the average — so m is a natural pivot for turning S into a different set of the same average. That gives me a candidate pairing move for free, and the only sets where this move can't produce something genuinely different from a non-singleton set are the singletons $\{m\}$ themselves, since removing m from $\{m\}$ leaves the empty set, which the problem has already excluded by requiring S non-empty.*

Layer 3 THE WHY

Why is "insert or delete the mean" the right pivot, rather than some other move? Because the arithmetic mean has a property that most other statistics of a set (the median, say, or the max) do not: it is exactly preserved under adding or removing one further copy of itself. Algebraically, if $\bar{S} = \frac{1}{|S|}\sum S = m$, then $\overline{S \cup \{m\}} = \frac{|S|m+m}{|S|+1} = m$ and $\overline{S \setminus \{m\}} = \frac{|S|m-m}{|S|-1} = m$ whenever $|S| > 1$. Both identities are immediate from linearity of the sum and require nothing beyond the definition of average — but they are precisely what makes "toggle membership of m" a move that stays inside the set of qualifying subsets, rather than a move that would have to be followed by a separate re-verification that the average is still an integer.

That the map is an involution — and not merely some pairing — follows from the same fact stated the other way: whether m belongs to S is a single bit, and the map flips exactly that bit and nothing else, so applying it twice restores the original set. And the map's fixed/exceptional set is forced to be exactly the singletons because "$S \setminus \{m\}$" is only undefined (or rather, produces the empty set, which is disallowed) precisely when $S = \{m\}$ — there is no other way for the toggle to fail, because for $|S| > 1$, $S \setminus \{m\}$ is guaranteed non-empty and $S \cup \{m\}$ is always non-empty. So the "nonempty subsets" hypothesis in the problem statement is not incidental bookkeeping: it is the exact condition that makes the singletons, and only the singletons, the involution's exceptional set, matching the "$-n$" in the claim term for term.

Layer $+\alpha$ BACKTESTING

1. The first useful cue is that the target claim is a parity statement — "$T_n - n$ is always even" — not a formula for T_n. On seeing this, write down: this is a claim about counting modulo 2, so the right tool is an involution pairing up (almost) all the objects being counted, not an exact enumeration.

2. That cue suggests constructing a fixed-point-free involution on the set of qualifying subsets, deliberately designed so its exceptional (unpaired) set matches the n objects already being subtracted in the claim.

3. Hesitation occurs right after noticing that the average condition's divisor depends on the subset's own size k, which makes a roots-of-unity or generating-function filter feel like the obvious tool, since that is the standard machine for "sum divisible by k" conditions. This is reinforced away by re-reading the question and asking: do I actually need the value of T_n, or only its parity relative to n? — and then asking what natural quantity attached to S (here, the mean) stays fixed under a simple insert/delete move.

4. The trap is solving a strictly harder problem than the one asked: computing or closely approximating T_n exactly via heavy generating-function machinery, when the actual claim only concerns T_n's parity relative to n, a far weaker and more structurally accessible statement.

5. Recognise this family faster by treating any "prove this count is even" or "prove these two counts differ by an even number" statement as a standing invitation to build an explicit involution first, and to look specifically for an invariant of the objects (here, the arithmetic mean of a subset) that survives a natural single-element insert or delete.

6. Reusable rule: to prove a count X minus a count Y is even, search for a fixed-point-free involution on the X-objects whose exceptional set is exactly the Y-objects; build the involution's single move out of whatever statistic defines the objects (mean, sum, a distinguished element) and check that adding or removing a value equal to that statistic leaves the statistic, and hence the qualifying condition, unchanged.

▷ **Cue:** asked to prove a count is even, or that two counts differ by an even number $\longrightarrow$ **Move:** construct a fixed-point-free involution on the counted objects, isolating the exact exceptional/fixed set that matches the claimed difference, rather than computing the count exactly

▷ **Cue:** objects are subsets defined by a condition on their arithmetic mean (or another statistic

invariant under self-insertion) $\longrightarrow$ **Move:** use insertion/deletion of the mean itself as the pairing move, since the mean is unchanged by adding or removing one further copy of itself

2002 A4

2002 A4. In Determinant Tic-Tac-Toe, Player 1 enters a 1 in an empty 3×3 matrix. Player 0 counters with a 0 in a vacant position, and play continues in turn until the 3×3 matrix is completed with five 1's and four 0's. Player 0 wins if the determinant is 0 and player 1 wins otherwise. Assuming both players pursue optimal strategies, who will win and how?

Fields: combinatorial games; linear algebra (structure of the determinant) **Position:** A4 — mid-paper. No theory is needed beyond the definition of the determinant; the difficulty is choosing a goal for Player 0 that is small enough to force. **Answer:** Player 0 wins. Strategy: answer Player 1's first 1 in a cell sharing neither its row nor its column, then build towards a zero row, zero column or 2×2 block of zeros; Player 1 cannot block all three.

Layer 1 THE WHAT — surface analysis

The natural attempt. You play a few games on scrap paper. Player 1 puts a 1 in the top-left corner. As Player 0 you want the determinant to vanish, and the first picture of "determinant zero" you own is "two rows equal" or "a row of zeros". So you drop your 0 next to the 1, in row 1, planning to fill that row with zeros — and immediately see that you can't, because the 1 is already in it. You try instead to make two rows equal: that needs you to control where the 1's go, and the 1's belong to your opponent. You start expanding the determinant for particular positions, $a(ei - fh) - b(di - fg) + c(dh - eg)$, and every partially filled board has several empty cells whose values change the answer. The game tree has 9! move orders; symmetry and transpositions cut it down, but you are still left with hundreds of positions. You can grind it (a computer confirms Player 0 wins), but by hand, without a target, you drown. Worse: your instinctive first reply in the same row or column as the 1 is a *losing* move. A full search shows that after a 1 at A_{11}, the replies $A_{12}, A_{13}, A_{21}, A_{31}$ all lose for Player 0, and only the four cells in the lower-right 2×2 block win.

The model chain. Write A_{ij} for the cell in row i, column j.

Target. Player 0 will force one of: a row of three zeros, a column of three zeros, or a 2×2 submatrix of zeros (rows $\{r, r'\}$, columns $\{c, c'\}$). Each forces $\det = 0$ whatever the other entries are. Expanding along a zero row or zero column gives 0. For a 2×2 zero block in rows r, r', both of those rows have nonzero entries only in the single remaining column, so they are scalar multiples of the same unit vector, hence linearly dependent.

Symmetries. Permuting rows, permuting columns and transposing all change det at most by a sign, so they preserve "$\det = 0$". They also map zero rows, zero columns and zero 2×2 blocks to the same kinds of object.

Play. By a row and a column permutation, Player 1's first 1 is at A_{11}. Player 0 plays A_{22}. Player 1 makes a second move; it occupies at most one of A_{23}, A_{32}, and transposition swaps these two while fixing A_{11}, A_{22}, so assume A_{23} is vacant. Player 0 plays A_{23}. Row 2 is now $(A_{21}, 0, 0)$.

If Player 1's third move is not A_{21}, Player 0 plays A_{21} and row 2 is zero. So Player 1 has played A_{11}, A_{21} and one more cell $X \in \{A_{12}, A_{13}, A_{31}, A_{32}, A_{33}\}$ (in some order).

Choose $i \in \{1, 3\}$ with A_{i2}, A_{i3} both empty, and $j \in \{2, 3\}$ with A_{1j}, A_{3j} both empty. This is possible: if X is in row 1, take $i = 3$ and j the column of $\{2, 3\}$ not containing X; if X is A_{32} or A_{33}, take $i = 1$ and j likewise; if $X = A_{31}$, take $i = 1$, $j = 2$. Player 0 plays A_{ij}. Now there are two threats. Column j has zeros at A_{2j}, A_{ij} and one empty cell $A_{i'j}$ (i' the other element of $\{1, 3\}$). The block rows $\{2, i\}$, columns $\{2, 3\}$ has three zeros and one empty cell $A_{ij'}$ (j' the

other element of $\{2,3\}$). These empty cells are different (different rows). Player 1 can fill only one; Player 0 fills the other on the fourth 0. $\det = 0$. □

Caveat. Verified independently by exhaustive minimax over all move sequences: Player 1 cannot force a nonzero determinant, and Player 0 can in fact force a zero row, zero column or zero 2×2 block. The same search shows that after a 1 at A_{11}, Player 0's winning replies are exactly $A_{22}, A_{23}, A_{32}, A_{33}$.

Layer 2 THE HOW — the bifurcation

Bifurcating point. Both chains agree that Player 0 wants $\det = 0$. They split at the question (†) "which kind of zero determinant do I aim for?" The natural attempt aims at "two rows equal", which depends on where the opponent's 1's go. The model aims only at configurations made entirely of Player 0's own zeros, which force $\det = 0$ regardless of the rest of the board.

> *I only control the zeros. So my target has to be a pattern of zeros that kills the determinant whatever the 1's and blanks turn out to be. Which zero patterns do that? A zero row, a zero column, a* 2×2 *zero block. Four zeros is enough for any of them.*

> *My first 0 should keep as many of those patterns alive as possible. The 1 at* A_{11} *has already killed row 1, column 1, and every* 2×2 *block through* A_{11}*. The patterns still alive all live in rows* $2,3$ *and columns* $2,3$ *— so I play there.*

Layer 3 THE WHY

Why is "structural zeros" the natural target rather than a trick? Go to the definition. The 3×3 determinant is

$$\det A = \sum_{\sigma \in S_3} \operatorname{sgn}(\sigma)\, A_{1\sigma(1)} A_{2\sigma(2)} A_{3\sigma(3)},$$

six terms, one for each way of choosing one cell in every row and every column (a *transversal*). Player 0's pieces are zeros, and a zero does exactly one thing: it kills every term through it. So the honest question for Player 0 is: can I place zeros so that every one of the six transversals passes through a zero? If so, every term is 0 and $\det = 0$ no matter what else happens. That is a game Player 0 can analyse, because it depends only on Player 0's own cells.

Which zero sets meet every transversal? A transversal uses all three rows, so it meets any full row; likewise any full column. A 2×2 zero block on rows $\{r, r'\}$, columns $\{c, c'\}$: a transversal must put the entries of rows r, r' into two distinct columns, and only one column lies outside $\{c, c'\}$, so one of them lands in the block. In general a $p \times q$ zero block with $p + q \geq n + 1$ does this (the Frobenius–König theorem, which also says these are the only minimal ways), and for $n = 3$ the options are 1×3, 3×1 and 2×2. That is where the three targets in Kedlaya and Ng's solution come from. Their solution states the three patterns and proceeds; this is why those three and no others.

Now the opening. Player 1's 1 at A_{11} is not a threat to make the determinant nonzero — it is a *block*: it poisons row 1, column 1, and the four 2×2 blocks containing A_{11}. Count what survives: rows 2 and 3, columns 2 and 3, and the 2×2 blocks avoiding A_{11} — that is, those using rows $\{2,3\}$ or columns $\{2,3\}$. Every surviving pattern passes through the lower-right square $\{2,3\} \times \{2,3\}$, and a 0 at A_{22} lies on row 2, column 2, and several live blocks at once. A reply at A_{12} lies on nothing alive except column 2 and one block. The computer search confirms the exact line: the four cells of the lower-right square win, the other four lose. This is ordinary tic-tac-toe thinking — play the cell that lies on the most live lines — once you have identified what the lines are.

Paul Cheng's note in the official solutions makes that literal: arrange the nine cells in a grid so that the six transversals are the three rows and three columns of the grid, and Player 0 is trying to put a zero on every line. That is the WHY at its most transparent: the determinant, read through its definition, *is* a tic-tac-toe board with six lines.

The rest of the model chain is double-threat construction, the only way to force anything in a game where the opponent may always block one cell. The choice of A_{ij} is exactly the cell lying on two live, nearly-complete patterns at once.

Layer $+\alpha$ BACKTESTING

1. The first useful cue is "Player 0 wins if the determinant is 0" together with "Player 0 counters with a 0". Write down $\det A = \sum_\sigma \operatorname{sgn}\sigma \prod A_{i\sigma(i)}$ and the sentence "each of my zeros kills the terms through it".

2. Reduce the game to covering: find zero patterns that meet every transversal (row, column, 2×2 block), then play for a double threat among them.

3. Hesitation comes at Player 0's first reply, where the instinct is to play near the 1. It is reinforced away by listing which target patterns the 1 has killed and playing in the region where all survivors live.

4. The trap is aiming at $\det = 0$ through relations such as equal or proportional rows, which need the opponent's cooperation, and then expanding the determinant position by position. A second trap: the "adjacent" reply A_{12} loses.

5. Slogan: *in a game, choose a win condition built only from your own pieces; if the scoring function is a sum of products, a zero piece kills every product through it.*

6. Decision rule: **when a quantity is a sum of products over a family of "lines" (determinants, permanents, polynomial expansions), a forced-zero or forced-cancellation argument becomes a covering problem — hit every line — and a game on it becomes tic-tac-toe on those lines.** Then use the tic-tac-toe heuristics: occupy cells on many live lines, win by a double threat.

▷ **Cue:** the win condition is "the determinant is zero" and your pieces are zeros $\longrightarrow$ **Move:** expand det as a sum over transversals and aim to put a zero on every transversal

▷ **Cue:** you need a pattern of zeros that forces $\det = 0$ for every filling of the other cells $\longrightarrow$ **Move:** use a $p \times q$ zero block with $p + q \geq n + 1$ (for 3×3: a zero row, a zero column, or a 2×2 zero block)

▷ **Cue:** an opponent's first piece in a positional game $\longrightarrow$ **Move:** list which of your target lines it has killed; reply in the cell lying on the most surviving lines

▷ **Cue:** the opponent can always block one cell per move $\longrightarrow$ **Move:** construct a move that creates two threats whose completing cells are different

▷ **Cue:** a game on a matrix with row/column structure $\longrightarrow$ **Move:** reduce cases by row permutations, column permutations and transposition, which preserve the win condition

2002 A5

2002 A5. Define a sequence by $a_0 = 1$, together with the rules $a_{2n+1} = a_n$ and $a_{2n+2} = a_n + a_{n+1}$ for each integer $n \geq 0$. Prove that every positive rational number appears in the set

$$\left\{ \frac{a_{n-1}}{a_n} : n \geq 1 \right\} = \left\{ \frac{1}{1}, \frac{1}{2}, \frac{2}{1}, \frac{1}{3}, \frac{3}{2}, \ldots \right\}.$$

Fields: number theory; recursive sequences (Stern's diatomic sequence, Calkin–Wilf tree) **Position:** A5 — late paper, but short once the right statement is chosen. The whole difficulty is strengthening the claim into something that inducts. **Answer:** Every coprime pair (r, s) occurs as (a_n, a_{n+1}) for some $n \geq 0$, by induction on $r + s$ running the subtractive Euclidean algorithm backwards; hence $r/s = a_{m-1}/a_m$ with $m = n + 1$.

Layer 1 THE WHAT — surface analysis

The natural attempt. You compute. $a_0 = 1$, $a_1 = a_0 = 1$, $a_2 = a_0 + a_1 = 2$, $a_3 = a_1 = 1$, $a_4 = a_1 + a_2 = 3$, $a_5 = a_2 = 2$, $a_6 = a_2 + a_3 = 3$, $a_7 = 1$, $a_8 = 4$, $a_9 = 3$, $a_{10} = 5$, $a_{11} = 2$, $a_{12} = 5$, $a_{13} = 3$, $a_{14} = 4$, $a_{15} = 1$. The ratios match the displayed set. Now you look for a formula: given p/q, which n has $a_{n-1}/a_n = p/q$? You notice $a_{2^k-1} = 1$ and $a_{2^k} = k+1$, so $1/(k+1)$ appears at $n = 2^k$. You try n in binary and see that the digits seem to encode something like a continued fraction, but pinning down the rule takes a page and you are not sure of it. Alternatively you try to prove "every positive integer r appears as some a_n" first, then "for each r, every s coprime to r appears next to it" — and you find that the first statement does not help with the second: knowing that $a_n = r$ tells you nothing about a_{n+1}. The wall is that you are trying to track single terms, and the question is about two terms at once.

The model chain. *Claim.* For every pair of positive integers r, s with $\gcd(r,s) = 1$ there is $n \geq 0$ with $(a_n, a_{n+1}) = (r,s)$.

This suffices: any positive rational is r/s with $\gcd(r,s) = 1$, and then $r/s = a_{m-1}/a_m$ for $m = n+1 \geq 1$.

Two transitions. For $n \geq 0$ the rules give

$$(a_{2n+1}, a_{2n+2}) = (a_n,\ a_n + a_{n+1}), \qquad (a_{2n+2}, a_{2n+3}) = (a_n + a_{n+1},\ a_{n+1}),$$

the second using $a_{2n+3} = a_{2(n+1)+1} = a_{n+1}$. So if the pair (x,y) occurs at position n, the pairs $(x, x+y)$ and $(x+y, y)$ occur at positions $2n+1$ and $2n+2$.

Proof of the claim, by strong induction on $r+s$. If $r+s = 2$ then $r = s = 1$ and $(a_0, a_1) = (1,1)$. Suppose $r+s > 2$ with $\gcd(r,s) = 1$. Then $r \neq s$, since $r = s$ with $\gcd = 1$ forces $r = s = 1$.

- If $r > s$: the pair $(r-s, s)$ has positive entries, $\gcd(r-s, s) = \gcd(r,s) = 1$, and smaller sum r. By hypothesis $(a_n, a_{n+1}) = (r-s, s)$ for some n, and then $(a_{2n+2}, a_{2n+3}) = (r,s)$.
- If $r < s$: likewise $(r, s-r)$ occurs at some n, and $(a_{2n+1}, a_{2n+2}) = (r,s)$.

This completes the induction. □

Remark (uniqueness). For $m \geq 1$, a_m and a_{m+1} are positive, and the displayed transitions show that at odd m the pair is increasing and at even $m \geq 2$ it is decreasing; so the parent position and parent pair are determined by (r,s). The same induction then shows each coprime pair occurs at exactly one position: every positive rational appears exactly once in the list.

> **Caveat.** Verified numerically: every coprime (r,s) with $r+s \leq 19$ appears among a_{n-1}/a_n, $n < 2^{18}$; consecutive terms are coprime there; and no ratio repeats for $n < 2^{16}$. The proof above is complete; the computation is a guard only.

Layer 2 THE HOW — the bifurcation

Bifurcating point. Both chains compute the first sixteen terms. They split at (†): what to prove. The natural attempt tries to locate each rational by a formula in n, or to prove facts about single terms a_n. The model rewrites the recursion as a rule acting on the pair (a_n, a_{n+1}) and proves that every coprime pair is reachable.

> *The set is built from a_{n-1}/a_n — two neighbours. So my object is the pair (a_n, a_{n+1}), not the term a_n. What does the recursion do to a pair?*

> *From (x,y) I get $(x, x+y)$ and $(x+y, y)$. Run backwards, that is subtract the smaller from the larger — the Euclidean algorithm. Every coprime pair runs down to $(1,1)$ that way, so every*

> *coprime pair is reachable from* $(1,1)$*. Induct on* $r+s$*.*

Layer 3 THE WHY

Kedlaya and Ng's solution begins: "It suffices to prove that for any relatively prime positive integers r, s, there exists an integer n with $a_n = r$ and $a_{n+1} = s$." Here is where that comes from.

Start from the phrasing. The set is $\{a_{n-1}/a_n\}$. A rational number is a pair of integers up to scaling, and in lowest terms it is exactly a coprime pair. So the statement "every positive rational appears" already *is* a statement about pairs of consecutive terms: it says the map $n \mapsto (a_{n-1}, a_n)$ hits every coprime pair up to a common factor. Before you can prove anything about which pairs are hit, you need to know how pairs evolve, and the recursion as written is about single indices. So the first thing to do is translate it into a rule on pairs — the only question being which pairs are "consecutive" in the recursion's sense. The indices $2n+1$ and $2n+2$ are consecutive, as are $2n+2$ and $2n+3$; writing out both gives the two transitions $(x,y) \mapsto (x, x+y)$ and $(x,y) \mapsto (x+y, y)$. Nothing clever has happened: this is reading the definition with the question's object in mind.

Now the transitions are the moves of the subtractive Euclidean algorithm in reverse. The Euclidean algorithm is the one procedure you know that acts on a pair of positive integers, preserves the gcd, and strictly shrinks the pair until it reaches (g,g), where g is the gcd. For a coprime pair it ends at $(1,1)$. Read backwards, every coprime pair is built from $(1,1)$ by the moves $(x,y) \to (x+y,y)$ and $(x,y) \to (x,x+y)$ — exactly our two transitions. Since $(a_0, a_1) = (1,1)$ is the seed, every coprime pair is reachable. The induction on $r+s$ is the formal version of "run Euclid". And $r+s$ is the right measure because each subtraction step lowers it by $\min(r,s) \geq 1$, whichever entry is larger, so one inequality covers both cases. (The maximum also drops, but the sum is the quantity the two cases share without comment; the minimum does not always drop — $(2,5) \to (2,3)$ keeps it.)

Why must the claim be strengthened? Because the weaker claim — every ratio appears — does not induct. If you know $r-s$ over s appears as a_{n-1}/a_n, you know nothing yet about whether $a_{n-1} = r-s$ exactly or $a_{n-1} = 2(r-s)$, and the transition rule works on actual values, not ratios. The strengthened claim pins down actual values, and it also forces you to discover the gcd condition, since $\gcd(a_n, a_{n+1})$ is preserved by the transitions and starts at 1. This is the general law of inductive proofs: the statement you induct on must carry enough information to be fed back into the recursion.

The official Note — the Stern–Brocot/Farey insertion process $\frac{a}{b}, \frac{c}{d} \mapsto \frac{a+c}{b+d}$ — is the same structure in another costume: mediants add numerators and denominators, which is again a sum-of-pair move.

Layer $+\alpha$ BACKTESTING

1. The first useful cue is the set $\{a_{n-1}/a_n\}$: ratios of neighbours. Write down (a_n, a_{n+1}) as the object, then compute what $a_{2n+1}, a_{2n+2}, a_{2n+3}$ do to it.

2. Recursion on pairs, recognised as the inverse subtractive Euclidean algorithm; strong induction on $r+s$.

3. Hesitation comes after computing terms, when you look for an explicit n for each p/q. Reinforce it away by asking not "where is p/q?" but "from which smaller pair can (p,q) be produced?" — the recursive question needs no formula.

4. The trap is proving facts about single terms ("every integer appears"), which cannot be fed back into a two-term recursion; and the secondary trap of chasing the binary-expansion formula, which is true (continued fraction via run lengths) but far more work.

5. Slogan: *when the question is about a ratio or relation of neighbouring terms, make the pair of neighbours the state and rewrite the recursion as a map on states.*

6. Decision rule: **to show a recursive process reaches every object of a type, run the process backwards: find a reduction that strictly decreases a size measure and ends at the seed.**

If the forward moves are $(x,y) \mapsto (x+y,y), (x,x+y)$, the reduction is Euclid and the reachable set is exactly the coprime pairs.

- ▷ **Cue:** a set of ratios a_{n-1}/a_n of consecutive terms of a recursive sequence $\longrightarrow$ **Move:** take the pair (a_n, a_{n+1}) as the state and write the recursion as transitions on pairs
- ▷ **Cue:** transitions $(x,y) \mapsto (x+y,y)$ and $(x,y) \mapsto (x,x+y)$ $\longrightarrow$ **Move:** recognise the reversed subtractive Euclidean algorithm; induct on $x+y$ down to $(1,1)$
- ▷ **Cue:** "every positive rational appears" $\longrightarrow$ **Move:** prove the stronger statement that every coprime pair appears as actual values; ratios alone do not induct
- ▷ **Cue:** an induction whose hypothesis cannot be fed back into the recursion $\longrightarrow$ **Move:** strengthen the statement until it records every quantity the recursion uses

2002 A6

2002 A6. Fix an integer $b \geq 2$. Let $f(1) = 1$, $f(2) = 2$, and for each $n \geq 3$, define $f(n) = nf(d)$, where d is the number of base-b digits of n. For which values of b does

$$\sum_{n=1}^{\infty} \frac{1}{f(n)}$$

converge?

Fields: analysis (series, integral comparison); self-similar recursions **Position:** A6 — the last problem of the morning, and hard in the sense that the usual convergence tests are too blunt to decide it. Once the self-similarity is written down, the proof is a page. **Answer:** The series converges for $b = 2$ only, and diverges for every $b \geq 3$. The dividing line is $\log b < 1$, i.e. $b < e$.

Layer 1 THE WHAT — surface analysis

The natural attempt. You unwind the definition. Write $\ell(n)$ for the number of base-b digits, so $\ell(n) \approx \log_b n$. Then $f(n) = n\,f(\ell(n)) = n\,\ell(n)\,f(\ell(\ell(n))) = \cdots$, and $f(n)$ is roughly $n \cdot \log_b n \cdot \log_b \log_b n \cdots$, the product running until the iterates fall to 1 or 2. So the series looks like

$$\sum \frac{1}{n \log n \log\log n \log\log\log n \cdots},$$

and you know the finite truncations of this: $\sum 1/(n \log n)$ diverges, $\sum 1/(n \log n \log\log n)$ diverges, and so on, each by Cauchy condensation or the integral test. The infinitely-iterated version sits right on the border. You try the integral test on each layer and every time the answer is "diverges, up to a constant factor" — and the constant factors are exactly what you have been throwing away with $\approx$. You try a computer-style estimate: for $b = 2$ the first million terms sum to about 1.91, for $b = 3$ about 2.87, and both are still creeping upwards. Numerics cannot tell you which one stops. The wall: every tool you are using is insensitive to constants, and in this problem the answer *is* a constant.

The model chain. Let $\ell(n)$ be the number of base-b digits of n, and for $d \geq 1$ let $B_d = \{b^{d-1}, \ldots, b^d - 1\}$ be the d-digit block. Write

$$H_d = \sum_{n \in B_d} \frac{1}{n}.$$

Block estimates. Since $1/x < 1/n$ for $x \in (n, n+1)$,

$$H_d > \int_{b^{d-1}}^{b^d} \frac{dx}{x} = \log b. \tag{1}$$

For $b = 2$, with $N = 2^{d-1}$, compare $\sum_{n=N+1}^{2N} 1/n < \int_N^{2N} dx/x = \log 2$, so

$$H_d = \frac{1}{N} + \sum_{n=N+1}^{2N} \frac{1}{n} - \frac{1}{2N} < \log 2 + \frac{1}{2^d}. \tag{2}$$

Case $b \geq 3$: divergence. Here $1, 2 \in B_1$, and $f(1) = 1 = 1 \cdot f(1)$, $f(2) = 2 = 2 \cdot f(1)$, so $f(n) = n\, f(\ell(n))$ holds for *every* $n \geq 1$. Suppose the series converges, with sum $S > 0$. Grouping the positive terms by blocks (legitimate for a convergent series of positive terms),

$$S = \sum_{d \geq 1} \sum_{n \in B_d} \frac{1}{n\, f(d)} = \sum_{d \geq 1} \frac{H_d}{f(d)} > \log b \sum_{d \geq 1} \frac{1}{f(d)} = (\log b)\, S,$$

using (1). Since $b \geq 3 > e$, $\log b > 1$, so $S > S$, a contradiction. The series diverges.

Case $b = 2$: convergence. Now $B_1 = \{1\}$, $B_2 = \{2, 3\}$ with $f(2) = 2$ (not $2f(2)$), $f(3) = 3f(2) = 6$; for $d \geq 3$ and $n \in B_d$, $f(n) = n\, f(d)$. Let $P_i = \sum_{n=1}^{2^i - 1} 1/f(n)$ for $i \geq 3$. Then

$$P_i = 1 + \tfrac{1}{2} + \tfrac{1}{6} + \sum_{d=3}^{i} \frac{H_d}{f(d)} = \tfrac{5}{3} + \sum_{d=3}^{i} \frac{H_d}{f(d)}.$$

By (2), $H_d < c := \log 2 + \frac{1}{8} \approx 0.818 < 1$ for $d \geq 3$. Also, since $i \leq 2^i - 1$, the terms $1/f(4), \ldots, 1/f(i)$ all occur in P_i beyond the first three terms, so $\sum_{d=3}^{i} 1/f(d) \leq \frac{1}{6} + (P_i - \frac{5}{3})$. Hence

$$P_i - \tfrac{5}{3} < c\left(\tfrac{1}{6} + P_i - \tfrac{5}{3}\right) \quad \Longrightarrow \quad P_i < \tfrac{5}{3} + \frac{c}{6(1-c)} = \tfrac{3}{2} + \frac{1}{6(1-c)} < 2.42.$$

Every partial sum is at most some P_i, so the partial sums are increasing and bounded, and the series converges. □

Caveat. Verified independently. The bound $P_i < \frac{3}{2} + \frac{1}{6(1-c)}$ equals the official constant $L \approx 2.4165$; I derived it by a one-line self-referential inequality in place of the official induction on i, and checked both arguments. Numerically, partial sums to 10^6 are 1.91 ($b = 2$) and 2.87 ($b = 3$); these illustrate why numerics cannot settle the question and are not part of the proof.

Layer 2 THE HOW — the bifurcation

Bifurcating point. Both chains unwind $f(n) = nf(\ell(n))$. They split at (†): what to do with the unwound form. The natural attempt expands f into an iterated-log product and applies comparison tests, losing constants. The model does not unwind at all; it groups n by digit count and sees the original series reappear, multiplied by the exact constant $H_d \approx \log b$.

> *Inside the d-digit block, $f(n)/n$ is the constant $f(d)$. So the block contributes $\frac{1}{f(d)}$ times a harmonic chunk, and a harmonic chunk from b^{d-1} to b^d is about* $\log b$. *The whole series is the same series, re-indexed by d, times* $\log b$.

> *$S \approx (\log b)\, S$. A positive finite S can only survive if* $\log b < 1$. *So the answer turns on $b < e$: $b = 2$ converges, $b \geq 3$ diverges. Now make "$\approx$" into strict inequalities in the right direction for each case.*

Layer 3 THE WHY

Why group by digit count? Because the definition tells you to. $f(n)$ depends on n in two ways — the explicit factor n and the digit count d — and the second is constant on the huge blocks B_d. Whenever a function is piecewise "simple on known blocks", you sum block by block first. On B_d the summand is $\frac{1}{f(d)} \cdot \frac{1}{n}$, and $\sum_{n \in B_d} \frac{1}{n}$ is a harmonic sum over a range whose endpoints have ratio b.

Why is that sum $\log b$, independent of d? Scale invariance of dx/x: $\int_N^{bN} dx/x = \log b$ for every N. The harmonic series gives each multiplicative range the same weight, and base-b blocks are multiplicative ranges. That is the one piece of analysis the problem needs, and it is A-level: the integral of $1/x$ plus the rectangle comparison.

Put the two together and the series reproduces itself: the block sums $H_d/f(d)$ are the original terms $1/f(d)$ scaled by $H_d \approx \log b$. The self-similar structure of f becomes self-similarity of the series. A quantity satisfying $S = \lambda S + (\text{small})$ with $\lambda > 1$ cannot be finite and positive; with $\lambda < 1$ it is bounded by $(\text{small})/(1-\lambda)$. That is the complete logic of both halves — the official solution states it as a contradiction for $b \geq 3$ and an induction for $b = 2$, but the idea is the fixed-point equation. Kedlaya and Ng introduce $c = \frac{1}{8} + \log 2$ and $L = 1 + \frac{1}{2} + \frac{1}{6(1-c)}$ without comment. Here is where they come from: c is the worst block multiplier over $d \geq 3$ (the $\frac{1}{8} = 2^{-3}$ is the error in (2) at $d = 3$), and L is the solution of $P = \frac{5}{3} + c(\frac{1}{6} + P - \frac{5}{3})$, the fixed-point bound.

Why is the threshold e, and why does it make the problem hard? The natural attempt was right that $f(n)$ behaves like $n \log n \log\log n \cdots$; what it missed is that the *base* of those logarithms matters. Changing base multiplies each logarithm by a constant, and with infinitely many nested logarithms those constants compound. Comparison tests discard constants, so they are blind exactly where the answer lives. The problem is built so that e falls between the only two candidate small bases: $\log 2 \approx 0.69 < 1 < 1.10 \approx \log 3$. The asymmetry in the definition ($f(2) = 2$ imposed by hand) exists to make $b = 2$ well defined — for $b = 2$ the rule $f(2) = 2f(2)$ would be circular — and it costs the $b = 2$ proof its first few terms, which is why the official identity starts $1 + \frac{1}{2} + \frac{1}{6}$.

Also notice the direction of the inequalities. For divergence you need H_d bounded *below* (rectangles above the curve); for convergence, H_d bounded *above* and strictly less than 1 (rectangles below the curve, plus a correction). The bound $\log 2 + \frac{1}{2^d}$ exceeds $\log 2$, and it matters that the excess is small enough: at $d = 3$, $H_3 = \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} \approx 0.760$, comfortably below $c \approx 0.818$.

Layer $+\alpha$ BACKTESTING

1. The first useful cue is "$f(n) = nf(d)$ where d is the number of digits": $f(n)/n$ is constant on digit blocks. Write down $\sum_n \frac{1}{f(n)} = \sum_d \frac{1}{f(d)} \sum_{n \in B_d} \frac{1}{n}$.

2. Block summation plus integral comparison of the harmonic block with $\log b$; then a fixed-point (self-referential) inequality for the sum.

3. Hesitation comes when the unwound form $n \log n \log\log n \cdots$ looks like a standard borderline series. Reinforce it away: borderline series are decided by constants, so keep the exact constant and do not unwind.

4. The trap is Cauchy condensation / comparison "up to constants", which cannot see the difference between $b = 2$ and $b = 3$; and numerical partial sums, which at 10^6 terms are still far from the $b = 2$ limit (about 2.08–2.19).

5. Slogan: *when a sum is defined in terms of itself at a smaller scale, group by scale and look for $S = \lambda S + \text{const}$; the answer is decided by whether $\lambda < 1$.*

6. Decision rule: **if a summand is constant, up to an explicit factor, on blocks of a known shape, sum over each block exactly first; if the block sums reproduce the original terms, the question reduces to comparing a single multiplier with 1, and all later estimates must preserve that constant.**

▷ **Cue:** a definition like $f(n) = nf(\text{number of digits of } n)$ ⟶ **Move:** sum over each digit block; $f(n)/n$ is constant there

- ▷ **Cue:** a harmonic sum over $[N, bN)$ $\longrightarrow$ **Move:** it is about $\log b$ for every N, by $\int_N^{bN} dx/x = \log b$ with rectangles above or below as needed
- ▷ **Cue:** a series whose grouped form is $S = \sum \lambda_d \, t_d$ with the t_d themselves terms of S $\longrightarrow$ **Move:** write $S \gtrless \lambda S + C$ and decide by λ versus 1
- ▷ **Cue:** a borderline series like $\sum 1/(n \log n \log\log n \cdots)$ $\longrightarrow$ **Move:** comparison tests lose constants; find an exact self-similarity instead
- ▷ **Cue:** a bounded increasing sequence of partial sums defined self-referentially $\longrightarrow$ **Move:** bound P by $C + c\,P$ with $c < 1$, giving $P \leq C/(1-c)$ without induction

Putnam 2002 — Paper B

2002 B1

2002 B1. Shanille O'Keal shoots free throws on a basketball court. She hits the first and misses the second, and thereafter the probability that she hits the next shot is equal to the proportion of shots she has hit so far. What is the probability she hits exactly 50 of her first 100 shots?

Fields: Probability (Pólya urns, exchangeability, Beta–Binomial) **Position:** B1 – opening problem of the second session **Answer:** The probability is $1/99$, because after any $n \geq 2$ shots the number of hits is uniformly distributed over $1, \ldots, n-1$; this is a Pólya urn started from one hit and one miss, and it is exactly the predictive process of a hidden success probability with a Uniform$(0,1)$ prior.

Layer 1 THE WHAT — surface analysis

The natural attempt. You read "probability she hits exactly 50 of her first 100 shots" and reach for a direct computation: condition on the full sequence of 100 outcomes, and try to sum the probabilities of every sequence with exactly 50 hits and 50 misses (with the first shot forced to be a hit and the second forced to be a miss). Each such sequence has a probability that is a product of 98 fractions, one per shot after the second, and each fraction depends on how many hits and misses have occurred *by that point in that specific sequence* – not just on the final tally. This looks tractable at first: you try $n = 3$ and $n = 4$ by hand, list the few sequences, multiply out the fractions, and notice something suspicious – every attainable hit-count seems to come out with the same probability. That observation is real progress, but if you do not stop and ask why it is happening, the natural next step is to try to verify it for $n = 100$ the same way: write down the sum over all $\binom{98}{48}$ order-98 sequences with 48 further hits among 48 further hits and 50 further misses (after the forced H, M start), and hope the algebra collapses. It does collapse, eventually, but only because of a cancellation that is invisible unless you have already guessed the general pattern; attacking $n = 100$ head-on, sequence by sequence, is not a computation a human being finishes in an exam.

The model chain. Claim: for every $n \geq 2$ and every i with $1 \leq i \leq n-1$, the probability of exactly i hits among the first n shots is $\frac{1}{n-1}$ – uniform over all attainable counts. Prove this by induction on n. Base case $n = 2$: there is exactly one attainable count, $i = 1$ (the forced H, M), with probability $1 = \frac{1}{2-1}$. Inductive step: suppose the claim holds for n. After n shots with i hits, the probability of hitting shot $n+1$ is i/n; after n shots with $i-1$ hits, the probability of missing shot $n+1$ is $1-(i-1)/n = (n-i+1)/n$. So

$$P(i \text{ hits after } n+1) = \frac{i-1}{n} \cdot \frac{1}{n-1} + \left(1 - \frac{i}{n}\right) \cdot \frac{1}{n-1} = \frac{(i-1)+(n-i)}{n(n-1)} = \frac{1}{n},$$

which is exactly $\frac{1}{(n+1)-1}$, independent of i. This closes the induction. Setting $n = 100$, $i = 50$ gives probability $\frac{1}{99}$.

Layer 2 THE HOW — the bifurcation

Bifurcating point. Both paths start from the same recursive structure – the probability of the next shot depends only on the current hit count and shot count. They split at the point of deciding *what to compute*: the natural attempt tries to compute the single number $P(50 \text{ hits after } 100 \text{ shots})$ directly, by summing over sequences. The model chain instead conjectures and proves a stronger, general statement – that the whole distribution over hit-counts is uniform for every n – and reads off $n = 100$ as one instance of it.

> *Small cases at $n = 3$ and $n = 4$ are coming out uniform across every attainable hit-count, and that is too clean to be a coincidence of small numbers – I should stop computing $n = 100$ directly and instead try to prove the uniform-distribution statement for general n by induction, because if it is true in general, $n = 100$ falls out for free.*

> *When I substitute the inductive hypothesis $\frac{1}{n-1}$ into the recursion for $P(i \text{ hits after } n+1)$, the two fractions $\frac{i-1}{n}$ and $\frac{n-i}{n}$ add to exactly $\frac{n-1}{n}$ regardless of i – the i-dependence cancels on its own, and that cancellation is precisely what makes the uniform hypothesis self-propagating.*

Layer 3 THE WHY

Caveat. The induction above is a complete, self-contained proof that I verified independently (matching it against $n = 2, 3, 4$ by direct enumeration) before comparing it to Kedlaya–Ng's published solution, which runs the identical induction. No step is unverified. The Bayesian/Pólya-urn framing below is offered as the structural explanation for *why* the induction works, and I checked its central claim – that Beta$(1,1)$-Binomial is uniform – by direct computation of the Beta function, so it is verified as well, not merely asserted.

Why is the distribution uniform at all – why does self-reinforcement (next-shot probability equals the running success rate) produce something this clean, rather than concentrating around some typical value the way a fixed-p binomial would? The process described is a Pólya urn: after the forced H, M start, think of an urn with one "hit" ball and one "miss" ball; each shot draws a ball according to the current composition and returns it *with an extra ball of the same colour* – this is exactly "probability of a hit equals the current proportion of hits." Pólya urns started from one ball of each colour have a classical closed form: the number of additional hits X among m further draws satisfies $P(X = k) = \binom{m}{k} \frac{k!(m-k)!}{(m+1)!} = \frac{1}{m+1}$ for every $k = 0, \ldots, m$ – uniform, with no dependence on k at all. This is not a coincidence of the $(1,1)$ starting point; it is the statement that a Beta$(1,1)$ prior (the uniform distribution on $[0,1]$) is the unique symmetric conjugate prior for a Bernoulli success probability, and the predictive distribution of a count drawn from a Bernoulli process with an unknown $p \sim \text{Uniform}(0,1)$, integrated over p, is the discrete uniform distribution on the count. So the free-throw process is, structurally, indistinguishable from: nature secretly chooses a fixed shooting percentage p uniformly at random on $[0,1]$, and every shot is then an independent Bernoulli(p) trial. Once you see that equivalence, the uniformity of the hit-count is not a fact to memorise about this particular problem; it is the single fact that a uniform prior stays exchangeable under Bayesian updating with Bernoulli data, applied here in its rawest form.

Layer $+\alpha$ BACKTESTING

1. The first useful cue is the phrase "the probability that she hits the next shot is equal to the proportion of shots she has hit so far." On seeing it, write down "this is self-reinforcement with linear feedback – flag it as a Pólya urn" before doing any arithmetic.

2. That cue suggests two viable methods: prove a general uniform-distribution lemma by induction on the shot count, or recall/derive the Beta$(1,1)$-Binomial identity directly. Either one avoids ever summing over the $\binom{98}{48}$ sequences for $n = 100$.
3. Hesitation occurs when you have verified uniformity for $n = 3, 4$ by hand and are tempted to keep verifying small cases instead of committing to a general induction; the fix is to write out the inductive step's recursion symbolically the moment two or three small cases agree, rather than trusting the pattern by further enumeration.
4. The trap is assuming that a probability which changes at every single step must produce a complicated, non-uniform final distribution – self-reinforcement toward the empirical rate feels like it should concentrate mass, when in fact linear reinforcement from a symmetric $(1,1)$ start produces the flattest possible distribution.
5. To recognise this faster: whenever "probability of the next success equals the current success rate" appears, and the process starts from one success and one failure, name it a Pólya urn on sight and expect uniformity over the final count – do not attempt a direct combinatorial sum first.
6. The reusable rule: a sequential Bernoulli-type process whose success probability at each step equals the running proportion of past successes is equivalent to drawing i.i.d. Bernoulli trials with a success probability chosen once from Uniform$(0,1)$; consequently the number of successes after any number of trials is uniformly distributed over every attainable value.

2002 B2

2002 B2. Consider a polyhedron with at least five faces such that exactly three edges emerge from each of its vertices. Two players play the following game: each player, in turn, signs his or her name on a previously unsigned face. The winner is the player who first succeeds in signing three faces that share a common vertex. Show that the player who signs first will always win by playing as well as possible.

Fields: Combinatorics, elementary graph theory (trivalent polyhedra, Euler's formula) **Position:** B2 – second problem of the second session **Answer:** The first player wins in at most three moves: sign any face with at least four edges (such a face is guaranteed to exist), then sign the middle of three consecutive still-open neighbouring faces, creating two independent one-move-from-winning triples that the second player cannot both block.

Layer 1 THE WHAT — surface analysis

The natural attempt. You read "trivalent polyhedron" and "sign three faces sharing a vertex" and reach for a concrete example – a cube, say, or a dodecahedron – and try to find a first move and a follow-up that wins on that specific solid, hoping the argument generalises. On the cube this is not hard to find by trial: sign a face, watch where the opponent replies, sign an adjacent face, and a winning fork appears. But the moment you try to write the argument for an *arbitrary* polyhedron satisfying only "at least five faces, exactly three edges per vertex," the case-by-case reasoning that worked on the cube stops transferring. What if every face is a triangle? What if the polyhedron is highly irregular, with faces of wildly different sizes? You do not know, in general, whether there is always a face with enough neighbours to build the fork you found on the cube – and without that guarantee, the strategy that worked on one example is not a proof. You can spend a long time trying more examples (tetrahedron, triangular prism, dodecahedron) and convincing yourself first player always wins, without ever isolating the one structural fact that makes the fork possible on every valid input.

The model chain. Step one, a pure existence lemma, no game yet: some face of the polyhedron has at least four edges. Suppose not – every face is a triangle. Fix any face F_1 with vertices v_1, v_2, v_3. Since exactly three edges meet at v_1, and two of them (v_1v_2 and v_1v_3) already belong

to F_1, the third edge out of v_1 has some other endpoint v_4. Because every face is a triangle, the face on the other side of edge v_1v_4 must be a triangle containing v_1, v_4, and its third vertex is forced (by the same trivalence argument applied at v_2 and v_3) to also be v_4: the faces adjacent to F_1 are exactly $v_1v_2v_4$, $v_1v_3v_4$, and $v_2v_3v_4$. So the whole structure closes up on the four vertices v_1, v_2, v_3, v_4 – it is a tetrahedron, with only four faces, contradicting the hypothesis of at least five. So a face A with $k \geq 4$ edges exists.

Step two, the game. Player 1 signs A. List the $k \geq 4$ faces adjacent to A in their cyclic order around A: $F_1, \ldots, F_k$. Because every vertex of A has exactly three edges, each vertex of A is shared by A and exactly two cyclically consecutive neighbours F_i, F_{i+1} – so $\{A, F_i, F_{i+1}\}$ is precisely the set of faces meeting at that vertex. Player 2 signs one face somewhere on the board; at most one of $F_1, \ldots, F_k$ is thereby removed from play. Removing one entry from a cyclic list of length $k \geq 4$ leaves a run of at least $k - 1 \geq 3$ consecutive survivors; call three consecutive survivors B, C, D (in that cyclic order). Player 1 signs C. Now $\{A, B, C\}$ shares a vertex, and $\{A, C, D\}$ shares a (different) vertex – both triples are one move from complete, and both already contain Player 1's two signed faces A, C. Player 2 has one move left before Player 1's third turn, and can occupy at most one of B, D. Player 1 signs the other on the third move and completes a triple. First player wins by the third move.

Layer 2 THE HOW — the bifurcation

Bifurcating point. Both approaches accept the win condition (three faces at a common vertex) and both eventually look for a "fork" – two threats at once. They split earlier than that, at whether to first establish *which face to start on*. The natural attempt picks a face (any face, or a face that happens to work on a specific familiar solid) and tries to build the fork from there. The model chain refuses to choose a starting face until it has proved, independently of the game, that a face with at least four edges must exist – and only then builds the fork using that guaranteed face as the anchor.

> *I cannot pick an arbitrary face and hope a fork appears on it; the fork needs three consecutive untouched neighbours to survive the opponent's one reply, and that is only guaranteed if my starting face has at least four neighbours to begin with – so before touching the game at all, I need to ask whether such a face is forced to exist under these exact hypotheses, and prove it as its own lemma.*

> *If I sign the middle face C of three consecutive unoccupied neighbours B, C, D of A, then $\{A, B, C\}$ and $\{A, C, D\}$ are both one move from complete at two genuinely different vertices, sharing only my own moves A and C – my opponent has exactly one move against two independent threats and cannot block both.*

Layer 3 THE WHY

Caveat. Both stages of this proof – the existence lemma and the fork – I verified independently before comparing to Kedlaya–Ng: the existence lemma by direct vertex-counting on the tetrahedron collapse, and the fork by checking it explicitly on the cube (a face with $k = 4$ neighbours) and on a hexagonal face of a truncated-octahedron-type solid ($k = 6$), confirming the pigeonhole step (one deletion from a cyclic list of length ≥ 4 leaves ≥ 3 consecutive survivors) both times. No step is unverified.

Why does trivalence force an all-triangle polyhedron to collapse all the way down to a tetrahedron, rather than merely constraining it? Because trivalence is a rigidity condition with zero slack: once a vertex v_1 lies on two known triangular faces, its two edges are already spoken for, and it has exactly one more edge – not "at most one," exactly one, because the hypothesis fixes the degree precisely. That single remaining edge is forced, and the argument shows it is

forced to the *same* fourth vertex from every corner of F_1 simultaneously. There is no freedom anywhere in the structure to add a fifth face; the moment you fix one triangular face under trivalence, the rest of an all-triangle solid is already determined. This is why the lemma needs both hypotheses together – "at least five faces" only becomes a contradiction because trivalence removes every degree of freedom that could otherwise have produced a bigger all-triangle solid.

Why does the fork specifically require degree ≥ 4, not ≥ 3? A face with exactly three neighbours offers the opponent's single reply move enough leverage to break the run: remove one of three, and at most two consecutive survivors remain – one shared vertex, not two, so no fork. Only at $k \geq 4$ does removing one entry from the cycle leave a run of ≥ 3 consecutive faces, which is the minimum needed to have a genuine middle. And why is it specifically the *middle* face C of the surviving run, not B or D, that must be signed? Because "three faces share a vertex" around A is a property of *cyclically consecutive* pairs, by the trivalence-forces-adjacency argument used to build the lemma in the first place – only the middle face of a run of three belongs to two overlapping consecutive pairs at once, and it is exactly that overlap, not anything about B or D individually, that produces two threats from one move.

Layer $+\alpha$ BACKTESTING

1. The first useful cue is "exactly three edges emerge from each vertex" sitting next to "sign three faces that share a common vertex." On seeing it, write down: trivalence means any three faces sharing a vertex are exactly a face together with its two cyclically adjacent neighbours – go looking for a face with enough neighbours to exploit that.

2. That cue suggests a two-stage method: an existence lemma proved independently of the game (does a face of degree ≥ 4 have to exist?), followed by a fork built purely from cyclic adjacency once that lemma is in hand.

3. Hesitation occurs when you try to handle "what if every face is a triangle" as a case inside the game strategy itself, worrying about how to fork on a triangle-only solid. The fix is to isolate it as a separate structural question before touching the game at all – and discover it cannot happen given the stated hypotheses.

4. The trap is picking a face that "seems central" on a familiar example (a cube face, say) without checking the general count: what matters is not that the face looks prominent, but that removing one opposing move from its cyclic neighbour list still leaves three *consecutive* survivors, which needs degree ≥ 4 exactly.

5. To recognise this faster: whenever a positional game's win condition is "k objects sharing a structural feature" (here, a vertex), ask first whether the starting configuration guarantees an anchor object with enough incident neighbours that one opposing move cannot prevent a two-pronged fork – and prove that existence claim on its own before designing any move sequence.

6. The reusable rule: to win a "complete k things meeting at a point" game in the fewest moves, first prove structurally that some point has enough incident objects that a single opposing move cannot destroy every possible fork, then always sign the object lying in two overlapping winning sets at once – the middle of a surviving run, never an end.

2002 B3

2002 B3. Show that, for all integers $n > 1$,

$$\frac{1}{2ne} < \frac{1}{e} - \left(1 - \frac{1}{n}\right)^n < \frac{1}{ne}.$$

Fields: Analysis (Taylor/Maclaurin series, termwise comparison of positive series) **Position:** B3 – third problem, entry to the harder half of the second session **Answer:** The double inequality holds for every

integer $n > 1$: taking logarithms converts it into a comparison of three convergent power series in $1/n$, and the inequality is proved by comparing those series term by term, with equality occurring only at the very first index of the lower bound.

Layer 1 THE WHAT — surface analysis

The natural attempt. You see $(1-1/n)^n$ sandwiched between expressions built from $1/e$, and your first instinct is calculus in n: treat n as continuous, define $h(n) = 1/e - (1-1/n)^n$, and try to bound h by differentiating it, or by comparing it to a Riemann sum, or by expanding $(1-1/n)^n$ directly with the binomial theorem and grouping terms. Each of these runs into the same wall. Differentiating $(1-1/n)^n$ with respect to n requires logarithmic differentiation anyway (the variable sits in both the base and the exponent), and the resulting derivative is itself a sum of a logarithm term and a rational term with a sign that is not easy to pin down – you have replaced one hard-to-bound quantity with another. The binomial expansion $(1-1/n)^n = \sum_{k=0}^{n} \binom{n}{k}(-1/n)^k$ has $n+1$ alternating terms whose partial sums oscillate around the limit, and comparing that finite alternating sum term by term to $1/(2ne)$ and $1/(ne)$ – two quantities with no clean, direct link to any single binomial coefficient – does not close cleanly. You can spend real effort trying to bound the alternating tail of the binomial expansion and get inequalities in the right shape but with the wrong constants, or constants that only work for n above some threshold rather than for all $n > 1$ as the problem demands.

The model chain. Divide the whole inequality by $1/e$ and rearrange so that every term is manifestly positive (which will justify taking logarithms in a moment):

$$1 - \frac{1}{n} \;<\; e\left(1-\frac{1}{n}\right)^n \;<\; 1 - \frac{1}{2n}.$$

All three quantities lie in $(0,1)$ for $n > 1$ (the middle one because $(1-1/n)^n$ increases to $1/e$ from below as $n \to \infty$, so $e(1-1/n)^n < 1$; and $1 - 1/(2n) > 1 - 1/n > 0$), so taking (natural) logarithms preserves the direction of both inequalities:

$$\log\left(1-\frac{1}{n}\right) \;<\; 1 + n\log\left(1-\frac{1}{n}\right) \;<\; \log\left(1-\frac{1}{2n}\right).$$

Rearrange each side to isolate $-1 - n\log(1-1/n)$ in the middle, and negate throughout (flipping both inequalities):

$$-\log\left(1-\frac{1}{2n}\right) \;<\; -1 - n\log\left(1-\frac{1}{n}\right) \;<\; -\log\left(1-\frac{1}{n}\right).$$

Now every one of these three quantities has a standard convergent power-series expansion, using $-\log(1-x) = \sum_{i=1}^{\infty} x^i/i$ for $|x| < 1$. For the outer two, with $x = 1/(2n)$ and $x = 1/n$ respectively:

$$-\log\left(1-\frac{1}{2n}\right) = \sum_{i=1}^{\infty} \frac{1}{i\,2^i\,n^i}, \qquad -\log\left(1-\frac{1}{n}\right) = \sum_{i=1}^{\infty} \frac{1}{i\,n^i}.$$

For the middle quantity: $n\log(1-1/n) = -n\sum_{i=1}^{\infty} \frac{1}{i\,n^i} = -\sum_{i=1}^{\infty} \frac{1}{i\,n^{i-1}} = -\left(1 + \sum_{i=1}^{\infty} \frac{1}{(i+1)n^i}\right)$, using the $i = 0$ shift; so

$$-1 - n\log\left(1-\frac{1}{n}\right) = \sum_{i=1}^{\infty} \frac{1}{(i+1)\,n^i}.$$

The whole inequality has become a comparison of three series with a common index i:

$$\sum_{i=1}^{\infty} \frac{1}{i\,2^i\,n^i} \;<\; \sum_{i=1}^{\infty} \frac{1}{(i+1)\,n^i} \;<\; \sum_{i=1}^{\infty} \frac{1}{i\,n^i}.$$

Compare coefficients termwise, for each fixed $i \geq 1$: the right-hand comparison $\frac{1}{i+1} < \frac{1}{i}$ holds for every i, strictly. The left-hand comparison $\frac{1}{i2^i} \leq \frac{1}{i+1}$, i.e. $i+1 \leq i \cdot 2^i$, holds for every $i \geq 1$ (with equality exactly at $i = 1$: $2 = 1 \cdot 2$; strict for $i \geq 2$, since $i \cdot 2^i$ then grows far faster than $i+1$). Since all terms are positive and the coefficient inequalities hold at every index – strictly for $i \geq 2$ on the left and strictly for every i on the right – summing over i preserves strict inequality on both sides (the left sum cannot be forced to equality by one tied term when every other term is strictly smaller). This proves both halves of the original inequality for every integer $n > 1$.

Layer 2 THE HOW — the bifurcation

Bifurcating point. Both approaches begin at the same object, $(1-1/n)^n$ compared against multiples of $1/e$. They split at the very first move: the natural attempt keeps $(1-1/n)^n$ in exponential form and tries to differentiate or expand it directly, with n still appearing in both the base and the exponent. The model chain takes logarithms of the rearranged inequality *before* doing anything else, which turns the exponent n into a coefficient multiplying a logarithm – a linear, not exponential, role.

> *The quantity $(1-1/n)^n$ has n tangled into it twice, once in the base and once in the exponent, so differentiating or expanding it directly keeps both roles coupled together; if I take logarithms first, $n\log(1-1/n)$ has n appearing only as a multiplying coefficient in front of a logarithm with a known, universal power-series expansion, and that is the operation that actually separates the two roles.*

> *Once every term in the inequality is rewritten as a power series in $1/n$ with the same index i, a three-way inequality between three specific real numbers becomes a three-way inequality between three sums indexed the same way – and if the coefficient of $1/n^i$ satisfies the required inequality for every single i, I never need to evaluate any of the three sums in closed form; termwise comparison closes the whole problem at once.*

Layer 3 THE WHY

Caveat. I re-derived every series expansion above from $-\log(1-x) = \sum x^i/i$ independently (including the index-shift in the middle series, which I checked against $n\log(1-1/n)$ numerically at $n = 2, 5, 10$) before comparing to Kedlaya–Ng's published solution, which follows the identical route. The one place I add detail beyond the official write-up is flagging that the left-hand termwise comparison is an equality at $i = 1$, not a strict inequality there – the official solution states the comparison is "evident... term by term" without flagging this, and I verified explicitly that strictness at $i \geq 2$ is still enough to force the overall sum inequality to be strict. No step is unverified.

Why does taking logarithms expose a power series that was invisible in the original exponential form? Because log turns exponentiation into multiplication: $\log(a^n) = n\log(a)$. That alone is not yet the whole answer – $n\log(1-1/n)$ still has n appearing twice, once as a multiplier and once inside the argument of log. What makes it useful is that $\log(1-x)$ has a single universal convergent series valid for every value of $x = 1/n \in (0,1)$, and multiplying that series through by n turns each term $x^i/i = 1/(i\,n^i)$ into $1/(i\,n^{i-1})$ – shifting the whole family down one power of n and, crucially, producing a series whose *coefficients* ($1/(i+1)$ after reindexing) no longer depend on n at all. The original problem is a statement about one specific number, n; after taking logs, it becomes a statement about the coefficients of a family of series that is the same for every n – which is exactly why a single termwise argument, done once, settles the inequality simultaneously for every $n > 1$.

Why do the bounds land specifically at $1/(2ne)$ and $1/(ne)$, and not some other constants? These are read directly off the leading (index $i = 1$) term of the difference series: $-1 - n\log(1 - 1/n) = \sum_{i\geq 1} \frac{1}{(i+1)n^i}$ has leading term $\frac{1}{2n}$, which is exactly why the lower comparison ties at $i = 1$ (coefficient $\frac{1}{1\cdot 2^1} = \frac{1}{2}$ matching $\frac{1}{1+1} = \frac{1}{2}$ precisely) while every higher-order term of the true series is strictly larger than the corresponding term of $-\log(1 - 1/(2n))$. The constant "2" in "$2n$" is not an arbitrary bracketing choice; it is the smallest constant c for which $-\log(1 - \frac{1}{cn})$ stays termwise below $\sum \frac{1}{(i+1)n^i}$ for every i, because $c = 2$ is exactly where the two series first touch (at $i = 1$) before the higher terms pull them apart. This is the same underlying fact as the well-known asymptotic $1/e - (1 - 1/n)^n \sim \frac{1}{2ne}$ as $n \to \infty$ – the problem is asking you to prove that this leading asymptotic behaviour is in fact a genuine two-sided bound for every $n > 1$, not merely a limiting statement.

Layer $+\alpha$ BACKTESTING

1. The first useful cue is an expression like $(1 - 1/n)^n$ appearing next to $1/e$ in an inequality. On seeing it, write down: the variable n is doing two jobs at once (base and exponent), so before manipulating this expression at all, take logarithms to separate those two jobs.
2. That cue suggests the method directly: rewrite the inequality with logarithms, then expand every logarithmic term using the standard series $-\log(1 - x) = \sum_{i\geq 1} x^i/i$, and compare the resulting series termwise.
3. Hesitation occurs right after taking logs, when the recursive-looking term $n\log(1 - 1/n)$ tempts you to truncate its series after one or two terms and bound the remainder separately – worrying about error terms. The fix is to keep the full infinite series on all three sides and compare coefficients index by index; no remainder ever needs to be bounded because the comparison is exact at every single index.
4. The trap is trying to differentiate or binomially expand $(1 - 1/n)^n$ directly, in its original exponential form, hoping calculus alone will produce a sign. Differentiating an expression with the same variable in the base and the exponent does not decouple those two roles – it just relocates the difficulty into a derivative that is no easier to sign than the original expression.
5. To recognise this faster: whenever a bound involves e or a limit-defining expression such as $(1 \pm x/n)^n$ set against rational functions of n, expect that taking logarithms and invoking the $\log(1 \pm x)$ series is the route in, because it is the one operation that turns a two-parameter (base, exponent) tangle into a single power series in one parameter.
6. The reusable rule: to compare a limit-defining exponential expression like $(1 - 1/n)^n$ against rational bounds in n, take logarithms to expose the underlying convergent power series in $1/n$, then prove the required inequality coefficient by coefficient rather than manipulating the exponential form directly – and read the sharp constants in the bound off the leading term of the resulting difference series.

2002 B4

2002 B4. An integer n, unknown to you, has been randomly chosen in the interval $[1, 2002]$ with uniform probability. Your objective is to select n in an **odd** number of guesses. After each incorrect guess, you are informed whether n is higher or lower, and you **must** guess an integer on your next turn among the numbers that are still feasibly correct. Show that you have a strategy so that the chance of winning is greater than $2/3$.

Fields: combinatorics; games and strategies; binary search trees; probability (counting) **Position:** B4 — mid-paper; nothing beyond counting is needed, but the strategy has to be invented, and the natural search strategies all sit at $1/2$ **Answer:** Guess $1, 3, 4, 6, 7, 9, \ldots$ (every integer not $\equiv 2 \pmod 3$) until

told "lower"; this wins for all $n \not\equiv 0 \pmod 3$, i.e. 1335 of the 2002 values, and $1335/2002 > 2/3$. (In fact 1335 is the best possible.)

Layer 1 THE WHAT — surface analysis

The natural attempt. The strong student's first reflex is binary search: it is the canonical "higher or lower" strategy, and it finds n in about 11 guesses. But binary search controls the *number* of guesses, not its *parity*. In the search tree of binary search, depth d holds 2^{d-1} nodes, so the deepest levels dominate, and the parity of the deepest level decides everything: roughly half the numbers, sometimes a bit more, sometimes a bit less. Not $2/3$.

The second attempt is linear: guess $1, 2, 3, \ldots$ in order. Then n is guessed on turn n, and you win exactly when n is odd: probability $1001/2002 = 1/2$. The student then tries to "skip" so that more numbers land on odd turns — guess $1, 3, 5, \ldots$? — and discovers that after guessing 3 and hearing "lower", the rule forces the guess 2, on turn 3. That is a win. But guessing $1, 3, 5, 7, \ldots$ puts 5 on turn 3 and 7 on turn 4, and the bookkeeping of which numbers land on which parity gets muddled. The student has the right raw material (forced guesses) and no organising principle, so the tinkering stalls somewhere near $1/2$.

The model chain. **Step 1: the strategy.** Guess, in order, the integers in $[1, 2002]$ that are not congruent to 2 modulo 3:

$$1,\ 3,\ 4,\ 6,\ 7,\ 9,\ \ldots,\ 3k,\ 3k+1,\ \ldots$$

continuing as long as the answer is "higher". The guess $3k$ is made on turn $2k$ and the guess $3k+1$ on turn $2k+1$. Every such guess is feasible when it is made, since all earlier guesses were smaller than n.

Step 2: three cases.

- $n = 3k+1$ $(0 \le k \le 667)$. All earlier guesses are below n, so n is reached on turn $2k+1$, which is odd. Win.
- $n = 3k$ $(1 \le k \le 667)$. n is reached on turn $2k$, even. Loss.
- $n = 3k+2$ $(0 \le k \le 666)$. Turn $2k+1$ guesses $3k+1$: "higher". Turn $2k+2$ guesses $3k+3 \le 2001$: "lower". Now the only feasible integer is $3k+2$, which the rules force on turn $2k+3$, odd. Win.

Step 3: count. The winning n are the 668 numbers $\equiv 1$ and the 667 numbers $\equiv 2 \pmod 3$ in $[1, 2002]$ (note $2002 = 3 \cdot 667 + 1$). So the probability of winning is

$$\frac{1335}{2002} > \frac{2}{3} \iff 3 \cdot 1335 = 4005 > 4004 = 2 \cdot 2002. \qquad \square$$

Layer 2 THE HOW — the bifurcation

Bifurcating point. Both chains discover the same raw fact: if the feasible set shrinks to a single number, the next guess is forced and correct. They split at what is done with it. (N) treats forced guesses as accidents that happen along the way and counts outcomes number by number. (M) treats a forced guess as a *resource* — a win you can schedule — and asks how cheaply each one can be bought. The answer is one even-turn guess that brackets a single number from above, and repeating that purchase every three integers is the whole strategy.

> *Guessing $g+1$ right after $g-1$, and hearing "lower", leaves exactly one feasible number. The rules then hand it to me on the next turn. So a gap of one number is a guaranteed hit exactly two turns after the guess below it — I choose its parity by choosing where the gap goes.*

> *Every guess is a node in a binary search tree and the turn is the node's depth. I want as many nodes as possible at odd depth. An even-depth node can have at most two children, so the best pattern is: every even-depth node has two odd-depth children. That is the pattern* $1 \to 3 \to \{2,4\} \to 6 \to \{5,7\} \to \cdots$.

- ▷ **Cue:** a guessing game with higher/lower feedback where the *number* or *parity* of guesses matters, not only success $\longrightarrow$ **Move:** draw the strategy as a binary search tree; the turn on which n is found is the depth of the node n
- ▷ **Cue:** a rule that forces your next move when only one option remains $\longrightarrow$ **Move:** treat the forced move as a free win you can schedule; engineer situations where the feasible set has size one
- ▷ **Cue:** a target fraction such as $2/3$ in a counting problem $\longrightarrow$ **Move:** look for a period-3 pattern that wins on two residues out of three
- ▷ **Cue:** you want to maximise the number of tree nodes of one parity of depth $\longrightarrow$ **Move:** bound by counting parent–child pairs: each node of the other parity has at most two children

Layer 3 THE WHY

Why a period of three, and why is $2/3$ the natural target rather than an arbitrary one? Go to the definition of a strategy. Because of the feasibility rule, every strategy is a binary search tree on $\{1,\dots,2002\}$: the first guess is the root; the numbers below it form the left subtree, those above it the right subtree, and so on recursively. Each number appears exactly once as a node, and n is guessed on the turn equal to its depth (root at depth 1). So the question is exactly: *how many nodes of a binary tree with* 2002 *nodes can sit at odd depth?*

Now count. Let O and E be the numbers of nodes at odd and even depth. Every odd-depth node except the root has a parent, and that parent is at even depth. Each node has at most two children. So

$$O-1 \le 2E, \qquad O+E=2002 \quad\Longrightarrow\quad 3O \le 2\cdot 2002+2 = 4006 \quad\Longrightarrow\quad O \le 1335.$$

Equality needs every even-depth node to have two children — and a node at even depth with two children is a guess with one number on each side still feasible, which is exactly the "bracket a single number" pattern. This is where the period 3 comes from: one even-depth node carries two odd-depth nodes, a block of three. The strategy $1,3,4,6,7,\dots$ is the tree in which the right spine alternates odd, even, odd, even, and each even node $3k$ has the leaf $3k-1$ hanging to its left. So the strategy is not something to be found by trial; it is what the inequality $O-1 \le 2E$ tells you to build when you ask for equality.

Kedlaya and Ng begin by writing down the strategy $1,3,4,6,7,9,\dots$ without saying where it came from. Here is where it comes from: the tree bound above says at most two odd-depth nodes per even-depth node, the forcing rule says a leaf costs nothing to reach once it is bracketed, and the cheapest way to make every even node a double parent is to walk up the right spine with each even node bracketing one leaf. Their note proves optimality by a recursion $a_m = 1+\max_k(b_{k-1}+b_{m-k})$, $b_m = \max_k(a_{k-1}+a_{m-k})$ with $a_m = \lfloor (2m+1)/3 \rfloor$, $b_m = \lfloor 2m/3 \rfloor$; the depth-counting argument above reaches the same bound $\lfloor (2m+1)/3 \rfloor$ in two lines.

Caveat. In the official note's induction, the two displayed bounds are interchanged: it should read $1+b_{k-1}+b_{m-k} \le \lfloor (2m+1)/3 \rfloor$ and $a_{k-1}+a_{m-k} \le \lfloor 2m/3 \rfloor$. Also, equality in the second is not attained at $k=1$ when $3 \mid m$ (e.g. $m=3$: $a_0+a_2 = 1 < 2$); $k=2$ attains it for $m \ge 2$. The recursion's values were checked by exhaustive computation for $m < 60$ and agree with the stated formulas; only the wording of the note is affected, and the main solution is unaffected.

The natural attempts were not stupid. Binary search optimises the wrong quantity (length, not parity), and the linear scan gives away the forced-guess resource by never leaving a gap. The zombie version of this problem is simulating strategies number by number; the reroute is the one-line tree count, which tells you both the ceiling and the shape of the strategy that reaches it.

Layer $+\alpha$ BACKTESTING

1. The first useful cue is the bold **must**: after a wrong guess you are confined to feasible numbers. On seeing it, write "strategy = binary search tree on $[1, 2002]$; turn = depth".
2. Count nodes by depth parity: $O - 1 \leq 2E$ gives the ceiling 1335 and demands that every even-depth node have two children; build the tree that achieves this.
3. Hesitation comes at "which numbers do I skip?". It is reinforced away by the forced-guess observation: a single-number gap bracketed from above is taken on the next turn, so gaps should sit directly below even-turn guesses.
4. The trap is binary search, which is the right tool for the wrong objective, and the linear scan, which never uses the forcing rule and caps out at $1/2$.
5. When a process can be described as a tree (search, recursion, a game), and the objective concerns levels or parities, count parent–child relations between levels before designing anything; the counting gives both the bound and the extremal shape.
6. Reusable decision rule: *if a constraint forces your move when only one option remains, treat forced moves as free and design around creating them; then bound the achievable total by a local count (each object of type A supports at most c objects of type B) and build the configuration that makes the local count tight.*

2002 B5

2002 B5. A palindrome in base b is a positive integer whose base-b digits read the same backwards and forwards; for example, 2002 is a 4-digit palindrome in base 10. Note that 200 is not a palindrome in base 10, but it is the 3-digit palindrome 242 in base 9, and 404 in base 7. Prove that there is an integer which is a 3-digit palindrome in base b for at least 2002 different values of b.

Fields: number theory; base representations; polynomial identities **Position:** B5 — late in the paper, but the difficulty is entirely in finding the construction; once found, the verification is three lines **Answer:** Take $M = 2002!$ and $N = M^2$. For each $d = 1, \ldots, 2002$, in base $b = M/d - 1$ the number N has digits d^2, $2d^2$, d^2.

Layer 1 THE WHAT — surface analysis

The natural attempt. The strong student writes the definition: N is a 3-digit palindrome in base b when

$$N = ab^2 + cb + a, \qquad 1 \leq a \leq b - 1, \quad 0 \leq c \leq b - 1.$$

Then the student tries to make one N satisfy this for many b at once, and the natural thing is to fix the digits and vary b: take $a = 1$, so $N - 1 = b(b + c)$ with $0 \leq c < b$. That asks for $N - 1$ to have many factorisations $b \cdot (b + c)$ with $b \leq b + c < 2b$, i.e. many divisors of $N - 1$ in the narrow window $\left(\sqrt{(N-1)/2}, \sqrt{N-1}\right]$. This is true for suitable N (Kedlaya and Ng's second note builds one), but constructing it needs products of consecutive integers and a limiting argument in m. Alternatively the student looks at the example in the statement, $200 = 242_9 = 404_7$, and tries to see a pattern in $2 \cdot 81 + 4 \cdot 9 + 2$ and $4 \cdot 49 + 0 + 4$ — and does not, because the example was chosen to illustrate the definition, not the method. The student is now doing divisor-counting in a window, with 2002 bases to land, which is heavy.

The model chain. **Step 1: a palindrome that is a square.** For any base b and positive integer d,

$$d^2(b+1)^2 = d^2\,b^2 + 2d^2\,b + d^2.$$

If $2d^2 \le b-1$, the three coefficients d^2, $2d^2$, d^2 are legitimate base-b digits (each in $[0, b-1]$, with the leading one nonzero), so $d^2(b+1)^2$ is the 3-digit palindrome $(d^2)(2d^2)(d^2)$ in base b.

Step 2: one square, many roots. Let $M = 2002!$ and $N = M^2$. For each $d \in \{1, \ldots, 2002\}$, d divides M, so $b_d = M/d - 1$ is a positive integer and $M = d(b_d + 1)$. Hence

$$N = M^2 = d^2(b_d+1)^2.$$

Step 3: the digits are digits. We need $2d^2 \le b_d - 1$, i.e. $2d^2 + 2 \le M/d$. Since $d \le 2002$, $M/d \ge 2001!$, while $2d^2 + 2 \le 2 \cdot 2002^2 + 2 < 2001!$ (already $11! = 39\,916\,800 > 2 \cdot 2002^2 + 2 = 8\,016\,010$). So Step 1 applies, and N is a 3-digit palindrome in base b_d.

Step 4: the bases are distinct. $b_d = M/d - 1$ is strictly decreasing in d, so $b_1, \ldots, b_{2002}$ are 2002 different bases. □

Layer 2 THE HOW — the bifurcation

Bifurcating point. Both chains start from $N = ab^2 + cb + a$. They split at the choice of what to hold fixed. (N) fixes the digits ($a = 1$) and lets the base vary freely, which turns the problem into locating many divisors of $N - 1$ in a window. (M) fixes the *shape* of the digits, $c = 2a$, which turns the palindrome into a perfect square $a(b+1)^2$, and then lets both the digit and the base vary together. That choice converts "many bases" into "many ways to split one number M as $d \cdot (b+1)$", and $M = 2002!$ supplies those for free.

> 121 *is* 11^2 *in every base:* $b^2 + 2b + 1 = (b+1)^2$. *So a palindrome can be a perfect square with the base sitting inside the square root. If I scale the digits by* d^2, *I get* $(d(b+1))^2$. *One square* M^2 *is then a palindrome in base* b *whenever* $M = d(b+1)$ *with* d *small.*

> *I need* 2002 *different ways to write* $M = d(b+1)$ *with* d *small compared with* b. *The number with the most small divisors is a factorial: take* $M = 2002!$ *and* $d = 1, \ldots, 2002$.

- ▷ **Cue:** a palindromic polynomial in the base, $ab^2 + cb + a$ ⟶ **Move:** choose $c = 2a$ so that it factors as $a(b+1)^2$; palindromes ↔ reciprocal polynomials, and $(b+1)^k$ has palindromic coefficients
- ▷ **Cue:** "find one number with property P for many parameters simultaneously" ⟶ **Move:** find a one-parameter family where P reduces to a divisibility or factorisation, then take the number to be highly divisible (a factorial or lcm)
- ▷ **Cue:** digits of a base-b representation are given by formulae in b ⟶ **Move:** check they lie in $[0, b-1]$, with nonzero leading digit; choose b huge compared with the digit formulae

Layer 3 THE WHY

Why is $c = 2a$ the natural choice and not a rabbit from a hat? Go to the definition. A 3-digit base-b number is the value at $x = b$ of a quadratic $ax^2 + cx + a'$; "palindrome" means the coefficient list is symmetric, $a' = a$. Symmetric coefficient lists are exactly the coefficient lists of reciprocal polynomials, and the most familiar family of those is the binomial rows: $1, 2, 1$ is the coefficient list of $(x+1)^2$. That is the whole content of the schoolroom fact that 121 is a perfect square in every base $b \ge 3$. Once a palindrome can be a perfect square in *every* base, the requirement "palindrome in many bases" becomes "one square number whose root can be written as $d(b+1)$ in many ways", and a question about digits has become a question about divisors. Divisors are what factorials are built to supply.

Why must d be small compared with b? Because the formula $d^2b^2 + 2d^2b + d^2$ is a base-b representation only if every coefficient is a digit, $< b$. This is the only real constraint in the problem, and it is why the construction uses the *small* divisors $d \le 2002$ of a *huge* number M: then $b = M/d - 1$ is enormous and the digits $d^2, 2d^2$ are tiny beside it. Taking $M = 2002!$ is the cheapest way to have 2002 small divisors at once; $\operatorname{lcm}(1, \dots, 2002)$ would do as well, provided one checks it exceeds $d(2d^2 + 2)$, which it does by a wide margin.

The example in the statement, $200 = 242_9$, is of exactly this type: $2 \cdot 81 + 4 \cdot 9 + 2 = 200 = 2 \cdot (9 + 1)^2$, and the digits $2, 4, 2$ are $a, 2a, a$ with $a = 2$. The problem statement handed the construction over in its example. $404_7 = 4 \cdot 49 + 4 = 200$ is the other shape ($c = 0$), which does not factor, and that is the one the reader's eye tends to fix on. When a statement gives a worked example, test whether the example already has the structure you need.

Kedlaya and Ng's solution (due to Dan Bernstein) opens "Put $N = 2002!$. Then for $d = 1, \dots, 2002$, the number N^2 written in base $b = N/d - 1$ has digits $d^2, 2d^2, d^2$." Here is where that comes from: $(b + 1)^2$ is always the palindrome 121; multiplying through by d^2 scales the digits; and requiring $d(b + 1)$ to equal a fixed number for many d forces that number to have many small divisors. Their other two notes (divisors of $N - 1$ in a window; an averaging argument over bases) are genuine alternative proofs, and both are harder, which is the point: the natural attempt with fixed digits *works*, and costs ten times as much.

Layer $+\alpha$ BACKTESTING

1. The first cue is "3-digit palindrome in base b". Write $N = ab^2 + cb + a$ and immediately ask which choice of c makes this polynomial in b factor.

2. Choose $c = 2a$, giving $N = a(b + 1)^2$; with $a = d^2$, $N = (d(b + 1))^2$. Then fix the square root M and vary the divisor d.

3. Hesitation comes at "are d^2 and $2d^2$ really digits?". It is reinforced away by making M so large (2002!) that $b = M/d - 1$ dwarfs $2 \cdot 2002^2$.

4. The trap is fixing the digits (say $a = 1$) and hunting for many bases, which turns into finding many divisors of $N - 1$ in a narrow window — doable, but an analytic construction instead of a three-line one. A second trap is fixing on 404_7 in the example instead of 242_9.

5. When digits in base b are involved, read the digit string as a polynomial in b and ask whether a symmetric or binomial pattern makes it factor; factorisation converts digit questions into divisibility questions.

6. Reusable decision rule: *to make one object satisfy a condition for many parameter values at once, find a family in which the condition becomes "the object factors as (small) × (parameter-dependent)", then choose the object to be highly divisible — a factorial, an lcm, or a product of the needed moduli.*

2002 B6

2002 B6. Let p be a prime number. Prove that the determinant of the matrix

$$\begin{pmatrix} x & y & z \\ x^p & y^p & z^p \\ x^{p^2} & y^{p^2} & z^{p^2} \end{pmatrix}$$

is congruent modulo p to a product of polynomials of the form $ax + by + cz$, where a, b, c are integers. (We say two integer polynomials are congruent modulo p if corresponding coefficients are congruent modulo p.)

Fields: algebra; polynomials over $\mathbb{F}_p$; linear algebra (determinants); number theory (Fermat, Frobenius)
Position: B6 — the last problem on the paper and genuinely hard for anyone who has not met the

Frobenius map; for anyone who has, it is the Vandermonde argument transplanted **Answer:** Modulo p, the determinant equals $x\prod_{i=0}^{p-1}(y+ix)\prod_{i,j=0}^{p-1}(z+ix+jy)$ (the Moore determinant).

Layer 1 THE WHAT — surface analysis

The natural attempt. The strong student expands the determinant:

$$D = xy^pz^{p^2} + yz^px^{p^2} + zx^py^{p^2} - xz^py^{p^2} - yx^pz^{p^2} - zy^px^{p^2},$$

homogeneous of degree $1+p+p^2$. Then the student tries $p=2$: $D = xy^2z^4 + yz^2x^4 + zx^2y^4 - xz^2y^4 - yx^2z^4 - zy^2x^4$, of degree 7, and tries to factor it mod 2. x, y, z each divide every term, so $D = xyz\cdot(\text{quartic})$. The quartic is found, after some pain, to be $(x+y)(x+z)(y+z)(x+y+z)$ mod 2. So for $p=2$ the answer is the product of *all seven* nonzero linear forms over $\mathbb{F}_2$. Now the student has a conjecture but no proof: the factorisation for $p=2$ came from trial division, and trial division by p^2+p+1 linear forms in general p is not a proof strategy. The wall is that the approach is "find the factors", when what is needed is a reason why each factor divides.

The model chain. Work throughout in $\mathbb{F}_p[x,y,z]$, the polynomials with coefficients reduced mod p; "congruent mod p" means "equal in $\mathbb{F}_p[x,y,z]$". Let D be the image of the determinant.

Step 1: Frobenius is linear. For integers a, b, c,

$$(ax+by+cz)^p = ax^p + by^p + cz^p \quad \text{in } \mathbb{F}_p[x,y,z].$$

Indeed every multinomial coefficient $\binom{p}{i,j,k}$ with $i,j,k<p$ is divisible by p (the numerator $p!$ has a factor p, the denominator does not), and $a^p = a$ in $\mathbb{F}_p$ by Fermat. Applying this twice, $(ax+by+cz)^{p^2} = ax^{p^2} + by^{p^2} + cz^{p^2}$.

Step 2: each candidate factor divides D. The candidates are

$$x; \qquad y+ix\ (0\le i\le p-1); \qquad z+ix+jy\ (0\le i,j\le p-1),$$

$1+p+p^2$ linear forms in all. Take $L = z+ix+jy$. Substituting $z = -ix-jy$ is a ring homomorphism $\phi : \mathbb{F}_p[x,y,z] \to \mathbb{F}_p[x,y]$ with kernel (L) (divide by L as a monic polynomial in z). By Step 1, ϕ sends the third column $(z, z^p, z^{p^2})^T$ to $-i\,(x, x^p, x^{p^2})^T - j\,(y, y^p, y^{p^2})^T$, a combination of the first two columns. So $\phi(D) = 0$ and $L \mid D$. Likewise $y+ix$: substituting $y = -ix$ makes the second column $-i$ times the first. And x: setting $x=0$ makes the first column zero.

Step 3: all of them together divide D. The $1+p+p^2$ forms are irreducible (degree 1) and pairwise non-associate: each is normalised to have coefficient 1 on its last-appearing variable in the order x, y, z, and no two coincide. $\mathbb{F}_p[x,y,z]$ is a unique factorisation domain, so their product P divides D.

Step 4: compare degree and one coefficient. P and D are both homogeneous of degree $1+p+p^2$, so $D = \lambda P$ for a constant $\lambda \in \mathbb{F}_p$. The six monomials of D are distinct (the exponents $1, p, p^2$ are distinct), so $xy^pz^{p^2}$ has coefficient 1 in D. In P, the only way to produce z^{p^2} is to take z from each of the p^2 factors $z+ix+jy$; then y^p must come from taking y in each of the p factors $y+ix$; then x from the factor x. So $xy^pz^{p^2}$ also has coefficient 1 in P, and $\lambda = 1$. Hence

$$D \equiv x\prod_{i=0}^{p-1}(y+ix)\prod_{i,j=0}^{p-1}(z+ix+jy) \pmod p. \qquad \square$$

Layer 2 THE HOW — the bifurcation

Bifurcating point. Both chains agree on the object: a determinant of degree $1+p+p^2$ that should factor into linear forms. They split at the question asked of it. (N) asks *what are the*

factors? and looks for them by expanding and dividing. (M) asks *when does the determinant vanish?* — and a determinant vanishes when its columns are dependent. The columns are $v \mapsto (v, v^p, v^{p^2})$ applied to x, y, z, and mod p that map is linear. So the determinant vanishes whenever z is a linear combination of x and y with $\mathbb{F}_p$-coefficients, which says precisely that each $z + ix + jy$ divides it. After that it is degree-counting.

> *This is a Vandermonde-shaped determinant: rows are successive powers. Vandermonde factors because it vanishes when two columns coincide. Here the powers are p-th powers, and mod p raising to the p-th power is additive — so the column map $v \mapsto (v, v^p, v^{p^2})$ is linear, and the determinant should vanish whenever the columns are* linearly *dependent over $\mathbb{F}_p$, not only when two are equal.*

> *I will not find the factors by dividing; I will prove each linear form divides by substituting it to zero, then count degrees. The number of normalised linear forms is $1 + p + p^2$, and the degree is $1 + p + p^2$. That is not a coincidence; that is the proof.*

- ▷ **Cue:** a determinant whose rows are successive powers (Vandermonde shape) $\longrightarrow$ **Move:** find when it vanishes (column dependence), convert each vanishing condition into a factor, then match degree and one coefficient
- ▷ **Cue:** p-th powers modulo a prime p $\longrightarrow$ **Move:** use $(u + v)^p \equiv u^p + v^p$ and $a^p \equiv a$: the map $v \mapsto v^p$ is $\mathbb{F}_p$-linear
- ▷ **Cue:** "prove f is (congruent to) a product of given factors" $\longrightarrow$ **Move:** show each factor divides f by substituting its root, show the factors are coprime, then compare degrees and a single coefficient
- ▷ **Cue:** a count of objects equal to a degree, such as $1 + p + p^2$ lines through the origin in $\mathbb{F}_p^3$ vs degree $1 + p + p^2$ $\longrightarrow$ **Move:** the objects are the roots/factors; the equality is the proof that there are no others

Layer 3 THE WHY

Why is "when does it vanish?" the natural question and not a trick? Go to the definition of a determinant. The determinant is the unique alternating multilinear function of the columns normalised on the identity; its defining property is that it is zero exactly when the columns are linearly dependent (over a field). So the one thing a determinant is designed to detect is dependence. In a Vandermonde matrix with columns $(1, v, v^2)$, dependence happens when two of the v's coincide, and that gives the factors $v_i - v_j$. Here the columns are (v, v^p, v^{p^2}). Over the integers, that map is thoroughly nonlinear; modulo p it is the Frobenius map, and Frobenius is additive: $(u + v)^p \equiv u^p + v^p$, because every interior binomial coefficient $\binom{p}{k}$ is divisible by p. So mod p the column map is linear, and the columns are dependent whenever x, y, z are. The problem's own wording hands this over: "modulo p" and exponents p, p^2 together are the signature of Frobenius, and "product of polynomials of the form $ax + by + cz$" is the signature of a determinant that vanishes on linear subspaces.

Why exactly those $1 + p + p^2$ forms? The forms $ax + by + cz$ mod p, up to nonzero scalar multiples, correspond to the lines through the origin in $\mathbb{F}_p^3$ (equivalently, the points of the projective plane over $\mathbb{F}_p$). There are $(p^3 - 1)/(p - 1) = p^2 + p + 1$ of them. Normalising the last nonzero coefficient to 1 lists them as x; $y + ix$; $z + ix + jy$. Each vanishing condition "$ax + by + cz = 0$" is a way for (x, y, z) to be dependent, so each form divides D, and the degree $1 + p + p^2$ of D leaves room for exactly those and nothing else. The degree count is not a check performed after the fact; it is the reason the answer has to be this product.

Kedlaya and Ng's main solution verifies the product by first proving $\prod_i (y + ix) \equiv y^p - x^{p-1}y$ and then multiplying out a long chain of congruences. Their note gives the argument above and calls it "a simpler conceptual proof". Here is where it comes from: the identity

$\prod_i (y + ix) \equiv y^p - x^{p-1}y$ is itself the 2×2 case, $\det \begin{pmatrix} x & y \\ x^p & y^p \end{pmatrix} = xy^p - x^p y = x\,(y^p - x^{p-1}y)$, whose factors are x and the $y + ix$ for the same reason — the second column is a multiple of the first exactly when $y = -ix$. The 3×3 case is the same sentence one dimension up. That sentence generalises to the $n \times n$ Moore determinant, with factors indexed by the points of the projective space $\mathbb{P}^{n-1}(\mathbb{F}_p)$.

This problem punishes zombie maths hard. Expanding and trial-dividing is feasible for $p = 2$ and hopeless for $p = 5$, and it never produces a reason. The reroute costs one sentence: a determinant is a dependence detector, and mod p the columns depend linearly on (x, y, z).

Layer $+\alpha$ BACKTESTING

1. The first cue is the combination "modulo p" with exponents $1, p, p^2$. Write down $(u + v)^p \equiv u^p + v^p \pmod{p}$ and $a^p \equiv a$ at once.

2. Treat the determinant as a function of columns (v, v^p, v^{p^2}), which is $\mathbb{F}_p$-linear in v; show each linear form divides by substitution; compare degree and the coefficient of $xy^p z^{p^2}$.

3. Hesitation comes at "which linear forms?". It is reinforced away by counting: normalised forms over $\mathbb{F}_p$ in three variables number $1 + p + p^2$, exactly the degree.

4. The trap is expanding and factoring by trial division, which works for $p = 2$ and gives a conjecture without a proof. A subtler trap is forgetting to check that the forms are pairwise coprime and that the leading constant is 1 rather than some other element of $\mathbb{F}_p$.

5. Whenever a polynomial is to be shown equal to a product of linear factors, ask "where does it vanish?" before "what does it expand to?"; for a determinant, vanishing means column dependence.

6. Reusable decision rule: *to prove $f = c\prod L_k$, prove each $L_k \mid f$ by substituting $L_k = 0$, prove the L_k are coprime, then match total degree and a single coefficient. The degree match is the proof there are no other factors.*