STATISTICAL REPORT · GENERATED 2026-11-15 00:51 UTC

Modular arithmetic on Codeforces: 1e9+7, 998244353 and the long tail

This report covers 10,044 problems with full statements, which is 87.2% of the 11,520-problem Codeforces archive. Statements cover contests from 2010 to February 2025. The interactive dashboard has the same data with filters.

1. Executive summary

Problems analyzed
10,044
87.2% of the archive

Use modular arithmetic
993
9.9% of analyzed problems

1e9+7
479
48.2% of modular problems

998244353
347
34.9% of modular problems

Crossover year
2019
998244353 leads every year since

About one Codeforces problem in ten (9.9%) asks for its answer modulo something. Two moduli cover 83% of those problems. The older one, 1e9+7, was effectively the only big choice until 2017. 998244353 replaced it as the most common modulus from 2019 and now accounts for roughly two thirds of new big-two usage. A further 116 problems take the modulus as input, and 24 other fixed moduli each appear only a handful of times.

- **998244353 overtook 1e9+7 in 2019.** Up to 2017, 1e9+7 appeared in 235 problems against 6 for 998244353. In 2018 the two were nearly even (35 vs 33). From 2019 on, 998244353 leads every year, and across 2022–2024 it took 66% of the problems that used either modulus.
- **Moduli are a hard-problem phenomenon.** Only 0.9% of problems rated below 1200 use a modulus, against 19.6% at 2400–2799 and 21.4% at 2800+. 998244353 problems average rating 2550, against 2340 for 1e9+7. Below 1600, 1e9+7 is still the default (998244353 holds only 28% of the big two at 1200–1599). From 2400 up the two are split almost evenly.
- **fft is the clearest modulus marker.** 73.7% of fft-tagged problems use a modulus, and 86% of those that use one of the big two pick 998244353. That fits $998244353 = 119 \cdot 2^{23} + 1$, which supports number-theoretic transforms of length up to 2^{23}. matrices problems lean the other way: 71% of their big-two usage is 1e9+7.
- **combinatorics nearly always means a modulus** (69.0% of 704 problems, 7.0× the archive rate). probabilities follows at 32.8%, where answers are fractions reported as $P \cdot Q^{-1}$. 998244353 is the more common choice there (59% of the big two).

- **Contest series have house styles.** In Educational rounds 998244353 takes 73% of big-two usage, and in Global rounds 80%, against 21% in Div. 2 rounds, which have a longer pre-2018 history. Div. 1 (16.1%) and Div. 1 + Div. 2 (16.6%) rounds use moduli far more often than Div. 3 (3.7%) or Div. 4 (2.3%).

---

2. Data collection methodology

2.1 Sources

- **Official Codeforces API**: `problemset.problems` (every archive problem with tags, rating and solved count) and `contest.list` (contest names, start times). Requests carry an honest crawler User-Agent, are spaced at least 2.1 s apart (Codeforces allows one call per 2 s), and are cached on disk. Re-running within the cache window makes no network calls.
- **Statements.** The API has no statement text, and Codeforces blocks automated access to statement pages, so statements were *not* scraped. They come from the published open-r1/codeforces dataset (CC BY 4.0), which provides statement, input, output, notes, time and memory limits, system-test count and, for 6,078 problems, the editorial. It is joined to the API archive by (contest id, index). The dataset's `aliases` field recovers 242 problems that Codeforces lists under the parallel Div. 1/Div. 2 contest.

2.2 Storage, resumability and data quality

Everything is stored in one SQLite database with these tables: contests, problems, statements, detections, features, validation and crawl_state. API responses are written atomically through `.part` files. Dataset shards are downloaded with HTTP range-resume, and each shard is marked done in `crawl_state` once imported, so an interrupted run continues where it stopped. All writes are upserts keyed by (contest id, index), so re-runs refresh rows instead of duplicating them. Malformed records (missing ids or indices, non-numeric contest ids) are skipped. Duplicate API rows are dropped. Problems shared between a Div. 1 and Div. 2 round appear once in the API archive and are counted once.

2.3 Detecting moduli

1. **Normalization.** LaTeX delimiters (`$$$`), thin and non-breaking spaces (`\,`, ~), digit-group separators (`998\,244\,353`, `998,244,353`, `1 000 000 007`), `\cdot`/`\times`, `\bmod`/`\pmod`, braces in exponents and unicode superscripts are all rewritten to plain arithmetic.
2. **Exact evaluation.** Every expression governed by a modulus phrase ("modulo X", "remainder ... divided by X", "mod (X)", "% (X)", "Let M = X") is evaluated exactly. `1e9+7`, `10^9+7`, `10^{9}+7`, `1000000007` and `1 000 000 007` all become 1000000007, and `119·2^23+1` becomes 998244353.
3. **Flattened superscripts.** Older statements in the dataset lost their superscripts, so 10^9+7 reads "109 + 7". Inside a modulus phrase only, "10k ± c" is re-read as 10^k ± c, because a literal modulus of 116 is never intended.
4. **False-positive guards.** Values below 1000 (parity, "mod 3", digit sums) are excluded. Results such as `(x) mod M = 998244345` are not treated as moduli. Variable definitions only count for the letter actually used after "mod". Matches found only in sample notes, or not about the answer and neither prime nor a power of two (e.g. "2024 mod 1000"), are downgraded to uncertain. Divisibility (`y mod x = 0`), cyclic indexing (`(i mod n) + 1`) and "x mod y denotes" definitions are ignored.
5. **Automatic discovery.** Any value meeting these rules counts as a modulus, which is how the 24 long-tail moduli were found. Prime moduli with at least 3 confirmed uses (998244353 (348), 1e6+3 (9), 1e9+9 (12), 1e9+7 (476)) are also searched for as bare numbers. A bare match only marks a problem *uncertain*, never confirmed.

> **Status of each problem: confirmed** means a numeric modulus is stated as a requirement. **Variable** means the modulus is an input (e.g. "print it modulo p", or "3 ≤ p ≤ 10^9+9, p is prime"). **Uncertain** means a frequent modulus

> appears with no modulo phrase. **None** means no modulus. Headline statistics count *confirmed + variable* problems. Uncertain problems (15) are listed in §9 and can be included from the dashboard.

3. Coverage

The API archive holds 11,520 problems. Statements were available for 10,044 (87.2%). Of the 1,476 problems without a statement, 1,343 come from contests after the dataset snapshot (February 2025). The remainder are mostly special, interactive-only or unlisted problems. Status counts over the whole archive: confirmed 877, variable 116, uncertain 15, none 9,036, no statement 1,476.

4. Overall frequency and ranking

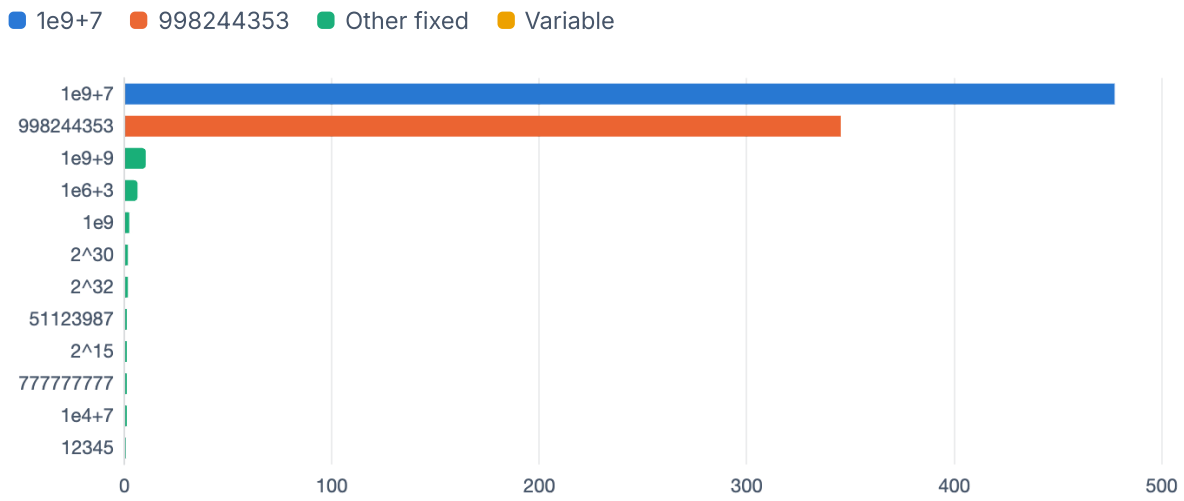


Figure 1. The 12 most-used fixed moduli (number of problems where the modulus is confirmed). Color follows the modulus: blue 1e9+7, orange 998244353, green others.

877 problems state a fixed modulus and 116 take a variable one. In total, 26 distinct fixed moduli were found. Only 2 problems state two different fixed moduli, and none states more than two. The full ranking:

#	Modulus	Problems	% of modular problems	% of analyzed problems
1	1e9+7	479	48.24%	4.77%
2	998244353	347	34.94%	3.45%
3	1e9+9	12	1.21%	0.12%
4	1e6+3	8	0.81%	0.08%
5	1e9	4	0.40%	0.04%
6	2^30	3	0.30%	0.03%
7	2^32	3	0.30%	0.03%
8	51123987	2	0.20%	0.02%
9	2^15	2	0.20%	0.02%
10	777777777	2	0.20%	0.02%
11	1e4+7	2	0.20%	0.02%
12	12345	1	0.10%	0.01%
13	1e8	1	0.10%	0.01%
14	1e3+13	1	0.10%	0.01%
15	7340033	1	0.10%	0.01%
16	95542721	1	0.10%	0.01%
17	1051131	1	0.10%	0.01%
18	1e3+9	1	0.10%	0.01%
19	2^64	1	0.10%	0.01%
20	490019	1	0.10%	0.01%
21	2^58	1	0.10%	0.01%
22	998244853	1	0.10%	0.01%
23	1234567891	1	0.10%	0.01%
24	31607	1	0.10%	0.01%
25	999999893	1	0.10%	0.01%
26	20240401	1	0.10%	0.01%

5. Difficulty

	Unrated	<1200	1200-1599	1600-1999	2000-2399	2400-2799	2800+
1e9+7	2.2%	0.3%	1.7%	4.1%	8.1%	8.3%	9.1%
998244353	1.8%	0.1%	0.7%	2.0%	4.3%	8.1%	9.0%
Other fixed	0.9%	0.0%	0.4%	0.2%	0.7%	1.3%	0.9%
Variable	0.4%	0.6%	0.6%	0.9%	1.2%	1.9%	2.4%
Any modulus	5.3%	0.9%	3.4%	7.2%	14.3%	19.6%	21.4%

Scale (share of problems in the column): 0.0%+ 4.3%+ 8.5%+ 12.8%+ 17.1%+

Figure 2. Heatmap: % of problems in each rating band that use each modulus. Darker cells mean a higher share.

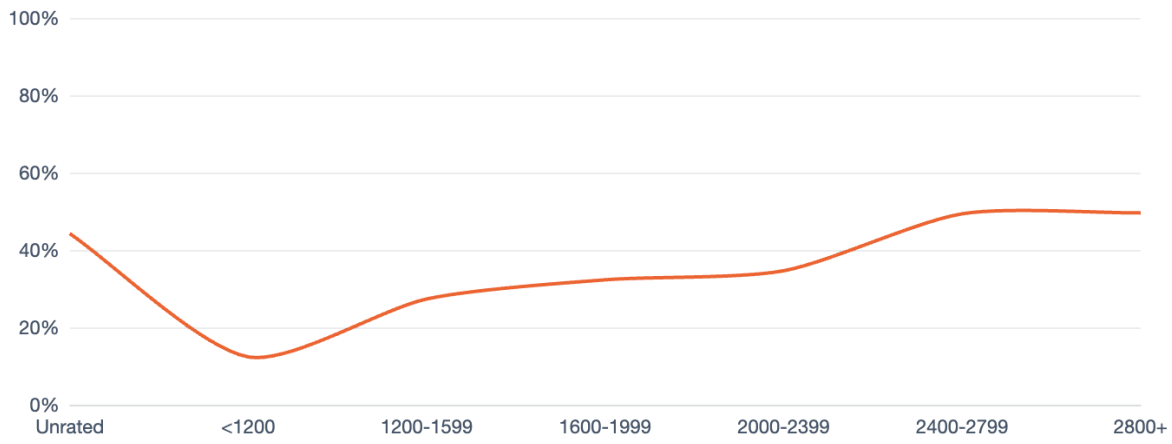


Figure 3. 998244353's share of big-two problems in each rating band.

Rating band	Problems	Modular	1e9+7	998244353	Other fixed	Variable	998244353 share of big two
Unrated	227	5.29%	2.20% (5)	1.76% (4)	0.88% (2)	0.44% (1)	44%
<1200	1,992	0.95%	0.35% (7)	0.05% (1)	0.00% (0)	0.55% (11)	12%
1200-1599	1,679	3.39%	1.73% (29)	0.66% (11)	0.42% (7)	0.60% (10)	28%
1600-1999	1,887	7.21%	4.08% (77)	1.96% (37)	0.21% (4)	0.95% (18)	32%
2000-2399	1,648	14.32%	8.07% (133)	4.31% (71)	0.73% (12)	1.21% (20)	35%
2400-2799	1,365	19.56%	8.28% (113)	8.13% (111)	1.25% (17)	1.90% (26)	50%
2800+	1,246	21.35%	9.07% (113)	8.99% (112)	0.88% (11)	2.41% (30)	50%

The modular share rises almost monotonically with rating. Problems that use a modulus average 2340 (1e9+7), 2550 (998244353), 2347 (other fixed) and 2244 (variable), against 1798 for problems without one. 998244353 is the modulus of hard problems, while low-rated counting problems still mostly default to 1e9+7.

6. Contest type and time

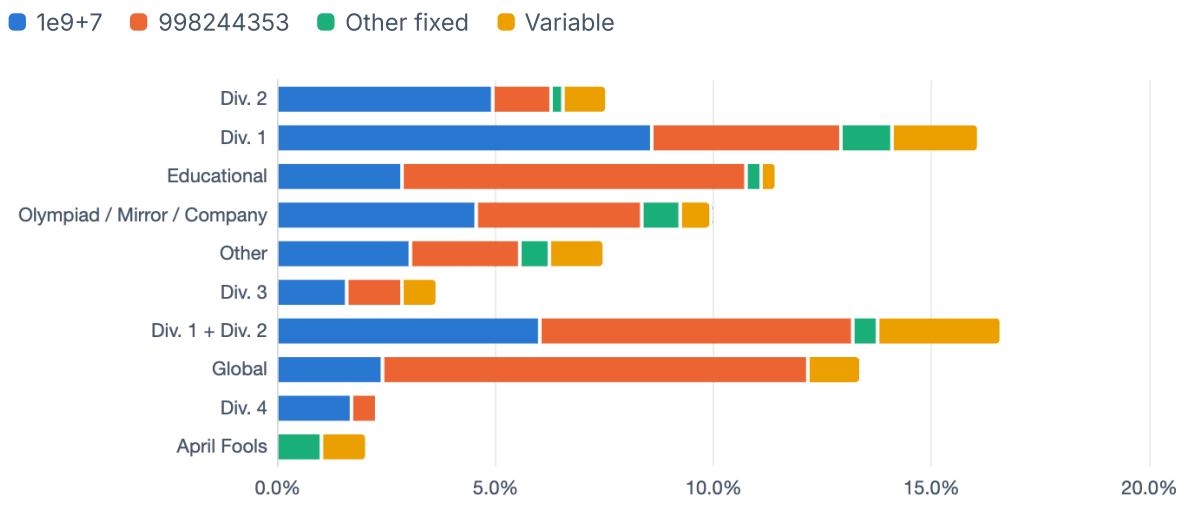


Figure 4. Modular usage by contest type (types with at least 20 problems), as % of that type's problems.

Contest type	Problems	Modular	1e9+7	998244353	Other fixed	Variable	998244353 share of big two
Div. 2	3,282	7.59%	4.97% (163)	1.34% (44)	0.27% (9)	1.01% (33)	21%
Div. 1	1,705	16.13%	8.62% (147)	4.34% (74)	1.17% (20)	1.99% (34)	33%
Educational	1,141	11.48%	2.89% (33)	7.89% (90)	0.35% (4)	0.35% (4)	73%
Olympiad / Mirror / Company	1,133	9.97%	4.59% (52)	3.80% (43)	0.88% (10)	0.71% (8)	45%
Other	876	7.53%	3.08% (27)	2.51% (22)	0.68% (6)	1.26% (11)	45%
Div. 3	863	3.71%	1.62% (14)	1.27% (11)	0.00% (0)	0.81% (7)	44%
Div. 1 + Div. 2	529	16.64%	6.05% (32)	7.18% (38)	0.57% (3)	2.84% (15)	54%
Global	246	13.41%	2.44% (6)	9.76% (24)	0.00% (0)	1.22% (3)	80%
Div. 4	173	2.31%	1.73% (3)	0.58% (1)	0.00% (0)	0.00% (0)	25%
April Fools	96	2.08%	0.00% (0)	0.00% (0)	1.04% (1)	1.04% (1)	–

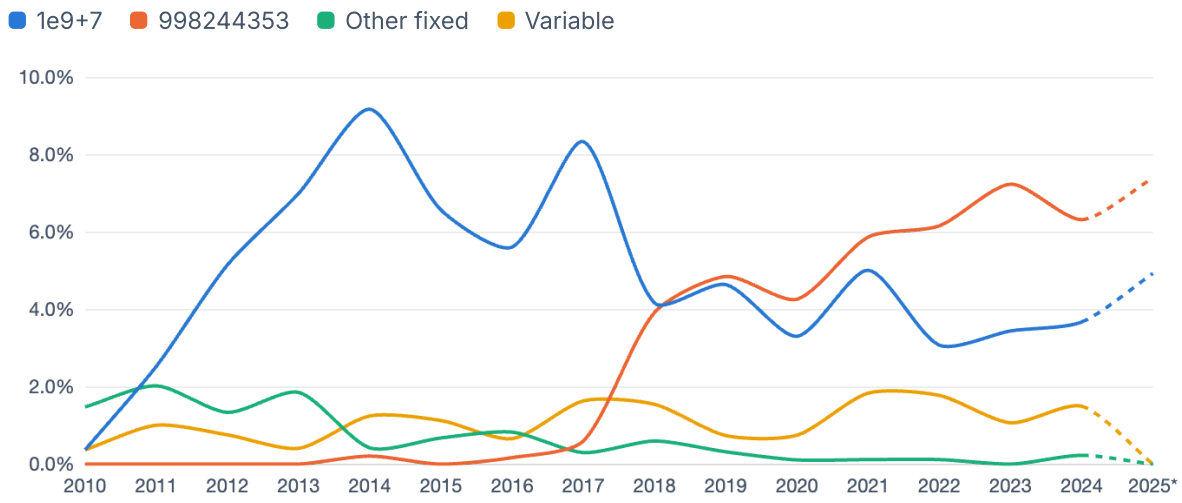


Figure 5. % of each year's problems using each modulus. 2025 is partial and drawn dashed: the statement dataset ends in February 2025.

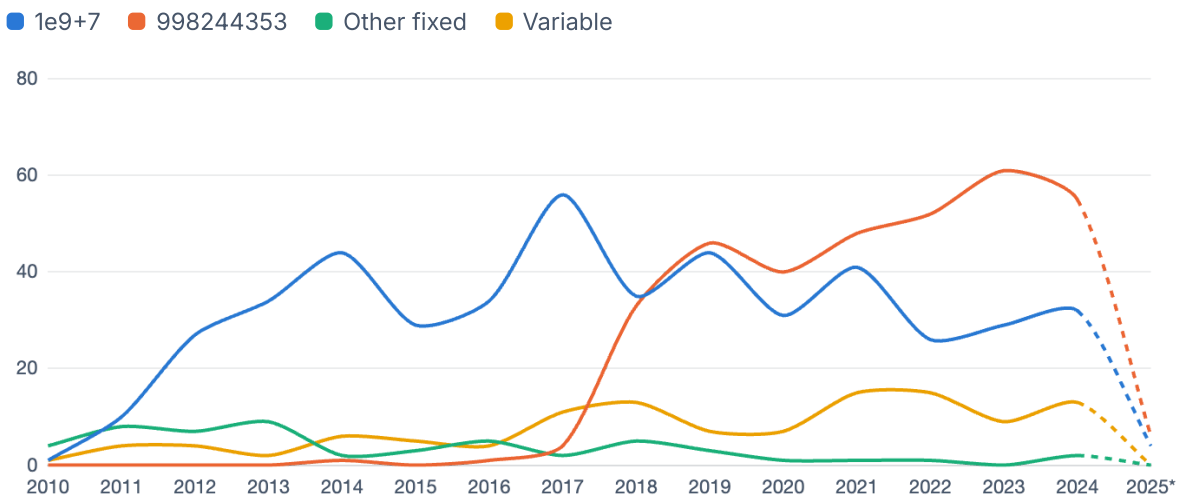


Figure 6. The same series as raw problem counts.

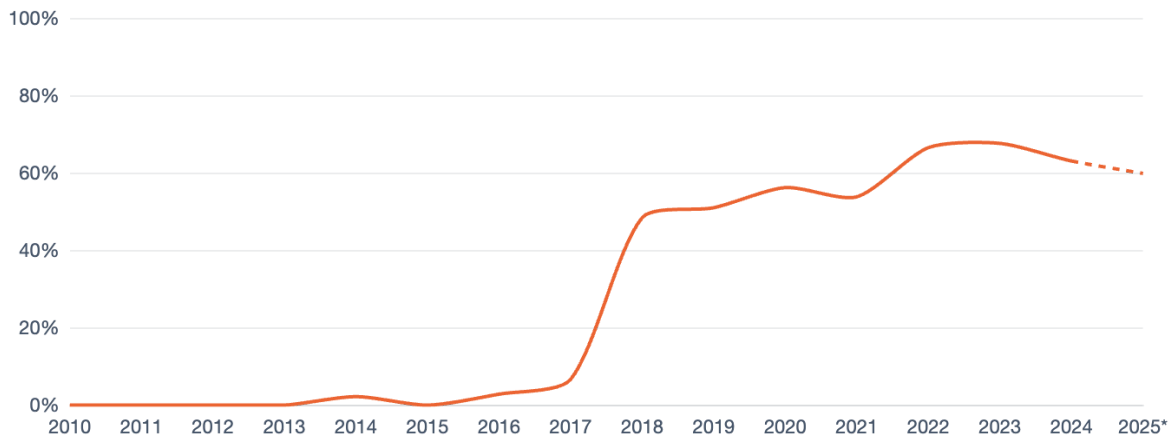


Figure 7. 998244353's share of the big two by year.

Year	Problems	Modular	1e9+7	998244353	Other fixed	Variable	998244353 share of big two
2010	271	2.21%	0.37% (1)	0.00% (0)	1.48% (4)	0.37% (1)	0%
2011	395	5.57%	2.53% (10)	0.00% (0)	2.03% (8)	1.01% (4)	0%
2012	523	7.27%	5.16% (27)	0.00% (0)	1.34% (7)	0.76% (4)	0%
2013	485	9.28%	7.01% (34)	0.00% (0)	1.86% (9)	0.41% (2)	0%
2014	479	11.06%	9.19% (44)	0.21% (1)	0.42% (2)	1.25% (6)	2%
2015	441	8.39%	6.58% (29)	0.00% (0)	0.68% (3)	1.13% (5)	0%
2016	605	7.27%	5.62% (34)	0.17% (1)	0.83% (5)	0.66% (4)	3%
2017	671	10.88%	8.35% (56)	0.60% (4)	0.30% (2)	1.64% (11)	7%
2018	840	10.24%	4.17% (35)	3.93% (33)	0.60% (5)	1.55% (13)	49%
2019	947	10.56%	4.65% (44)	4.86% (46)	0.32% (3)	0.74% (7)	51%
2020	936	8.44%	3.31% (31)	4.27% (40)	0.11% (1)	0.75% (7)	56%
2021	817	12.85%	5.02% (41)	5.88% (48)	0.12% (1)	1.84% (15)	54%
2022	843	11.15%	3.08% (26)	6.17% (52)	0.12% (1)	1.78% (15)	67%
2023	841	11.77%	3.45% (29)	7.25% (61)	0.00% (0)	1.07% (9)	68%
2024	869	11.74%	3.68% (32)	6.33% (55)	0.23% (2)	1.50% (13)	63%
2025	81	12.35%	4.94% (4)	7.41% (6)	0.00% (0)	0.00% (0)	60%

By calendar month

Month-of-year has no meaningful effect. Monthly modular shares range from 8.5% to 12.1%, which is within noise for roughly 700–960 problems per month.

Month	Problems	Modular	998244353 share
Jan	706	11.5%	40%
Feb	831	9.5%	40%
Mar	885	8.6%	43%
Apr	928	9.5%	36%
May	778	8.9%	43%
Jun	839	8.5%	25%
Jul	816	9.6%	46%
Aug	698	9.3%	31%
Sep	848	11.7%	38%
Oct	965	9.8%	37%
Nov	818	9.7%	48%
Dec	932	12.1%	66%

7. Problem tags

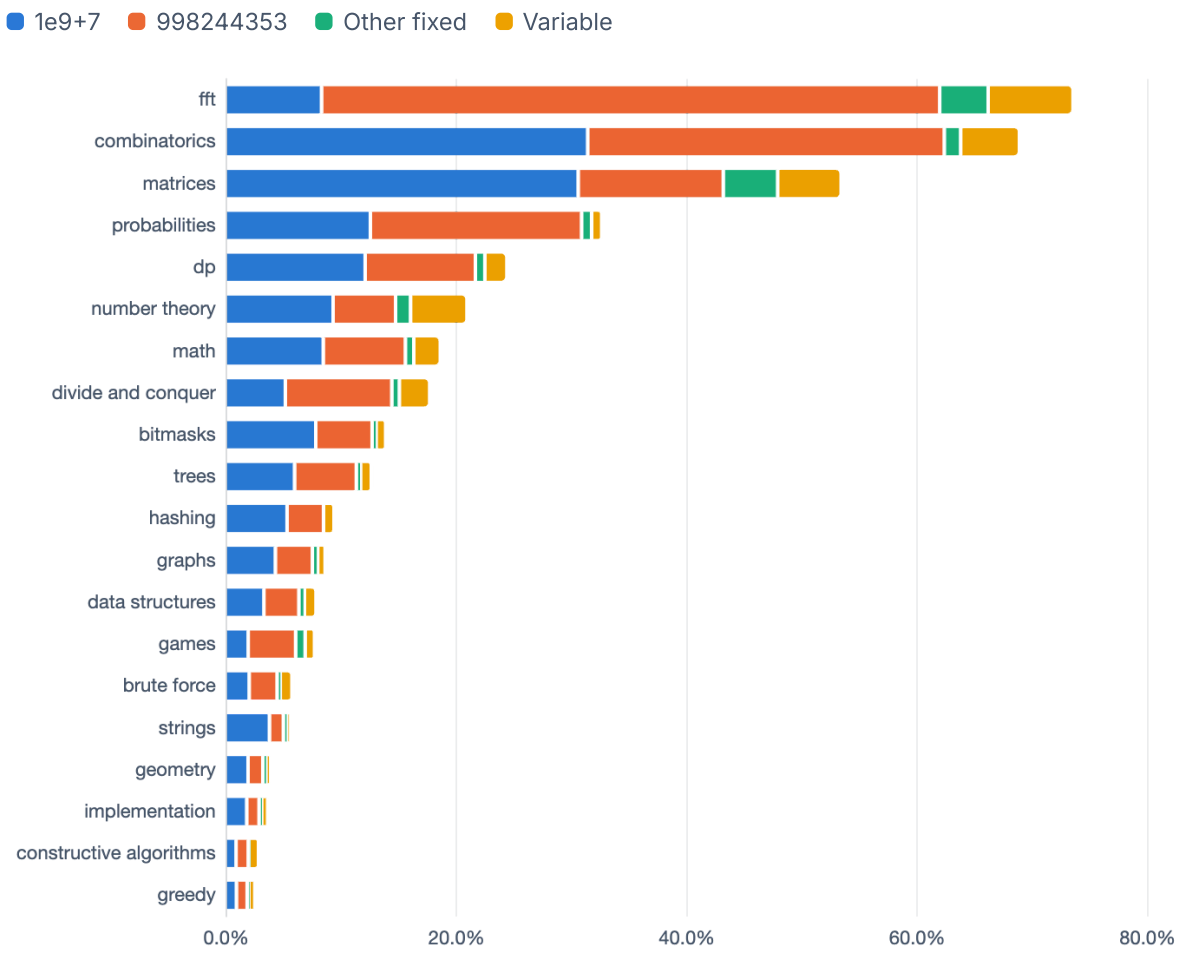


Figure 8. Modular usage within selected tags (% of each tag's problems), stacked by modulus.

Tag	Problems	Modular	Lift vs archive	1e9+7	998244353	Other fixed	Variable	998244353 share
fft	95	73.7%	7.45×	8	51	4	7	86%
combinatorics	704	69.0%	6.98×	222	218	10	36	50%
matrices	127	53.5%	5.42×	39	16	6	7	29%
probabilities	229	32.8%	3.31×	29	42	2	2	59%
dp	2,203	24.5%	2.48×	269	211	18	42	44%
number theory	774	21.1%	2.13×	73	42	10	38	37%
math	2,999	18.7%	1.90×	257	214	22	69	45%
divide and conquer	303	17.8%	1.80×	16	28	2	8	64%
bitmasks	593	14.0%	1.42×	47	29	2	5	38%
trees	839	12.8%	1.29×	51	45	3	8	47%
hashing	221	9.5%	0.96×	12	7	0	2	37%
graphs	1,088	8.7%	0.88×	48	35	5	7	42%
data structures	1,814	7.9%	0.80×	62	55	9	18	47%
games	243	7.8%	0.79×	5	10	2	2	67%
brute force	1,775	5.9%	0.59×	38	43	5	18	53%
strings	744	5.5%	0.56×	29	9	2	1	24%
geometry	391	3.8%	0.39×	8	5	1	1	38%
implementation	2,807	3.7%	0.37×	54	30	6	13	36%
constructive algorithms	1,795	3.0%	0.30×	18	19	1	15	51%
greedy	3,009	2.6%	0.26×	31	28	4	14	47%

Lift is the tag's modular rate divided by the archive rate (9.9%). The most common tag combinations among problems using each modulus:

Modulus	Top tag pairs	Top tag triples
1e9+7	combinatorics + math (141); combinatorics + dp (134); dp + math (121); math + number theory (60); combinatorics + number theory (37)	combinatorics + dp + math (78); combinatorics + math + number theory (27); dp + math + number theory (24)
998244353	combinatorics + math (149); combinatorics + dp (136); dp + math (132); fft + math (42); math + number theory (37)	combinatorics + dp + math (97); combinatorics + fft + math (28); combinatorics + math + number theory (25)
Other fixed	math + number theory (8); combinatorics + dp (5); math + matrices (4); combinatorics + math (4); dp + math (4)	combinatorics + dp + math (3); data structures + divide and conquer + graphs (2); combinatorics + dp + number theory (2)
Variable	math + number theory (30); combinatorics + dp (26); combinatorics + math (22); dp + math (20); brute force + math (12)	combinatorics + dp + math (14); brute force + math + number theory (8); combinatorics + math + number theory (6)

number theory problems use moduli at 21.1% and lean towards 1e9+7 (63% of big-two usage), as do strings (76%), where hashing traditionally uses 1e9+7-style primes.

8. Other dimensions

8.1 Problem index

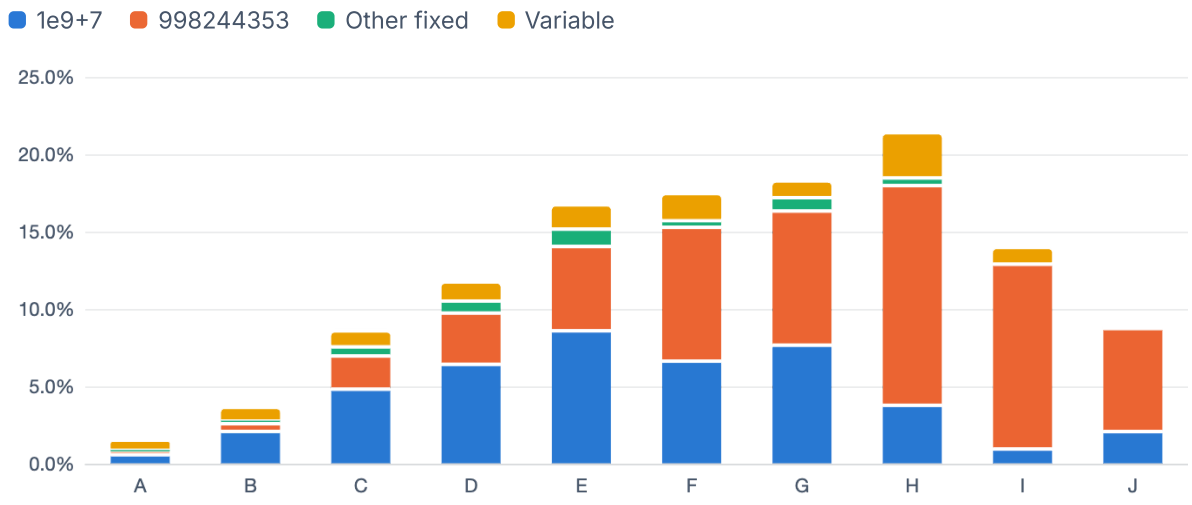


Figure 9. Modular usage by problem index (letters with at least 30 problems).

Index	Problems	Modular	1e9+7	998244353	Other fixed	Variable	998244353 share of big two
A	1,807	1.66%	0.72% (13)	0.11% (2)	0.17% (3)	0.66% (12)	13%
B	1,790	3.80%	2.23% (40)	0.50% (9)	0.17% (3)	0.89% (16)	18%
C	1,532	8.75%	4.96% (76)	2.15% (33)	0.59% (9)	1.04% (16)	30%
D	1,537	11.91%	6.57% (101)	3.32% (51)	0.78% (12)	1.24% (19)	34%
E	1,521	16.90%	8.74% (133)	5.46% (83)	1.12% (17)	1.58% (24)	38%
F	958	17.64%	6.78% (65)	8.66% (83)	0.42% (4)	1.77% (17)	56%
G	461	18.44%	7.81% (36)	8.68% (40)	0.87% (4)	1.08% (5)	53%
H	204	21.57%	3.92% (8)	14.22% (29)	0.49% (1)	2.94% (6)	78%
I	92	14.13%	1.09% (1)	11.96% (11)	0.00% (0)	1.09% (1)	92%
J	45	8.89%	2.22% (1)	6.67% (3)	0.00% (0)	0.00% (0)	75%

8.2 Time and memory limits, test counts

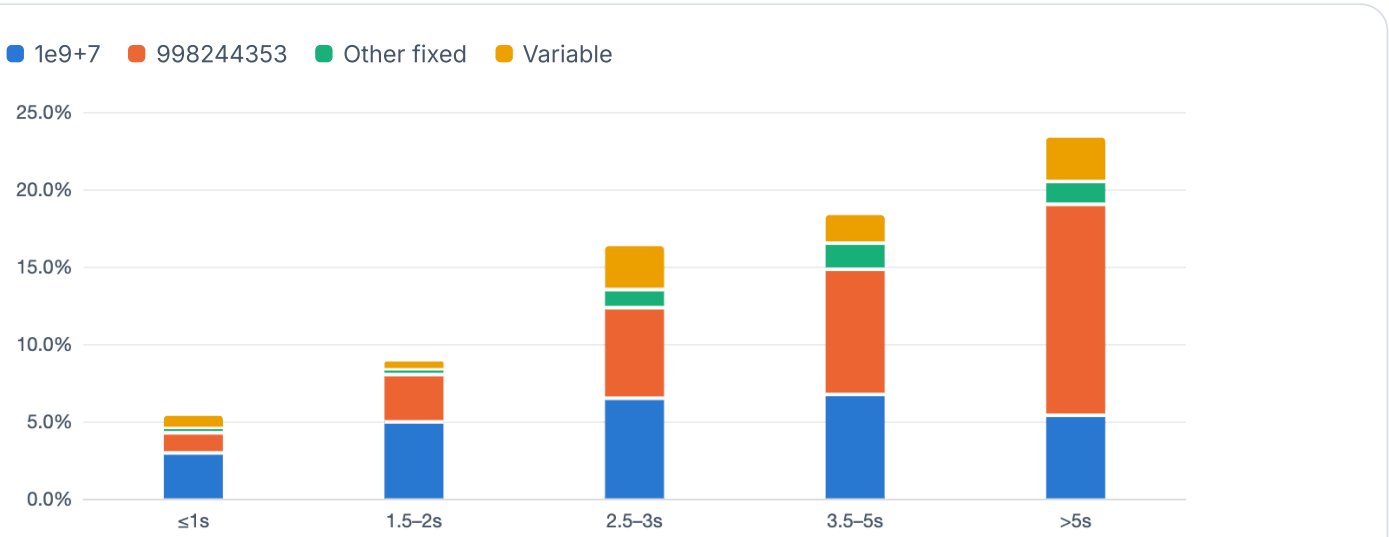


Figure 10. Modular usage by time-limit band.

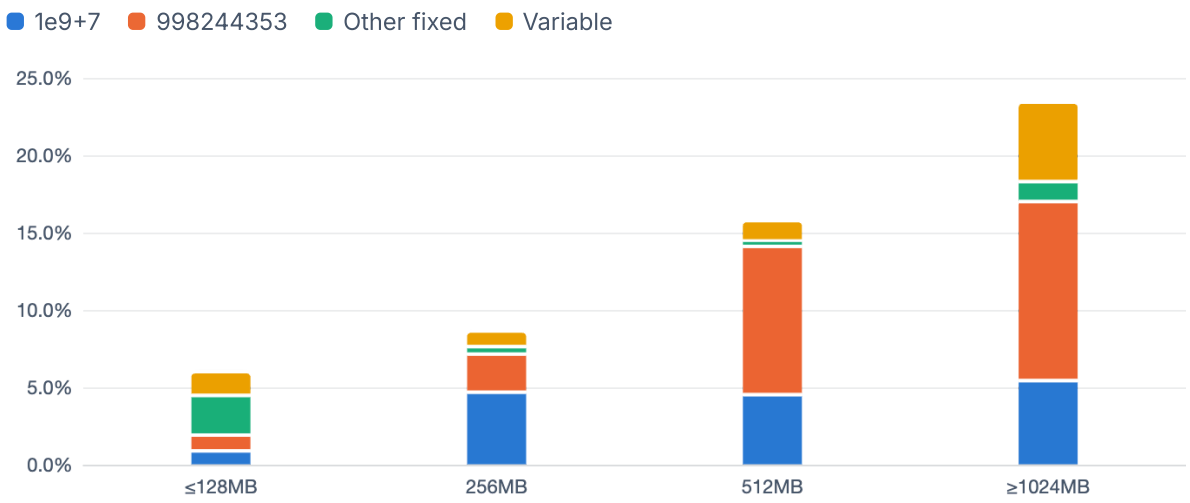


Figure 11. Modular usage by memory-limit band.

Group	Problems	Mean TL (s)	Median TL (s)	Mean ML (MB)	Median ML (MB)	Median system tests	Mean rating
1e9+7	477	2.37	2	303	256	45	2340
998244353	347	3.11	2	394	256	38	2550
Other fixed	53	3.21	3	293	256	42	2347
Variable	116	2.67	2	364	256	40	2244
None	9,047	2.03	2	294	256	40	1798

Multi-test input format: 6.9% of multi-test problems and 11.2% of single-test problems use a modulus. The number of system tests barely differs between groups (median 38–45).

8.3 Explicit vs. introduced in the solution

Nearly every modulus is explicit in the statement. Only 5 of 6,078 problems with an editorial use a modulus that appears only in the editorial (2520, 987898789, 2^64, 2020, 2^30). These are typically hashing moduli or bounds such as 2^{64}, and editorials rarely restate the hashing modulus, so this figure is a lower bound.

8.4 What the modular operations are

Statement signals are measured over all problems in the group. Editorial signals are measured over problems that have an editorial (1e9+7: 264, 998244353: 286, other: 13, variable: 73).

Operation / signal	1e9+7	998244353	Other fixed	Variable
counting	56%	58%	42%	34%
probability / expectation	10%	16%	6%	1%
modular inverse (p/q answer)	9%	15%	8%	1%
sum over objects	5%	6%	6%	5%
binomial coefficients	31%	46%	15%	21%
factorials	5%	12%	0%	4%
modular inverse	7%	8%	23%	5%
fast exponentiation	4%	4%	15%	4%
FFT / NTT	2%	11%	8%	5%
matrix exponentiation	4%	2%	0%	1%
inclusion-exclusion	5%	8%	8%	1%
CRT / Lucas	0%	0%	8%	5%
hashing	0%	2%	0%	1%

9. Trends, correlations and unexpected findings

- **998244353 overtook 1e9+7 in 2019.** Up to 2017, 1e9+7 appeared in 235 problems against 6 for 998244353. In 2018 the two were nearly even (35 vs 33). From 2019 on, 998244353 leads every year, and across 2022–2024 it took 66% of the problems that used either modulus.
- **Moduli are a hard-problem phenomenon.** Only 0.9% of problems rated below 1200 use a modulus, against 19.6% at 2400–2799 and 21.4% at 2800+. 998244353 problems average rating 2550, against 2340 for 1e9+7. Below 1600, 1e9+7 is still the default (998244353 holds only 28% of the big two at 1200–1599). From 2400 up the two are split almost evenly.
- **fft is the clearest modulus marker.** 73.7% of fft-tagged problems use a modulus, and 86% of those that use one of the big two pick 998244353. That fits $998244353 = 119 \cdot 2^{23}+1$, which supports number-theoretic transforms of length up to 2^{23}. matrices problems lean the other way: 71% of their big-two usage is 1e9+7.
- **combinatorics nearly always means a modulus** (69.0% of 704 problems, 7.0× the archive rate). probabilities follows at 32.8%, where answers are fractions reported as $P \cdot Q^{-1}$. 998244353 is the more common choice there (59% of the big two).
- **Contest series have house styles.** In Educational rounds 998244353 takes 73% of big-two usage, and in Global rounds 80%, against 21% in Div. 2 rounds, which have a longer pre-2018 history. Div. 1 (16.1%) and Div. 1 + Div. 2 (16.6%) rounds use moduli far more often than Div. 3 (3.7%) or Div. 4 (2.3%).
- **The later the problem, the more likely a modulus.** Usage rises from 1.7% of A problems to 16.9% of E and 21.6% of H. 998244353's share also grows with position, from 13% at A to 56% at F.
- **998244353 problems get more generous limits.** Their mean time limit is 3.11 s against 2.37 s for 1e9+7 and 2.03 s for non-modular problems. Mean memory is 394 MB against 303 MB and 294 MB. Modular share rises steadily with the time limit, from 5.6% at ≤1 s to 23.6% above 5 s, which is consistent with heavy convolution and DP workloads.

- **Expected-value problems prefer 998244353.** Among 998244353 problems, 16.1% ask for an expectation or probability and 15.0% require reporting a fraction as $P \cdot Q^{-1}$. For 1e9+7 the figures are 10.1% and 9.4%.

Unexpected

- Only 2 problems use both big moduli in one statement (1601F Two Sorts, 1909I Short Permutation Problem).
- The long tail has 24 distinct fixed moduli beyond the big two, each used in at most 12 problems. Examples: 1e9+9 (12), 10^6+3 (8), the dated 20240401 (April Fools 2024), 777777777, 51123987, the NTT prime 7340033, 998244853 (one digit off 998244353), and 999999893, which a statement itself flags as "the unusual modulo".
- Multi-test problems use moduli *less* often (6.9% against 11.2% for single-test problems). This is probably confounded rather than causal: the multi-test format spread after 2019, together with easy Div. 3/4 problems.
- Editorials almost never introduce a modulus the statement didn't already mention. Among 6,078 problems with an editorial in the dataset, only 5 use a modulus that appears only in the editorial (e.g. a hashing modulus).
- Overall modular usage is fairly stable over time (about 7%–13% of problems per year since 2013). The change is *which* modulus is used, not how often.

---

10. Validation

A stratified random sample of 110 problems (40 confirmed, 15 uncertain, 15 variable, 40 none, with half of the "none" sample drawn from problems whose text mentions modulo or remainder) was read and judged by hand. The sample is selected by a seeded hash, so it can be reproduced (`validate.py`). Every verdict is in validation.csv.

Predicted status	Sampled	Correct	Partial	Wrong
confirmed	40	40	0	0
variable	15	14	0	1
uncertain	15	11	4	0
none	40	40	0	0

Overall: 105/110 correct, 4 partial, 1 wrong. Precision on the 40 sampled confirmed problems was 40/40. A dedicated recall check, which searched every statement for 998244353 / 1e9+7 in any notation, found no remaining confirmed misses beyond intentional exclusions such as constraints, jokes and examples. Errors and partial cases:

- **1192B** (uncertain → partial): fixed modulus 2000 used only to decode online queries; correctly kept out of confirmed but is a real (incidental) modulus
- **1165A** (uncertain → partial): remainder 10^y modulo 10^x: modulus is an input parameter, should be variable rather than uncertain
- **1614E** (uncertain → partial): (x + lastans) mod (10^9+1) query decoding; real but incidental modulus, correctly not confirmed
- **792F** (uncertain → partial): query decoding "(ai + j) mod 10^6 + 1"; incidental modulus, correctly not confirmed
- **1237G** (variable → wrong): "(s_i + 1) mod n" is cyclic indexing, not a modulus requirement

Uncertain problems (excluded from headline counts): 792F (1e6+1), 991F (1e9+7), 1096A (998244353), 1096C (998244353), 1096D (998244353), 1165A (1e5), 1192B (2000), 1295A (998244353), 1355F (998244353), 1560F1 (998244353), 1560F2 (998244353), 1599F (1e9+7), 1614E (1e9+1), 1844C (998244353), 1933D (1e3).

11. Limitations and possible biases

- **Coverage.** 12.8% of the archive has no statement, almost all from after February 2025. 2025 figures are partial (dashed in charts), and 2026 is not covered.
- **Snapshot dependence.** Statements come from a third-party dataset. Text-rendering losses (flattened superscripts) were repaired by rule, and a statement edited on Codeforces after the snapshot would not be reflected.
- **Ratings and tags** are assigned after contests and can change. They are also missing for 227 analyzed problems, and tags reflect the solution as much as the statement.
- **Contest type** is inferred from contest names. Shared Div. 1/Div. 2 problems count once, under the contest the API lists, which biases Div. 2 counts slightly downward.
- **Correlation is not causation.** Year, difficulty, contest type and format are entangled. For example, multi-test format and 998244353 both became common after 2018.
- **Variable vs. fixed** is decided by text patterns. Problems that define the modulus in unusual words can fall into "variable" or "none" (see the validation error list).
- **Operations** such as factorials or NTT are measured from editorial keywords where an editorial exists, so they show tendencies rather than exact rates.

12. Reproducibility

```
python3 -m venv .venv && .venv/bin/pip install pyarrow   # only the crawler needs pyarrow
.venv/bin/python crawler.py      # API + dataset into SQLite (cached, resumable, incremental)
python3 detect.py                # modulus detection
python3 validate.py sample       # reproduce the validation sample; validate.py load imports
verdicts
python3 analyze.py               # aggregates + exports
python3 build.py                 # site + report + PDF
```

Scripts: source.zip. Statement text © the problem authors / Codeforces, via open-r1/codeforces (CC BY 4.0). The published data files contain metadata and short detection snippets only.