

UDC 004.8

Azamatuly Yerassyl

Grade 11 student

Supervisor: Onayev Ainabek

teacher of AP Statistics and Geometry

Republican Physics and Mathematics School

(Almaty, Kazakhstan)

WHY AI-GENERATED CODE FAILS A BENCHMARK: A FAILURE ANALYSIS OF THREE OPEN LANGUAGE MODELS ON 30 HUMANEVAL+ TASKS

Abstract: Many people now write programs by asking a language model for the code and running what it returns. Benchmarks such as HumanEval+ are meant to show how often such code actually works. I gave the first 30 tasks of HumanEval+ to three open models, Qwen3.8 27B, gpt-oss-120b and Cohere North Mini Code, in five runs, and recorded for every failed task why it failed. In the first scoring, the pass counts ranged from 4 to 29 out of 30, and Qwen passed 24 tasks at the default temperature but 29 at temperature 0. Re-checking the saved answers showed that most of these failures were not in the code's logic. Of 35 failed checks, 8 happened because the answer relied on an import line or a helper function that was in the task prompt but not in the answer, 22 because the code was wrapped in markdown fences, 2 because the answer was cut off at the token limit, and 1 was a check that failed once and passed when it was repeated. Only 2 failures, one for gpt-oss-120b and one for North Mini Code, were real logic errors. When the same 30 saved answers of Qwen were scored together with the prompt, the result changed from 24 to 30 out of 30 (exact McNemar test, $p = 0.031$). After the scoring was made the same for all runs, the models passed 29 or 30 tasks each, and no difference between models or temperatures could be seen. For small benchmark studies like this one, the way the answer is taken out of the model's reply and scored can matter more than the choice of model.

Keywords: large language models; code generation; HumanEval+; benchmark evaluation; reproducibility; failure analysis

Introduction. Writing code by talking to a language model has become common enough to get its own name. Sarkar and Drosos call it "vibe coding" and describe it as programming in which the developer mostly prompts the model, looks through the result quickly and tests whether the program works, instead of writing the code by hand [1]. This makes a simple question important: how often is the code that a model returns actually correct, and when it is not, why?

The usual way to measure this is a benchmark of programming tasks with tests. HumanEval, released by OpenAI, has 164 short Python tasks. Each task gives the model a function signature with a description and a few examples, and the code the model writes is run against hidden tests [2]. MBPP is a similar set of short Python problems [3]. Liu and co-authors later showed that the original HumanEval tests were too weak: they added many new test inputs, on average about 80 times more, and called the new version HumanEval+. With the stronger tests, code that had been counted as correct often failed, and the pass rates of many models went down noticeably [4].

A benchmark score depends not only on the model but also on how the test is run. Biderman and co-authors, who maintain a widely used evaluation tool, describe how small details of the setup can change results and make studies hard to compare [5]. Sclar and co-authors found that changing only the formatting of a prompt changed the accuracy of one model by up to 76 percentage points [6]. There is also randomness in the model itself. Ouyang and co-authors asked ChatGPT to solve the same coding problems several times and often got code that behaved differently from request to request; setting the temperature to 0 reduced this but did not remove it [7]. Atil and co-authors found the same for settings that are expected to be deterministic [8]. Because of this, several authors recommend repeating runs and reporting uncertainty, not just one number [9, 10].

Studies like these usually use hundreds of tasks and many repeated runs. A student, or a developer checking a model before using it, usually works on a much smaller scale: a free model, a notebook, a few dozen tasks, one run. I wanted to see what such a small test shows. My original plan was to give the same tasks to several free models and find out where and why their code fails. During the project, the answer to "why" turned out to be different from what I expected, and this paper is about that. The research questions are:

- How many of the first 30 HumanEval+ tasks do three open models pass, and for what reasons does their code fail?
- How much does the way the answer is taken from the reply and scored change the pass count for the same answers?
- After the scoring is made the same for all runs, is there a difference between the models, or between temperature 0 and the default temperature?

Data and methods.

Tasks and tests. I used the first 30 tasks of HumanEval+ (HumanEval/0 to HumanEval/29), version 0.1.10, with the EvalPlus package version 0.3.1 [4]. Each task has a small set of original tests and a larger set of added tests. Among the 30 tasks, the number of tests per task ranges from 69 to 1,006, and 20,613 in total. A task counts as passed only if the code passes all original and all added tests. The tests are checked by running the task's reference solution on the same inputs and comparing the outputs.

Two details of the task prompts turned out to matter. In 21 of the 30 tasks the prompt starts with an import line such as "from typing import List", which is needed because the function signature uses the type List. In one task, HumanEval/10 (make_palindrome), the prompt contains a second, finished function, is_palindrome, which the model is expected to use.

Models. I used three models that were free or had a free usage tier when I ran them (Table 1). Qwen3.8 27B is an open-weight dense model from the Qwen team that also works with images [11]. gpt-oss-120b is an open-weight model from OpenAI with 117 billion parameters in total, of which 5.1 billion are active for each token [12]. North Mini Code is Cohere's model for coding; it has 30 billion parameters in total and 3 billion active [13]. Qwen and gpt-oss were called through the Groq API, and North Mini Code through OpenRouter. I first tried Gemini as a third model, but the free limit ran out after about ten tasks, so it is not included.

Prompt. Each model received the task prompt exactly as it is in the dataset, followed by one sentence: "Return only the Python code for the solution. Do not include explanations or markdown." Apart from the repeats described below, each task was sent once in each run.

Runs. There were five runs (Table 1). Runs A and B, on 21 September 2026, did not set the temperature, so the provider's default was used; for Groq this is 1 [14]. Runs C, D and E, on 24 September 2026, used temperature 0. The runs also differed in how the code was taken out

of the reply, because I changed the notebook between runs. In runs A, B and C, if the reply contained a block marked with markdown fences (three backticks), only the code inside the first block was saved. In runs D and E the whole reply was saved. North Mini Code spends part of its token limit on reasoning, and at a limit of 2,000 tokens it returned an empty answer for one task and an answer that stopped in the middle for two others. These three tasks (HumanEval/1, 10 and 21) were sent again with a limit of 5,000 tokens. In run C, one request (HumanEval/20) hit the provider's rate limit and was sent again.

Table 1. The five runs. All runs used the same 30 tasks and the same prompt.

Run	Model (provider)	Temp.	Token limit	What was saved from the reply
A	Qwen3.8 27B (Groq)	1 (default)	1,000	code inside the first fenced block, or the whole reply
B	gpt-oss-120b (Groq)	1 (default)	1,000	code inside the first fenced block, or the whole reply
C	Qwen3.8 27B (Groq)	0	1,000	code inside the first fenced block, or the whole reply
D	gpt-oss-120b (Groq)	0	1,000	the whole reply
E	North Mini Code (OpenRouter)	0	2,000; 5,000 for three tasks	the whole reply; fences removed afterwards

Two ways of scoring. EvalPlus accepts an answer in two forms. A "solution" is a complete program and is run as it is. A "completion" is only the continuation of the prompt, so the prompt is placed in front of it before running [4]. The original HumanEval also runs the prompt together with the model's completion [2]. In my notebook, runs A, B and C were first scored as complete programs ("answer alone"). Runs D and E, and later a second check of run C, were scored with the task prompt placed in front of the answer ("prompt + answer"). When the prompt is in front, its import line and helper functions are available to the answer. If the answer defines the same function again, the new definition replaces the one from the prompt, so the model's own code is what gets tested.

The saved answers of runs A and B were kept in the notebook output, so I scored all 60 of them again in both ways with the same EvalPlus version. The re-scoring reproduced the notebook results exactly (24 and 29 passed tasks for the answer alone). For runs C, D and E, the answer files were stored only inside the Colab session, which does not keep files after it closes, so for these runs I report the scores that were printed in the notebook.

Failure causes. For every failed task I looked at the saved code and at the error it produced, and put it into one of five groups: a name that was defined only in the prompt (an import or a helper function), markdown fences in the saved code, an answer cut off at the token limit, a check that failed once and passed when repeated with the same code, and wrong logic. When a failure disappeared after one fix but the task still failed for another reason, it was counted under the later reason, so each failed check is counted once.

Statistics. For pass rates I give 95% Wilson intervals [15]. To compare the same 30 answers scored in two ways, I used the exact McNemar test, which looks only at the tasks where the two results disagree [16]. To compare two different runs, I used Fisher's exact test.

Results

Scores as first recorded. Figure 1 shows the pass counts in the first scoring and after the scoring was corrected. As first recorded, the results looked very different from each other. Qwen

passed 24 of 30 tasks at the default temperature (run A) and 29 at temperature 0 (run C). gpt-oss-120b passed 29 at the default temperature and 29 at temperature 0. North Mini Code passed only 4 tasks. If these numbers were taken at face value, one would conclude that North Mini Code is much weaker than the other two, and that Qwen became much better at temperature 0.

Even so, with only 30 tasks, a gap like 24 against 29 is not strong evidence. The 95% interval for 24 of 30 is 62.7% to 90.5%, and for 29 of 30 it is 83.3% to 99.4%. Fisher's exact test for 24 against 29 gives $p = 0.10$.

The same answers, scored in two ways. Table 2 shows what happened when the saved answers were scored again. The 30 answers of run A passed 24 tasks when each answer was run alone and 30 tasks when the prompt was placed in front. All six changes went in the same direction, and none of the answers that passed alone failed with the prompt. For these six discordant tasks the exact McNemar test gives $p = 0.031$. For run B the count went from 29 to 30, and for run C from 29 to 30. Nothing in the code was changed; only the scoring was.

Table 2. Number of tasks passed out of 30 in each way of scoring.

Run	Answer alone	Prompt + answer, as saved	After all corrections
A. Qwen, default	24	30	30
B. gpt-oss, default	29	30	30
C. Qwen, temp. 0	29	30	30
D. gpt-oss, temp. 0	not scored	29	29
E. North Mini Code, temp. 0	not scored	4	29

For run E, "after all corrections" means after removing the markdown fences, re-sending two answers that had been cut off, and repeating the check for HumanEval/15.

In run E the first check was already done with the prompt in front, but 26 of the 30 saved answers were wrapped in markdown fences, even though the prompt asked for no markdown. The reply for HumanEval/10 also had fences, but I had removed them by hand when I saved that answer again. Python cannot run a file that starts with three backticks, so all 26 failed without running a single test. After the fences were removed, 26 tasks passed. Two of the remaining failures, HumanEval/1 and HumanEval/21, were answers that had stopped in the middle at the 2,000-token limit; with a limit of 5,000 tokens both passed. One more, HumanEval/15, failed one of its 137 tests on the first check and passed all 137 when the same code was run again. The code did not change, so the most likely reason is the time limit of the test.

Where the failures came from. Across the five runs there were 35 failed checks. Figure 2 shows what caused them, and Table 3 lists the tasks.

Eight failures were caused by a name that the answer used but did not define. In six cases, five in run A and one in run B, the answer used the type List in the function signature but did not include the line "from typing import List". Python evaluates the signature when it defines the function, so the program stopped with "NameError: name 'List' is not defined" before any test was run. This is why these tasks show zero passed tests and not a partly working function. In the other two cases, runs A and C, the answer to HumanEval/10 called is_palindrome, the helper function that is given in the prompt. In a sense the model did what the task asked: it used the function it was given.

The other failures were 22 answers with markdown fences in run E, two answers cut off at the token limit and one check that failed once and passed when repeated. Only two failures were caused by wrong logic.

Table 3. All failed checks in the five runs.

Run	Task	What went wrong	Fixed by
A	5, 9, 17, 22, 26	"from typing import List" missing, NameError	putting the prompt in front
A, C	10	helper is_palindrome is in the prompt, not in the answer	putting the prompt in front
B	28	"from typing import List" missing, NameError	putting the prompt in front
D	24	wrong logic in largest_divisor	not fixed
E	22 tasks	code wrapped in markdown fences	removing the fences
E	1, 21	answer cut off at 2,000 tokens	new request with 5,000 tokens
E	15	failed 1 of 137 tests on the first check, most likely the time limit	running the check again
E	5	fences, then wrong logic in intersperse	not fixed

The two real errors. In run D, gpt-oss-120b solved HumanEval/24, which asks for the largest divisor of n that is smaller than n , by starting at the square root of n and going down. The first divisor found this way is the largest divisor that is not bigger than the square root, and the function returned n divided by it. For 100 this gives 10, while the correct answer is 50. The code's own comment says it looks for "the smallest factor", but the loop does not do that. The function passed 4 of 5 original tests, so the error shows up only for some inputs. In run B, at the default temperature, the same model solved this task correctly.

In run E, North Mini Code solved HumanEval/5, which asks to put a delimiter between every two neighbouring numbers of a list. Its code paired every number with the delimiter, so the delimiter also ended up after the last number: for [1, 2, 3] and 4 it gives [1, 4, 2, 4, 3, 4] instead of [1, 4, 2, 4, 3]. It passed 7 of 156 tests, only those where the list was empty.

Models and temperature after the correction. With the same scoring for every run, the counts were 30, 30 and 30 for runs A, B and C, and 29 and 29 for runs D and E. The differences between models are at most one task, and Fisher's exact test for 30 against 29 gives $p = 1.0$. For Qwen, temperature 0 and the default temperature gave the same result, 30 of 30. For gpt-oss-120b, the default run passed 30 and the temperature 0 run passed 29, and the one failure was the logic error described above.

Temperature did matter in one respect. When the answers were scored alone, Qwen's default-temperature answers left out the typing import in 5 of the 21 tasks that needed it, while none of the failures of its temperature 0 answers were caused by a missing import. So the difference between 24 and 29 came from whether the model copied the import line into its answer, not from whether it solved the task. With one run for each setting I cannot tell whether this was an effect of temperature or chance.

Discussion. The main result is that, in this small study, most of the failures were in the way the code was handled, not in the code. Of 35 failed checks, 30 disappeared when the answers were taken out of the reply and scored in the same way for every run, and 3 more when

the two cut-off answers were requested again and the unstable check was repeated. What was left, two logic errors in 150 answers, is too little to rank the three models.

This matters for how the first numbers would be read. The gap between Qwen's 24 and 29 looked like an effect of temperature, and North Mini Code's 4 of 30 looked like a weak model. In both cases the difference came from the pipeline and not from the models. This also explains a mistake we made during the project. In the results table for run C, none of the six tasks that Qwen had failed in run A failed again, and we took this as a sign that the model gives different results from run to run. Only after scoring the saved answers of run A again did it become clear that the scoring had also changed between the runs, and that five of the six failures had the same cause, a missing import line. Checking the saved answers, and not only the pass counts, would have shown this at once.

The results fit with what larger studies have found. Biderman and co-authors warn that evaluation details, which are often not reported, can change results [5], and Sclar and co-authors showed how much formatting alone can move a score [6]. Here the "formatting" was on the side of the answer: an import line, a helper function and markdown fences. The EvalPlus tool itself avoids some of these problems. When it generates answers from chat models, it asks for "a self-contained Python script", which would tell the model to include the import line [4]. My prompt asked only to return the code, so an answer without the import line was a reasonable reading of the instruction.

North Mini Code ignored the instruction not to use markdown in 27 of its 30 replies. gpt-oss-120b followed it in run D, where the whole reply was saved. For Qwen it is not possible to say, because in runs A and C the fences were removed when the code was saved. A pipeline that takes the code out of a fenced block, as in runs A, B and C, handles this without problems. A pipeline that saves the whole reply, as in runs D and E, depends on each model obeying the instruction, and the models did not all behave the same way.

A second point is about the token limit. North Mini Code uses part of its token budget for reasoning, and at 2,000 tokens it sometimes ran out before writing the answer. Such cases look like failures of the model, although the cause is a setting that suits one model better than another. Using the same limit for all models does not make a comparison fair when the models spend tokens differently.

Finally, the results agree with the advice to report uncertainty [9, 10]. With 30 tasks, even a result of 30 out of 30 has a 95% interval from 88.6% to 100%. Two models that differ by one task cannot be told apart with this number of tasks, and differences between runs such as those reported by Ouyang and co-authors [7] could not be measured with one run per setting.

Conclusion, limitations and recommendations. I tested three open language models on the first 30 tasks of HumanEval+ and looked at every failure. In the first scoring, the models seemed to differ a lot, and one model seemed to depend strongly on temperature. After the answers were scored in the same way, all three passed 29 or 30 tasks. Only 2 of 35 failed checks were caused by wrong logic; the rest came from how the code was taken out of the reply, how it was run, and the token limit. For a small benchmark, these choices can change the result more than the choice of model does.

Limitations

Few tasks. Thirty tasks, taken from the start of HumanEval+ and not at random, are too few to show small differences between models.

One run per setting. Each model was run once at each temperature, so run-to-run variation could not be measured, and the effect of temperature cannot be separated from chance.

Lost answer files. The answers of runs C, D and E were not saved outside the Colab session. For these runs I rely on the scores printed in the notebook and could not re-score them myself.

Settings changed between runs. Besides temperature, the way the code was saved and scored changed between the runs. This is the main subject of the paper, but it also means the runs are not a clean experiment on temperature.

Model versions. The models were used through Groq and OpenRouter in September 2026. Providers can update a model behind the same name, so the results may not repeat exactly later.

Possible training on the benchmark. HumanEval has been public since 2021 and may be in the training data of these models, which could make the tasks easier for them than new tasks would be.

Recommendations. For anyone who runs a small benchmark of this kind: keep every answer file, not only the scores; decide in advance whether answers are scored as complete programs or together with the prompt, and use the same choice for every model and every run; take the code out of the reply in the same way for all models; check that no answer was cut off at the token limit; and look at the error of every failed task before counting it as a failure of the model.

For future work, the same tasks should be run several times for each model and temperature, which would show how much the pass count changes between runs. A larger and random set of tasks from HumanEval+ or MBPP+ would make the comparison between models more informative.

Data availability. The HumanEval+ dataset and the EvalPlus package are publicly available [4]. The table of results for every task and run, the 60 saved answers of runs A and B, and the script used to score them again are available from the author on request.

AI-use disclosure. The author wrote the Colab notebook with help from ChatGPT, ran the models, and recorded the results. The re-scoring of the saved answers and the figures were done in Python with the help of an AI assistant, and generative AI was also used for language editing and structural revision of the text. No data, results or citations were generated by AI: every pass or fail result comes from running the models' code against the EvalPlus tests, all numbers were checked against the notebook output and the re-scoring, and all cited sources were checked against the originals. The author has reviewed the final manuscript and takes full responsibility for it.

Acknowledgements. I thank my supervisor, Ainabek Onayev, for supervising this project.

Conflicts of interest. The author declares no conflicts of interest.

References:

- [1] Sarkar, A., & Drosos, I. (2025). Vibe coding: Programming through conversation with artificial intelligence. arXiv preprint arXiv:2506.23253. <https://doi.org/10.48550/arXiv.2506.23253>
- [2] Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. de O., Kaplan, J., ... Zaremba, W. (2021). Evaluating large language models trained on code. arXiv preprint arXiv:2107.03374. <https://doi.org/10.48550/arXiv.2107.03374>

- [3] Austin, J., Odena, A., Nye, M., Bosma, M., Michalewski, H., Dohan, D., ... Sutton, C. (2021). Program synthesis with large language models. arXiv preprint arXiv:2108.07732. <https://doi.org/10.48550/arXiv.2108.07732>
- [4] Liu, J., Xia, C. S., Wang, Y., & Zhang, L. (2023). Is your code generated by ChatGPT really correct? Rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)* (pp. 21558-21572). <https://doi.org/10.52202/075280-0943>
- [5] Biderman, S., Schoelkopf, H., Sutawika, L., Gao, L., Tow, J., Abbasi, B., ... Zou, A. (2024). Lessons from the trenches on reproducible evaluation of language models. arXiv preprint arXiv:2405.14782. <https://doi.org/10.48550/arXiv.2405.14782>
- [6] Sclar, M., Choi, Y., Tsvetkov, Y., & Suhr, A. (2023). Quantifying language models' sensitivity to spurious features in prompt design or: How I learned to start worrying about prompt formatting. arXiv preprint arXiv:2310.11324. <https://doi.org/10.48550/arXiv.2310.11324>
- [7] Ouyang, S., Zhang, J. M., Harman, M., & Wang, M. (2025). An empirical study of the non-determinism of ChatGPT in code generation. *ACM Transactions on Software Engineering and Methodology*, 34(2), 1-28. <https://doi.org/10.1145/3697010>
- [8] Atil, B., Aykent, S., Chittams, A., Fu, L., Passonneau, R. J., Radcliffe, E., ... Baldwin, B. (2024). Non-determinism of "deterministic" LLM settings. arXiv preprint arXiv:2408.04667. <https://doi.org/10.48550/arXiv.2408.04667>
- [9] Miller, E. (2024). Adding error bars to evals: A statistical approach to language model evaluations. arXiv preprint arXiv:2411.00640. <https://doi.org/10.48550/arXiv.2411.00640>
- [10] Blackwell, R. E., Barry, J., & Cohn, A. G. (2024). Towards reproducible LLM evaluation: Quantifying uncertainty in LLM benchmark scores. arXiv preprint arXiv:2410.03492. <https://doi.org/10.48550/arXiv.2410.03492>
- [11] OpenRouter. (n.d.). Qwen: Qwen3.8 27B [model page]. Retrieved September 26, 2026, from <https://openrouter.ai/qwen/qwen3.8-27b>
- [12] OpenAI. (2025). gpt-oss-120b & gpt-oss-20b model card. arXiv preprint arXiv:2508.10925. <https://doi.org/10.48550/arXiv.2508.10925>
- [13] OpenRouter. (n.d.). Cohere: North Mini Code [model page]. Retrieved September 26, 2026, from <https://openrouter.ai/cohere/north-mini-code>
- [14] Groq. (n.d.). API reference: Chat completions. Retrieved September 26, 2026, from <https://console.groq.com/docs/api-reference>
- [15] Wilson, E. B. (1927). Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158), 209-212. <https://doi.org/10.1080/01621459.1927.10502953>
- [16] McNemar, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2), 153-157. <https://doi.org/10.1007/BF02295996>

Figures

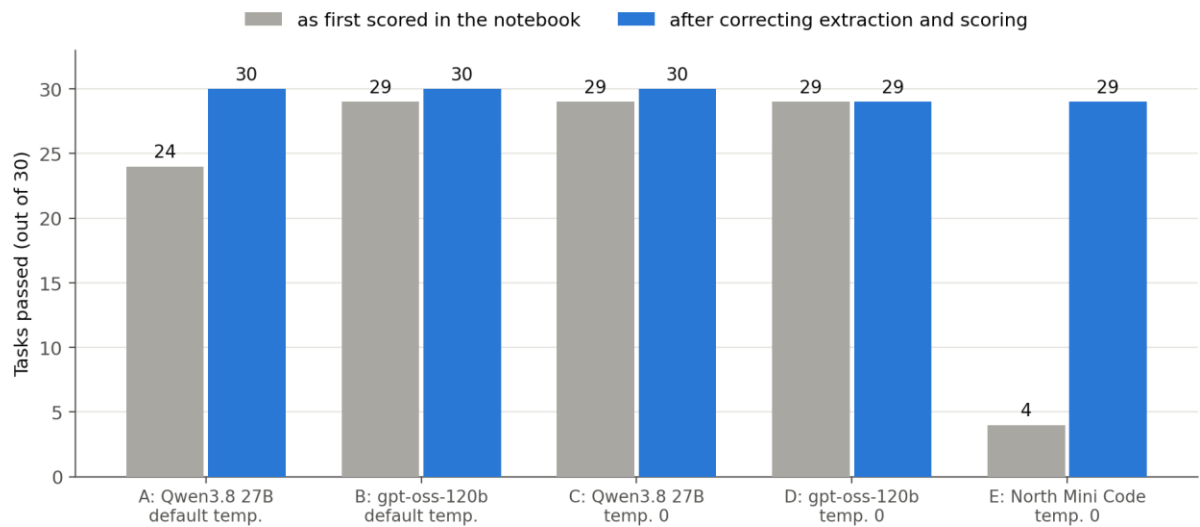


Figure 1. Tasks passed out of 30 in each run, as first scored in the notebook and after the code extraction and scoring were made the same for all runs.

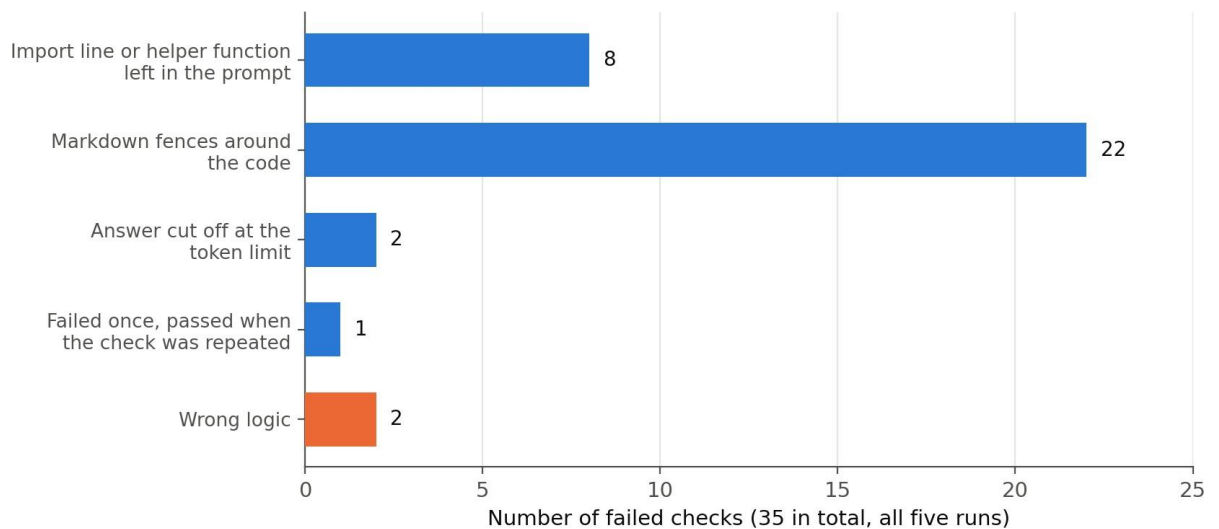


Figure 2. The 35 failed checks in all five runs, grouped by cause. Each check is counted once, under the reason it finally failed for.

МАЗМҰНЫ – СОДЕРЖАНИЕ – CONTENT

ӘЛЕУМЕТТІК-ЭКОНОМИКАЛЫҚ ЖӘНЕ ҚҰҚЫҚ ҒЫЛЫМДАРЫ - СОЦИАЛЬНО-ЭКОНОМИЧЕСКИЕ И ЮРИДИЧЕСКИЕ НАУКИ - SOCIO-ECONOMIC AND LEGAL SCIENCES

Төлегенов Тимур Сейфуллаулы

**ДИНАМИКА ТУРИСТСКИХ ПОТОКОВ И ЭФФЕКТИВНОСТЬ ИНФРАСТРУКТУРЫ
ТУРКЕСТАНА: ОТ ЭКСТЕНСИВНОГО РОСТА К СМАРТ-УПРАВЛЕНИЮ 10**

Alexander Gorevoy

**WHICH INEQUALITY STATISTICS SURVIVE AN ADMINISTRATIVE BOUNDARY CHANGE?
EVIDENCE FROM KAZAKHSTAN, 2010–2023 14**

Rakhimov Djalil

**DOES THE SHIFT TO CASHLESS PAYMENTS GO TOGETHER WITH FASTER GROWTH IN
HOUSEHOLD CREDIT? EVIDENCE FROM THE REGIONS OF KAZAKHSTAN, 2023-2025 26**

Kazhgerey Asan

**HOW MANY FOOD BASKETS DOES AN AVERAGE WAGE BUY? WAGES AND FOOD PRICES IN
THE 20 REGIONS OF KAZAKHSTAN, 2024-2026 37**

Tagimova Alua

**WHO WAS LEFT BEHIND? EDUCATION AND FINANCIAL INCLUSION IN KAZAKHSTAN AND
THREE CENTRAL ASIAN NEIGHBOURS, 2021-2024 44**

Wei Wei, Mynzhanova Gulzhakan, Niyetalina Gaukhar

**DIGITAL–PHYSICAL INTEGRATION AND ECONOMIC PERFORMANCE IN INDUSTRIAL
INTEGRATED ENERGY SYSTEMS: A SCENARIO-BASED ANALYSIS OF SMART GRID
CONTROL, BATTERY ENERGY STORAGE, AND DIGITAL TWINS 51**

Xu Fumin, Wei Wei, Ruziyeva Elvira, Moldabekova A.

**FROM CULTURAL RESOURCES TO GLOBAL CONSUMER VALUE: THE BRIDGE–IMC MODEL
AND THE INTERNATIONALIZATION OF BLACK MYTH: WUKONG UNDER THE BELT AND
ROAD INITIATIVE 64**

Ibraimkhan Alikhan

**WHICH CRIMES CAN A 14-YEAR-OLD BE TRIED FOR? THE LIST IN ARTICLE 15 OF
KAZAKHSTAN'S CRIMINAL CODE COMPARED WITH 30 OTHER JURISDICTIONS 80**

Yerboluly Madi

**DID IMPORTED CARS REALLY GET CHEAPER? CHINESE CARS AND THE AVERAGE
CUSTOMS VALUE OF PASSENGER CARS IMPORTED INTO KAZAKHSTAN, 2015-2025 91**

Салимбаев Ербол Аманжолович

**АНАЛИЗ ИНВЕСТИЦИОННОЙ АКТИВНОСТИ КАК ИНДИКАТОР ИЗМЕНЕНИЙ
ПОТЕНЦИАЛЬНОГО СПРОСА НА КОНСАЛТИНГОВЫЕ УСЛУГИ 99**

Rambergenov Ramil

**REGIONAL NEW-FIRM FORMATION IN UZBEKISTAN AFTER THE 2017 REFORMS:
PERSISTENCE IN SCALE, CONVERGENCE IN INTENSITY, AND RECOMPOSITION TOWARD
TRADE 107**

Abdesh Dinara

**FROM A WOMEN'S LEAD TO A MEN'S LEAD: THE GENDER GAP IN DIGITAL PAYMENTS IN
KAZAKHSTAN AND THREE CENTRAL ASIAN NEIGHBOURS, 2021-2024 116**

Amangeldi Aidana

**DID SMALL BUSINESS IN KAZAKHSTAN REALLY SHRINK IN 2026? THE NEW TAX CODE,
REGIONAL TAX RATES AND THE FALL IN THE NUMBER OF INDIVIDUAL ENTREPRENEURS 124**

Cheng Che	
BUILD, BUY, OR PARTNER? ENTRY MODE SELECTION AND RISK GOVERNANCE FOR WIND POWER INVESTMENT IN KAZAKHSTAN	133
Wang Liang	
FROM CAPACITY EXPANSION TO SYSTEM VALUE: COORDINATED DEVELOPMENT OF RENEWABLE ENERGY, GRID FLEXIBILITY AND ENERGY STORAGE IN KAZAKHSTAN	146
Chen Jun	
CONSTRUCTION-PERIOD CREDIT RISK MANAGEMENT FOR LARGE INFRASTRUCTURE PROJECT FINANCE IN KAZAKHSTAN	159
Alniyaz Danial	
HOW MUCH FUEL DOES A HYBRID REALLY SAVE? A SAME-MODEL COMPARISON OF HYBRID AND GASOLINE CARS IN US FUEL ECONOMY	173

ТЕХНИКАЛЫҚ ҒЫЛЫМДАР - ТЕХНИЧЕСКИЕ НАУКИ - TECHNICAL SCIENCE

Nurmakhan Almira	
ADVANTAGES OF FIBER OPTIC COMMUNICATION OVER TRADITIONAL COPPER CABLES	180
Kozhabai Bereke	
SECURITY CHALLENGES IN IOT-BASED TELECOMMUNICATION NETWORKS	188
Azamatuly Yerassyl	
WHY AI-GENERATED CODE FAILS A BENCHMARK: A FAILURE ANALYSIS OF THREE OPEN LANGUAGE MODELS ON 30 HUMANEVAL+ TASKS	197
Nurzhan Sanzhar	
A FEW SHELLS, A FEW OPERATORS: WHERE LOW EARTH ORBIT IS CROWDED IN AN ALTITUDE-INCLINATION MAP OF 16,620 ACTIVE SATELLITES	206
Askarbek Nurislam	
CAN THE WEATHER TELL WHICH DAYS IN ASTANA WILL BE POLLUTED? CLASSIFYING PM2.5 EXCEEDANCE DAYS WITH MACHINE LEARNING AND CHECKING THE DATA AGAINST A GROUND MONITOR	212
Iliyas Arman	
SHOOTING ON THE MOVE: HOW MUCH AIM A ROBOT LOSES BY IGNORING ITS OWN VELOCITY, AND WHETHER A VIRTUAL-TARGET CORRECTION RECOVERS IT	223
Ануар Әлімжан Бахытжанұлы, Сексенбаева Айша Курманбековна	
СРАВНИТЕЛЬНЫЙ АНАЛИЗ СТАНДАРТОВ И МЕТОДИК ОБУЧЕНИЯ КИБЕРБЕЗОПАСНОСТИ: NICE FRAMEWORK, ECSF И ПРАКТИКО- ОРИЕНТИРОВАННЫЕ ТЕХНОЛОГИИ	229
Mukhamedzhan Yerganat	
GROWTH OF GITHUB DEVELOPER COMMUNITIES IN KAZAKHSTAN, UZBEKISTAN AND KYRGYZSTAN, 2020-2026: EVIDENCE FROM THE GITHUB INNOVATION GRAPH	239

ПӘНДЕР АРАЛЫҚ ҒЫЛЫМДАР – МЕЖДИСЦИПЛИНАРНЫЕ НАУКИ – INTERDISCIPLINARY SCIENCES

Albiyev Asser	
HOW MUCH OF AN ARTICLE COUNT IS HUMAN WORK? DECOMPOSING THE CREATION OF FIVE WIKIPEDIAS, 2010–2025	251
Amangeldy Nurbolat	
HOW MUCH OF KAZAKHSTAN'S URBAN BUILT STOCK PREDATES ITS SEISMIC CODES, AND WHERE DOES THAT STOCK SIT RELATIVE TO HAZARD?	261

Torekhan Sabrina

WHERE ALMATY WALKS ITS OWN WAY: INFORMAL SHORTCUTS IN OPENSTREETMAP AND THE DETOURS THEY AVOID	267
---	------------

ӘЛЕУМЕТТІК ЖӘНЕ ГУМАНИТАРЛЫҚ ҒЫЛЫМДАР - СОЦИАЛЬНО-ГУМАНИТАРНЫЕ НАУКИ - SOCIAL SCIENCES AND HUMANITIES

Мұқатаева Айзат Амантайқызы, Стамбекова Жанат Байғазықызы, Слямханова Гүлім Болатқызы, Қызырбек Аяулым Ерболқызы

ЖАСАНДЫ ИНТЕЛЛЕКТ ДӘУІРІНДЕГІ ҒЫЛЫМИ ТАНЫМ: ПОППЕР, КУН ЖӘНЕ ЛАКАТОС ИДЕЯЛАРЫ ТҮРҒЫСЫНАН ТАЛДАУ	272
--	------------

Мұқатаева Айзат Амантайқызы, Сапарханова Мадина Төлегенқызы, Окенова Аружан Муратхановна, Тұрсынғазыева Ақерке Ерболатқызы

ШӘКӘРІМ ФИЛОСОФИЯСЫНДАҒЫ ЕРКІНДІК ЖӘНЕ МОРАЛЬДЫҚ ТАҢДАУ МӘСЕЛЕСІ	282
---	------------

Komekova Saida

DO UNHAPPY REVIEWERS MIX IN MORE RUSSIAN? STAR RATINGS AND KAZAKH-RUSSIAN CODE-SWITCHING IN 175,002 ONLINE REVIEWS	291
---	------------

Zhumagali Yerbol

DOES A BIGGER MODEL CLOSE THE GAP? KAZAKH-LANGUAGE ACCURACY ON KAZAKHSTAN-SPECIFIC VERSUS UNIVERSAL SCHOOL SUBJECTS IN SMALL OPEN-WEIGHT LANGUAGE MODELS	304
---	------------

Abdrakhmanova Madina

DOES A MEDAL BUY ATTENTION? WIKIPEDIA VIEWS OF KAZAKHSTAN'S ATHLETES BEFORE AND AFTER THE PARIS 2024 OLYMPIC	308
---	------------

ПЕДАГОГИКАЛЫҚ ЖӘНЕ ПСИХОЛОГИЯЛЫҚ ҒЫЛЫМДАР – ПЕДАГОГИЧЕСКИЕ И ПСИХОЛОГИЧЕСКИЕ НАУКИ – PEDAGOGICAL AND PSYCHOLOGICAL SCIENCES

Kaldarbek Mergen

FAMILY OR SCHOOL? SOCIOECONOMIC COMPOSITION AND MATHEMATICS ACHIEVEMENT IN KAZAKHSTAN, PISA 2022 AND 2025	316
--	------------

Zhenis Adilzhan

WHICH OLYMPIAD PROBLEMS DO STUDENTS TAKE TO THE BOARD? CHOICES AND DEFENCES IN THE 8TH-GRADE LEAGUE OF KARAGANDY MATHBATTLES 2026	323
--	------------

Базарбаева Айнура Бектуровна

ДЕТСКИЙ ФОЛЬКЛОР И ПОДВИЖНЫЕ ИГРЫ КАК ИНСТРУМЕНТЫ ФОРМИРОВАНИЯ ОСНОВ ПРАВОВОЙ КУЛЬТУРЫ У ДЕТЕЙ СТАРШЕГО ДОШКОЛЬНОГО ВОЗРАСТА	330
---	------------

БИОЛОГИЯ ҒЫЛЫМДАР - БИОЛОГИЧЕСКИЕ НАУКИ - BIOLOGICAL SCIENCES

Bazhenova Diana

GROWTH DYNAMICS OF STERLET (<i>Acipenser ruthenus</i> L.) REARED IN A RECIRCULATING AQUACULTURE SYSTEM	333
---	------------

ТУРИСТИЧЕСКИЕ И СПОРТИВНЫЕ НАУКИ - ТУРИСТІК ЖӘНЕ СПОРТ ҒЫЛЫМДАРЫ - TOURISM AND SPORTS SCIENCES

Төлеп Жазира Маратқызы

ГАСТРОНОМИЯЛЫҚ ФЕСТИВАЛЬДЕРДІҢ ТУРИСТІК ДЕСТИНАЦИЯ ТАРТЫМДЫЛЫҒЫН ҚАЛЫПТАСТЫРУДАҒЫ РӨЛІ МЕН ДАМУ ҮРДІСТЕРІ	340
--	------------

ЕСТЕСТВЕННЫЕ НАУКИ - ЖАРАТЫЛЫСТАНУ ҒЫЛЫМДАРЫ - NATURAL SCIENCES

Нургельдина Асель Ермаковна, Серикбаева Жанерке Алимжановна

ШЕКАРАСЫЗ ФИЗИКА: ӘР ЕЛДЕГІ ФИЗИКАЛЫҚ БІЛІМДІ САЛЫСТЫРУ 347