# Generative AI, Natural Language Processing, Assignment 2 Oleh Prostakov

## 1 Introduction

Advanced Math Tutor (AMT) is a purely agentic system designed to generate and solve college-level math problems. It consists of 3 independent agents, and relies on LangGraph for orchestration, and WolframAlpha for symbolic compute. The system relies on inputs for the **domain** (i.e. calculus), **topic** (i.e. integration with substitution) and **complexity** (easy to hard) to generate problems. The system operates using English, but, as requested by the description, both the problem and the solution can be presented in Ukrainian (or any other language).

Math tasks generated by AMT can be divided into 2 categories:

- Equations - small (mostly, one-liners) and relatively simple tasks, aimed at verifying purely mechanical skills of computing different concepts within the domain.
- Problems - large complex multidisciplinary tasks, which require multi-step reasoning to even arrive at the underlying equations. They **cannot** be solved by simply pasting them into Wolfram, and are aimed at verifying the actual understanding of the domain by the student.

And, consequently, they can be solved with 2 different approaches:

- Quickly, with AMT producing only an answer in either numeric or symbolic version. This approach is applicable to equations **only**.
- In a step-by-step manner, with a larger problem split into byte-sized pieces, with AMT providing a long and elaborate solution, explaining each step in detail. This approach can be applied to both problems and equations, but it's orders of magnitude more expensive in terms of both time and tokens spent

The evolution of the system can be roughly summarized into 3 main sections. First, there was an Equation Generator, capable of generating and solving equations. It was good enough, but not nearly **Advanced**, as stated by the title of the assignment. So, I upgraded it to Problem Generator. At this stage, the system could generate tasks with a level of complexity comparable to what we've seen on various math courses at UCU. Unfortunately, it was no longer capable of generating 'simple' equations. Restoring this ability without compromising problems was the idea of the final enhancement.

Although RAG is mentioned in the description as an option, I decided against it mostly because such a system would require preparing a RAG corpus, which no one really likes to do.

### 1.a Model

I use `gpt-5.2` throughout the system and for all the experiments mentioned in the report. I settled up on that model because it scores first for the math tasks according to HuggingFace's LMArena, so it was kind of an obvious pick.

### 1.b Artifacts

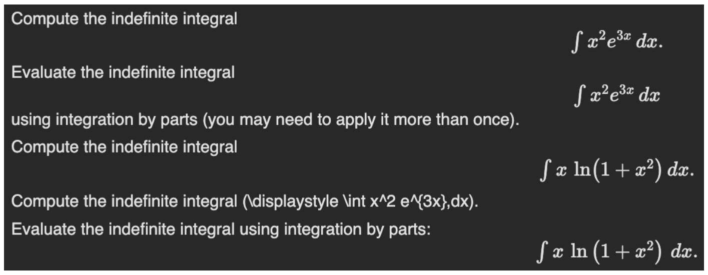
All the datasets and notebooks can be found in the same GitHub repository as this report.

## 2 Single-agent, Equation-based AMT

### 2.a Basic Task Creation

It took more time to set up the API credentials for Wolfram and OpenAI than the prototype for Equation Generator, which at this stage was capable of just generating the problems. It used to work by just calling an external model once, asking it to generate a problem; doing some basic rule-based validation of the output text and LaTex, checking for proper formatting and no answer leaks; and printing the problem back to the user.

Although this first version would generate good enough equations, it suffered from occasional incorrect LaTex formatting, which would get pass the validator, and even more crucially, lack of diversity in the generated tasks. Figure 1 shows 5 subsequent calls of the generator with identical parameters (the 4th one failed to render properly due to some LaTex issues, but you can still see the LaTex-like text), and the issue can be seen with a naked eye.



Compute the indefinite integral

$$\int x^2 e^{3x}\,dx.$$

Evaluate the indefinite integral

$$\int x^2 e^{3x}\,dx$$

using integration by parts (you may need to apply it more than once).

Compute the indefinite integral

$$\int x\ln(1+x^2)\,dx.$$

Compute the indefinite integral (\displaystyle \int x^2 e^{3x},dx).

Evaluate the indefinite integral using integration by parts:

$$\int x\ln\left(1+x^2\right)\,dx.$$

Figure 1: Consequent generation with Equation Generator

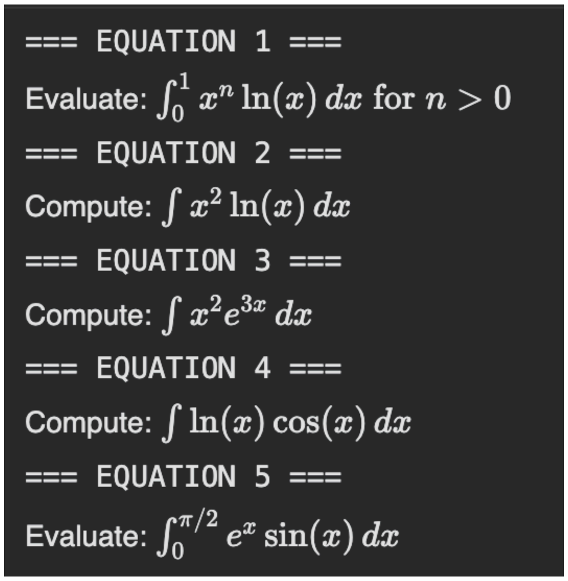
So, I set my focus on improving diversity.

### 2.b Diversity improvement

To improve diversity of generated equations, I went with the **prompt chaining** approach, where the first stage would generate a set of diverse 'skeleton' tasks - small objects consisting of the proposed equation structure, method to solve the equation, and a brief justification of the complexity level. Then, one of these skeletons would be chosen randomly by the agent, and fed into the second stage of the pipeline. This improved the diversity greatly, basically eliminating the duplicated tasks (although, we get somewhat-related ones occasionally). You can see the results in Figure 2

### 2.c Translation

The assignment requires all the **generated** text to be in Ukrainian. This is a problem, since models work better in English (Schut et al., 2025), and we can't compromise performance by prompting the model in Ukrainian for such a context-critical task as math problem generation. So, I solved this problem by adding a node to the end of the pipeline to translate the final output in Ukrainian (or any

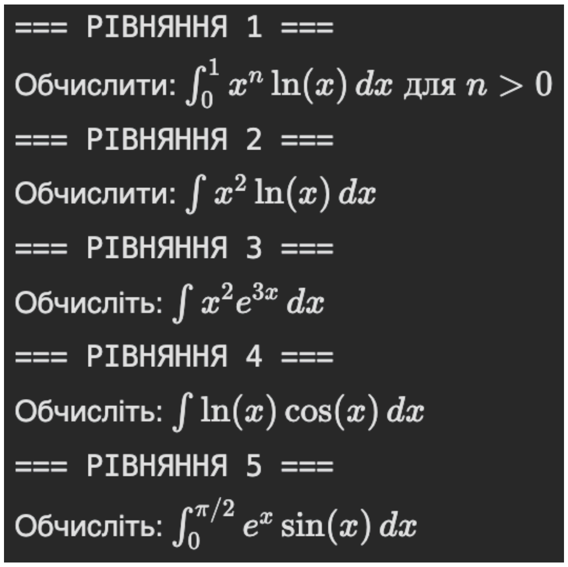


```
=== EQUATION 1 ===
Evaluate: $\int_0^1 x^n \ln(x)\,dx$ for $n > 0$
=== EQUATION 2 ===
Compute: $\int x^2 \ln(x)\,dx$
=== EQUATION 3 ===
Compute: $\int x^2 e^{3x}\,dx$
=== EQUATION 4 ===
Compute: $\int \ln(x)\cos(x)\,dx$
=== EQUATION 5 ===
Evaluate: $\int_0^{\pi/2} e^x \sin(x)\,dx$
```

Figure 2: Generation after applying diversity fixes

other language of your choosing). Not only this approach allows me to build prompt chains within a particular agent in English, but it also keeps English output intact, which will be important for the multiagent setup, and an introduction of the Solver agent.

As for the translation quality, although it remains at a decent level, the model occasionally uses constructions that a native Ukrainian-speaking person would not, like 'Обчислити' instead of 'Обчисліть' when giving a task to solve an equation. You can see the translated versions of the previously-generated set of equations in Figure 3.

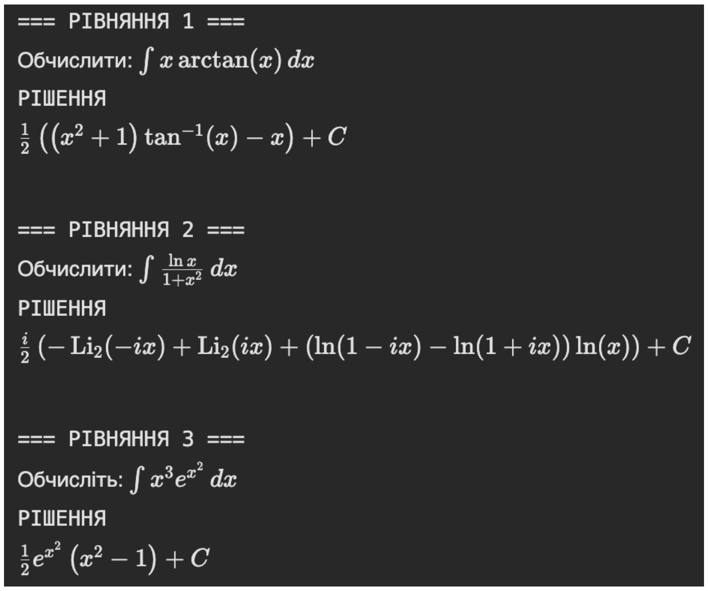


```
=== РІВНЯННЯ 1 ===
Обчислити: $\int_0^1 x^n \ln(x)\,dx$ для $n > 0$
=== РІВНЯННЯ 2 ===
Обчислити: $\int x^2 \ln(x)\,dx$
=== РІВНЯННЯ 3 ===
Обчисліть: $\int x^2 e^{3x}\,dx$
=== РІВНЯННЯ 4 ===
Обчисліть: $\int \ln(x)\cos(x)\,dx$
=== РІВНЯННЯ 5 ===
Обчисліть: $\int_0^{\pi/2} e^x \sin(x)\,dx$
```

Figure 3: Generation in Ukrainian

## 2.d Solver

Solving equations, even one-liners, directly with LLMs is never a good idea. LLMs shine at generating all kinds of text, and fail miserably when it comes to symbolic calculations. Thus, I gave my agent a tool - the API of WolframAlpha. With it, solving one-liners becomes trivial. Making this tool work, however, is anything but.



```
=== РІВНЯННЯ 1 ===
Обчислити: $\int x \arctan(x)\, dx$
РІШЕННЯ
$\frac{1}{2}\left(\left(x^2+1\right)\tan^{-1}(x)-x\right)+C$


=== РІВНЯННЯ 2 ===
Обчислити: $\int \frac{\ln x}{1+x^2}\, dx$
РІШЕННЯ
$\frac{i}{2}\left(-\operatorname{Li}_2(-ix)+\operatorname{Li}_2(ix)+(\ln(1-ix)-\ln(1+ix))\ln(x)\right)+C$


=== РІВНЯННЯ 3 ===
Обчисліть: $\int x^3 e^{x^2}\, dx$
РІШЕННЯ
$\frac{1}{2}e^{x^2}\left(x^2-1\right)+C$
```

Figure 4: Solutions

Two main problems I encountered with WA are response and input/output formatting. WolframAlpha is a very smart tool, with capabilities in basically any domain of science. And since its API was build to handle everything at once, parsing responses from the API is a real challenge. They use a concept of pods to store actual response data, but those pods are really poorly documented, and some of them can be either present or absent depending on the task. Fortunately, Claude managed to build a parser for these responses. As for the formatting, WA operates with either Wolfram Language, or Mathematica. None of these are compatible with LaTex, and there are no conversion libraries available. The only solution for this issue that I could see was making the LLM generate both LaTex and Wolfram Language statements for the problem on generation, and calling the LLM to convert WL into LaTex when solving the problem. Obviously, such an approach increases latency of the entire system, and is natively prone to errors, since we use an LLM.

## 2.e Evaluation

To get a more-or-less representative sample of the model outputs, I generated 30 problems for each of the 6 categories below. The groups are indexed for further experiments.

1. Calculus; Integration by Parts; Easy.
2. Calculus; Integration by Parts; Medium.
3. Calculus; Integration by Parts; Hard.
4. Calculus; Integration with Substitution; Hard.
5. Calculus; Volume Calculation; Hard.
6. Linear Algebra; Eigenvectors and Eigenvalues; Medium.

It took 35min to generate everything, and the operation consumed 337k tokens. Which comes down to 12sec and 1800 tokens per equation.

### 2.e.1 Intra-Group Diversity

First, I wanted to evaluate diversity of the outputs, which we had troubles with. I did so by comparing the generated LaTex and Wolfram statements within groups. Before comparison, both LaTex and Wolfram statements were normalized - integrals were made indefinite, all the spaces and extra symbols cut, an so on.

According to Table 1, the model does generate diverse equations, with most groups scoring uniqueness ratio of over 90%. You can also notice that the model did struggle with Easy Integration by Parts (group 1) a bit, where roughly every 4th equation gets repeated. I do not have an explanation as for why this particular group was challenging for the model.

| group | total | unique LaTex | unique Wolfram | ratio LaTex | ratio Wolfram |
|---|---|---|---|---|---|
| 1 | 30 | 23 | 23 | 0.76 | 0.76 |
| 2 | 30 | 27 | 28 | 0.9 | 0.93 |
| 3 | 30 | 29 | 28 | 0.96 | 0.93 |
| 4 | 30 | 29 | 29 | 0.96 | 0.96 |
| 5 | 30 | 29 | 30 | 0.96 | 1 |
| 6 | 30 | 30 | 30 | 1 | 1 |

Table 1: Intra-Group Equation Diversity

You can see that the numbers for unique Wolfram and LaTex equations are slightly different for most rows. This small difference can be attributed to the equation normalization error, rather than incorrect translation of LaTex into Wolfram Language.

### 2.e.2 Inter-Group Diversity

Next, I wanted to check the diversity between groups. I was especially interested in getting this metric for groups $1-3$, since it would demonstrate how well the model understands quite an abstract dimension of complexity.

Table 2 demonstrates that the model separates groups exceptionally well. Cells in the table demonstrate how many LaTex equations overlap between different groups. And we end up with zeros in almost every cell, which is just what we want.

### 2.e.3 Correctness

I couldn't come up with an approach to evaluate solution correctness. All the compute is done by WolframAlpha, so verifying mathematical operations with an LLM - which can hallucinate, unlike WA - is out of the question. In theory, we could check how well the model of our choosing translates LaTex into WA language and vise-versa using a council of judge models. And while such a method would work well for a small model, it's not really applicable to gpt-5.2, which is arguably more powerful than any other judge model.

| Group | Group | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | — | 1 | 0 | 0 | 0 | 0 |
| 2 | 1 | — | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | — | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | — | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | — | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | — |

Table 2: Inter-Group Equation Diversity

#### 2.e.4 Conclusion

Since AMT already manages to generate diverse sets of equations, respecting inputs for domain, topic and complexity, we can move on to making it generate something more complex than one-liners.

### 2.f Architecture

The final architecture of the single-agent system at this stage is demonstrated on Figure 5.

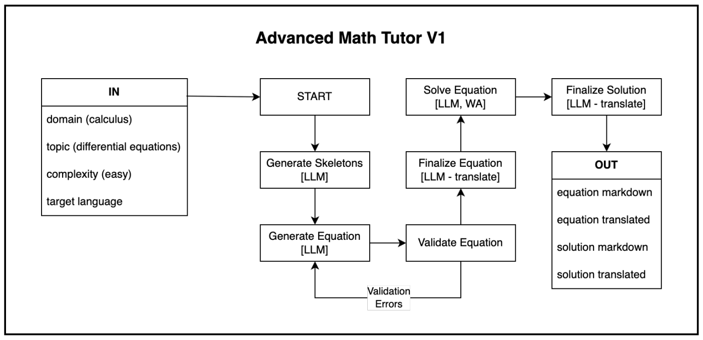


Figure 5: Architecture of the system

# 3 Problem-based AMT

### 3.a Equation vs Problem

As mentioned earlier, Equations and Problems differ in their structure and intended purpose. An Equation is a self-contained one-liner that tests a specific mechanical skill, such as computing $\int x^2 e^{3x}\,dx$ or solving $\frac{1}{x-2}+\frac{1}{x+3}=\frac{1}{2}$. These can be fed directly into WolframAlpha and solved without any additional reasoning.

On the other hand, problems are multi-part tasks that require understanding context, setting up the mathematical model, and then solving the resulting equations. In a Problem, we might ask the student to find the volume of a solid of revolution, optimize a cost function given certain constraints, or prove a property of a matrix transformation. The mathematical equations are not given explicitly. Instead, the student must derive them from the problem statement.

Problem-based system emerged as a result of my attempts to increase the difficulty of equations. Eventually, I was satisfied with the quality of generating both tasks and solutions. However, since I had to rebuild almost every component of the system to make it work with Problems, AMT of this stage could no longer generate Equations.

## 3.b Problem Generator

Problems and Equations are built in a somewhat-different manner. When generating Equation, the model must produce both LaTex and Wolfram query. Contrary to that, Problem contains only LaTex along with metadata. We cannot generate Wolfram query immediately - there are simply no individual equations that can solve a Problem. And we also don't want to strategize and generate a set of queries immediately. We'd force the model to do a lot of different things at once, which always leads to degradation in the response quality.

To ensure problem diversity, I adapted the skeleton-based approach used in Equations, making the structures richer. Each skeleton contains a `concept` (one-sentence description), a `problem_type` (computational, proof, conceptual, application, etc.), and a `key_twist` that makes the problem non-trivial. This ensures that generated Problems are diverse not just in their mathematical content, but also in the type of thinking they require.

## 3.c Problem Solver

Solving Problems required building an entirely new reasoning system. Unlike Equations, where a single WolframAlpha call suffices, Problems demand a multi-step approach that combines logical reasoning with symbolic computation.

The solver operates in four main phases: planning, execution, compilation, and validation.

### 3.c.1 Planning

During **planning**, the LLM analyzes the problem and produces a structured solution plan - a sequence of steps, each tagged with a type. Three step types are supported:

- `reasoning` - logical deduction, setting up equations, interpreting results
- `symbolic_compute` - integration, differentiation, solving equations (delegated to WolframAlpha)
- `verify` - checking that a result satisfies the original constraints

Each step also specifies its dependencies (which previous steps it builds upon) and, for `symbolic_compute` steps, a WolframAlpha query in plain text. The plan is capped at 10 steps (default, can be tweaked through configs) to prevent runaway token consumption.

### 3.c.2 Execution

During **execution**, the agent iterates through the plan. For each step, it constructs a prompt containing the problem statement, the current step's description, the results of previous steps (for context), and the WolframAlpha output if applicable. The LLM then generates the reasoning for that step. If the step requires compute (has a special tag), we also call WolframAlpha to perform the operation. This loop continues until all steps are executed.

### 3.c.3 Compilation

The **compilation** phase takes all executed steps and asks the LLM to produce a polished, coherent solution with smooth transitions and a clearly stated final answer. We do it because individual step outputs can be somewhat disjointed.

### 3.c.4 Validation

Finally, **validation** checks the compiled solution for common issues: missing final answers, incomplete reasoning, LaTeX formatting errors, and logical inconsistencies. If errors are found and the attempt count is below the threshold, the entire process restarts with a fresh plan. The system tracks the 'best' solution across attempts (one with the fewest errors) and returns it even if validation never fully passes. The best result tracking was implemented after burning 100k+ tokens on a single problem, and having all solutions rejected due to minor rule-based errors.

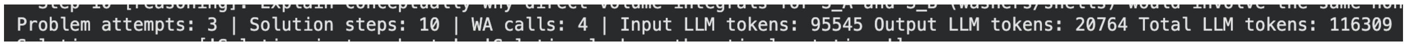
### 3.c.5 Resilience

The last detail that I'd like to mention for the section concerns JSON parsing. The OpenAI API with structured outputs occasionally truncates responses when they exceed `max_completion_tokens`. This results in malformed JSON that fails to parse, and occasionally ruins the whole solution, that might've been running for a few minutes and had burnt through tens of thousand of tokens. To handle this, I implemented a utility wrapper that attempts multiple retries with slightly varied prompts and increased `max_completion_tokens` before giving up.

## 3.d Problems with Problems

Great power of the reasoning-based solver, comes with great costs. A typical Problem **solution attempt** requires 8-12 LLM calls (1 for planning, 6-10 for step execution, 1 for compilation) plus a few WolframAlpha calls for symbolic computation steps. Each LLM call involves substantial context (the problem, previous steps, WA results), which burns through tokens.

To give you some context, solving a single Problem consumes 60-80k tokens and takes 2-3 minutes if we are lucky. And it can grow into 100k+ tokens and a few minutes of compute if we are not (Figure 6). For comparison, solving an Equation with a non-reasoning solver uses fewer than 500 tokens (just for the LaTex extraction and translation) and completes in under 5 seconds.



```
Problem attempts: 3 | Solution steps: 10 | WA calls: 4 | Input LLM tokens: 95545 Output LLM tokens: 20764 Total LLM tokens: 116309
```

Figure 6: When I 'got unlucky' and burnt through ∼115k tokens

On the good side, translation costs are negligible for the Problems. We only translate complete problems and compiled solutions, which rarely exceeds a few thousand tokens in total.

### 3.e Evaluation

For the evaluation, I generated 5 problems with solutions for 4 categories listed below.

1. Calculus; Volume Calculation; Hard.
2. Linear Algebra; SVD; Hard.
3. Statistics; Statistical Tests; Medium.
4. Calculus; Deep Learning Inspired; Medium.

77 minutes and 1.6kk tokens well-spent, which comes down to the average of 3min 50sec and 80k tokens per problem.

Evaluation for problems is much more complicated than evaluation for equations. We can no longer send the content to either an LLM endpoint or WA, since solving a problem requires reasoning and mathematical modeling. The only solution that I could think of was feeding the problems 1-by-1 into something like ChatGPT, where the models can reason and have an access to external tools.

Eventually, I picked 3 random problems out of each category, and evaluated them using the strategy described above. Then, I pasted solutions in the same sessions, asking for comparison between the generated and given content. Ultimately, I get 6/6 solutions for Medium-level tasks correct. For the Hard-level ones, however, I only got 4/6, with the solver making a mistake somewhere in the middle of the solution pipeline.

### 3.f Architecture

The final architecture of the at this stage is demonstrated on Figure 7. You can clearly see already that the system has become somewhat bloated. This issue will be addressed in the next section.

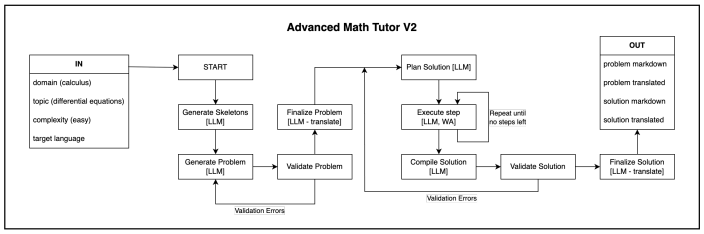


Figure 7: Architecture of the system

# 4 Multi-agent system

## 4.a Why we need many agents

I wanted AMT to work with both Equations and Problems. At that point, the Problem-based system looked bloated already, and making it work with Equations would make it completely unmaintainable. The solution that seemed natural was to split the system into three specialized agents: an **Equation Generator** for one-liners, a **Problem Generator** for complex tasks, and a **Problem Solver** that could handle both types of input. Although, in hindsight, I should've split the Solver as well, since pathes for solving Equations and Problems are almost completely different.

The shared layer between agents includes common utilities (translation, WA integration, LaTeX rendering), Pydantic models (`Problem`, `Equation`), and related prompt templates.

## 4.b Modifications to the solver

In this version of AMT, solver is managed by the `step_by_step` flag. Normally, the solver goes through the step-by-step pipeline. However, when the flag is ticked, and a Wolfram input is provided in a query, the solver fast-tracks through the dedicated node, providing solution through the WA, with no extra calls to the LLM.

Unfortunately, I couldn't avoid LLM calls entirely in this pipeline. WA still responds in a format-of-their-own, which is then parsed into LaTex through an LLM call.

## 4.c Step-by-step Equations

An interesting capability emerged from the unified solver: the ability to generate detailed, step-by-step solutions for Equations. Although I designed Equations for quick drill practice, a student might occasionally want to see the full derivation-for instance, when learning a new integration technique.

By setting `step_by_step = True` for an Equation, the solver treats it as a mini-Problem: it generates a plan (typically 3-4 steps for a one-liner), executes each step with full reasoning, and compiles a polished solution. The result is a detailed walkthrough that explains not just the answer, but the method. It looks similar to the step-by-step functionality of WolframAlpha, which I also wanted to add to the system, but found it hidden behind a paywall, which requires setting up a call with a sales agent to unlock.

Obviously, this ability comes at a cost. Step-by-step Equation solving consumes 10k-20k tokens compared to 300-500 for quick solving. However, it provides a nice pedagogical option for whenever the student needs more than just the answer. And, since Problem Solver now acts as an independent agent, step-by-step solution does not need to be generated at problem-generation anymore.

## 4.d Evaluation

At this stage, AMT is composed of 2 systems discussed before. The logic under the hood of the agents remains the same, so no extra evalutation was run.

## 4.e Architecture

Architecture of the final system is demonstrated on Figure 8. You can see that the agents became smaller in comparison to the previous version. Also you can notice that Equation and Problem generator look identical. And they use the same architecture indeed, but very different prompts under the hood.

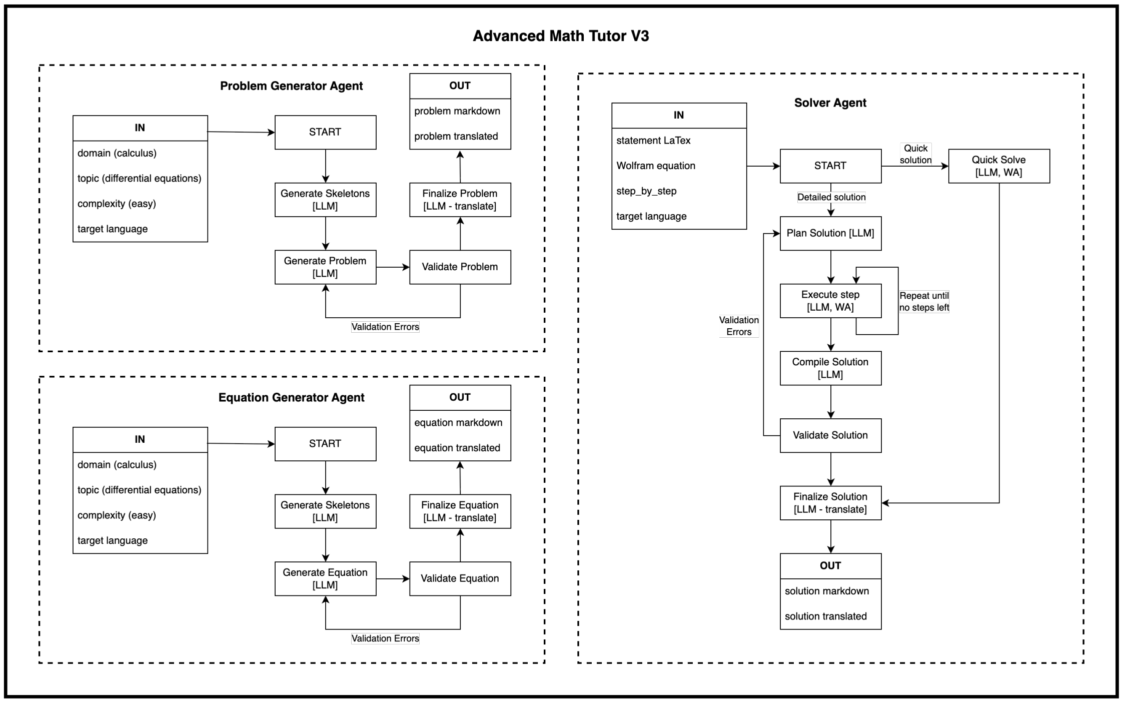


Figure 8: Architecture of Advanced Math Tutor

## 5 Conclusions

So far, it's been the most enjoyable lab of the course. I have not had any experience with agentic AI before, and I really liked building a system like AMT, with my favorite part definitely being the reasoning-like solver component.

However, there are also a few things which, if addressed, can make the lab even better. First is the requirement for all the generated content to be in Ukrainian. As far as I know, no one in the group has actually built a pipeline end-to-end in Ukrainian, and this requirement has been always handled by a translation call. I'm not sure if there is any educational value in slapping a 'translate' node on top of either RAG or an agentic pipeline, so I can't really understand why this requirement was added to the task. Another issue is the requirement about evaluation dataset. Most advanced math problems are not solved with just a single number, and require some modeling to even get to the equations. And I'm not sure if feeding both the generated and sample solution strategy to an LLM counts for a quality evaluation.

## References

Lisa Schut, Yarin Gal, and Sebastian Farquhar. Do multilingual llms think in english?, 2025. URL https://arxiv.org/abs/2502.15603.

# Supplement

## 5.a Equations

### Generation

=== EQUATION ===
Compute: $\int \frac{7x+3}{\sqrt{3x^2-2x+5}}\,dx$

=== EQUATION (Ukrainian) ===
Обчислити: $\int \frac{7x+3}{\sqrt{3x^2-2x+5}}\,dx$

Generation Tokens: Input: 953 | Output: 615 | Total: 1568
Errors: []
CPU times: user 189 ms, sys: 18 ms, total: 207 ms
Wall time: 10.7 s

### Quick Solution

=== SOLUTION ===

$$\frac{1}{9}\left(21\sqrt{3x^2-2x+5}-16\sqrt{3}\,\ln\left(\sqrt{9x^2-6x+15}-3x+1\right)\right)+C$$

### Step-by-step Solution (English)

Let

$$Q(x)=3x^2-2x+5 \qquad \Rightarrow \qquad Q'(x)=6x-2.$$

We want to compute

$$\int \frac{7x+3}{\sqrt{Q(x)}}\,dx.$$

**1) Decompose the numerator using $Q'(x)$.** Seek constants $A, B$ such that

$$7x+3=AQ'(x)+B=A(6x-2)+B.$$

Match coefficients:

$$6A=7 \;\Rightarrow\; A=\frac{7}{6}, \qquad -2A+B=3 \;\Rightarrow\; B=3+2A=3+\frac{7}{3}=\frac{16}{3}.$$

Thus,

$$\frac{7x+3}{\sqrt{Q(x)}}=\frac{\frac{7}{6}\,Q'(x)}{\sqrt{Q(x)}}+\frac{\frac{16}{3}}{\sqrt{Q(x)}}.$$

So the integral splits as

$$\int \frac{7x+3}{\sqrt{Q(x)}}\,dx=\frac{7}{6}\int \frac{Q'(x)}{\sqrt{Q(x)}}\,dx+\frac{16}{3}\int \frac{dx}{\sqrt{Q(x)}}.$$

**2) First integral.** Let $u=Q(x)$, so $du=Q'(x)\,dx$. Then

$$\frac{7}{6}\int \frac{Q'(x)}{\sqrt{Q(x)}}\,dx=\frac{7}{6}\int u^{-1/2}\,du=\frac{7}{6}\cdot 2u^{1/2}=\frac{7}{3}\sqrt{Q(x)}=\frac{7}{3}\sqrt{3x^2-2x+5}.$$

**3) Second integral (complete the square).** Complete the square:

$$3x^2 - 2x + 5 = 3\left(x - \frac{1}{3}\right)^2 + \frac{14}{3} = 3\left[\left(x - \frac{1}{3}\right)^2 + \frac{14}{9}\right].$$

Therefore,

$$\frac{16}{3}\int \frac{dx}{\sqrt{Q(x)}} = \frac{16}{3}\int \frac{dx}{\sqrt{3}\sqrt{\left(x-\frac{1}{3}\right)^2 + \left(\frac{\sqrt{14}}{3}\right)^2}} = \frac{16}{3\sqrt{3}}\int \frac{dx}{\sqrt{\left(x-\frac{1}{3}\right)^2 + \left(\frac{\sqrt{14}}{3}\right)^2}}.$$

Using the standard formula

$$\int \frac{du}{\sqrt{u^2 + a^2}} = \operatorname{asinh}\left(\frac{u}{a}\right) + C = \ln\left|u + \sqrt{u^2 + a^2}\right| + C,$$

with $u = x - \frac{1}{3}$ and $a = \frac{\sqrt{14}}{3}$, we get

$$\frac{16}{3\sqrt{3}}\operatorname{asinh}\left(\frac{x - \frac{1}{3}}{\frac{\sqrt{14}}{3}}\right) = \frac{16}{3\sqrt{3}}\operatorname{asinh}\left(\frac{3x-1}{\sqrt{14}}\right).$$

**4) Combine results.** Putting the two parts together,

$$\boxed{\int \frac{7x+3}{\sqrt{3x^2-2x+5}}\,dx = \frac{7}{3}\sqrt{3x^2-2x+5} + \frac{16}{3\sqrt{3}}\operatorname{asinh}\left(\frac{3x-1}{\sqrt{14}}\right) + C.}$$

An equivalent logarithmic form is

$$\boxed{\int \frac{7x+3}{\sqrt{3x^2-2x+5}}\,dx = \frac{7}{3}\sqrt{3x^2-2x+5} + \frac{16}{3\sqrt{3}}\ln\left|\left(x - \frac{1}{3}\right) + \sqrt{\left(x-\frac{1}{3}\right)^2 + \frac{14}{9}}\right| + C.}$$

**5) (Check by differentiation).** Differentiating $\frac{7}{3}\sqrt{Q(x)}$ gives

$$\frac{7}{3}\cdot\frac{Q'(x)}{2\sqrt{Q(x)}} = \frac{7}{6}\frac{Q'(x)}{\sqrt{Q(x)}},$$

and differentiating the asinh term gives

$$\frac{16}{3\sqrt{3}}\cdot\frac{1}{\sqrt{\left(x-\frac{1}{3}\right)^2 + \left(\frac{\sqrt{14}}{3}\right)^2}} = \frac{16}{3}\cdot\frac{1}{\sqrt{Q(x)}}.$$

Summing yields

$$\frac{\frac{7}{6}Q'(x) + \frac{16}{3}}{\sqrt{Q(x)}} = \frac{7x+3}{\sqrt{Q(x)}},$$

confirming the result.

## Step-by-step Solution (Ukrainian)

Нехай

$$Q(x) = 3x^2 - 2x + 5 \qquad \Rightarrow \qquad Q'(x) = 6x - 2.$$

Ми хочемо обчислити

$$\int \frac{7x+3}{\sqrt{Q(x)}}\,dx.$$

**1) Розкладемо чисельник, використовуючи $Q'(x)$.** Знайдемо сталі $A, B$ такі, що

$$7x + 3 = AQ'(x) + B = A(6x-2) + B.$$

Прирівняємо коефіцієнти:

$$6A = 7 \;\Rightarrow\; A = \frac{7}{6}, \qquad -2A + B = 3 \;\Rightarrow\; B = 3 + 2A = 3 + \frac{7}{3} = \frac{16}{3}.$$

Отже,

$$\frac{7x+3}{\sqrt{Q(x)}} = \frac{\frac{7}{6}Q'(x)}{\sqrt{Q(x)}} + \frac{\frac{16}{3}}{\sqrt{Q(x)}}.$$

Тому інтеграл розбивається як

$$\int \frac{7x+3}{\sqrt{Q(x)}}\,dx = \frac{7}{6}\int \frac{Q'(x)}{\sqrt{Q(x)}}\,dx + \frac{16}{3}\int \frac{dx}{\sqrt{Q(x)}}.$$

**2) Перший інтеграл.** Нехай $u = Q(x)$, тоді $du = Q'(x)\,dx$. Тоді

$$\frac{7}{6}\int \frac{Q'(x)}{\sqrt{Q(x)}}\,dx = \frac{7}{6}\int u^{-1/2}\,du = \frac{7}{6}\cdot 2u^{1/2} = \frac{7}{3}\sqrt{Q(x)} = \frac{7}{3}\sqrt{3x^2-2x+5}.$$

**3) Другий інтеграл (доповнення до квадрата).** Доповнимо до квадрата:

$$3x^2 - 2x + 5 = 3\left(x - \frac{1}{3}\right)^2 + \frac{14}{3} = 3\left[\left(x-\frac{1}{3}\right)^2 + \frac{14}{9}\right].$$

Тому

$$\frac{16}{3}\int \frac{dx}{\sqrt{Q(x)}} = \frac{16}{3}\int \frac{dx}{\sqrt{3}\sqrt{\left(x-\frac{1}{3}\right)^2 + \left(\frac{\sqrt{14}}{3}\right)^2}} = \frac{16}{3\sqrt{3}}\int \frac{dx}{\sqrt{\left(x-\frac{1}{3}\right)^2 + \left(\frac{\sqrt{14}}{3}\right)^2}}.$$

Використовуючи стандартну формулу

$$\int \frac{du}{\sqrt{u^2+a^2}} = \operatorname{asinh}\left(\frac{u}{a}\right) + C = \ln\left|u + \sqrt{u^2+a^2}\right| + C,$$

для $u = x - \frac{1}{3}$ та $a = \frac{\sqrt{14}}{3}$, дістаємо

$$\frac{16}{3\sqrt{3}}\operatorname{asinh}\left(\frac{x-\frac{1}{3}}{\frac{\sqrt{14}}{3}}\right) = \frac{16}{3\sqrt{3}}\operatorname{asinh}\left(\frac{3x-1}{\sqrt{14}}\right).$$

**4) Об'єднаємо результати.** Отже,

$$\boxed{\int \frac{7x+3}{\sqrt{3x^2-2x+5}}\,dx = \frac{7}{3}\sqrt{3x^2-2x+5} + \frac{16}{3\sqrt{3}}\operatorname{asinh}\left(\frac{3x-1}{\sqrt{14}}\right) + C.}$$

Еквівалентна логарифмічна форма:

$$\boxed{\int \frac{7x+3}{\sqrt{3x^2-2x+5}}\,dx = \frac{7}{3}\sqrt{3x^2-2x+5} + \frac{16}{3\sqrt{3}}\ln\left|\left(x-\frac{1}{3}\right)+\sqrt{\left(x-\frac{1}{3}\right)^2+\frac{14}{9}}\right| + C.}$$

**5) Перевірка диференціюванням.** Похідна від $\frac{7}{3}\sqrt{Q(x)}$ дорівнює

$$\frac{7}{3}\cdot\frac{Q'(x)}{2\sqrt{Q(x)}} = \frac{7}{6}\frac{Q'(x)}{\sqrt{Q(x)}},$$

а похідна від asinh-доданка дає

$$\frac{16}{3}\cdot\frac{1}{\sqrt{Q(x)}}.$$

У сумі отримуємо

$$\frac{\frac{7}{6}Q'(x)+\frac{16}{3}}{\sqrt{Q(x)}} = \frac{7x+3}{\sqrt{Q(x)}},$$

що підтверджує результат.

**Generation Metadata** The solution took quite a while to generate partially due to the 'flex' (i.e. non-priority) service tier.

Solution Tokens: Input: 31876 | Output: 17736 | Total: 49612
WA calls: 15 | Errors: 1
Solution errors: ['Solution lacks mathematical notation.']
CPU times: user 335 ms, sys: 30.4 ms, total: 365 ms
Wall time: 2min 41s

## 5.b Problems

### 5.b.1 Generation

#### English

Construct a counterexample to the claim:
*If two planar regions $R_1$ and $R_2$ lie in the strip $0 \le x \le 1$ and have the same area, then the solids obtained by revolving them about the $x$-axis have the same volume.*
**Your task:**
Define two regions $R_1$ and $R_2$ in the plane that both lie between the vertical lines $x = 0$ and $x = 1$, and that each can be described as

$$R_i = \{(x,y) : 0 \le x \le 1,\ 0 \le y \le f_i(x)\}$$

for some nonnegative, piecewise-continuous functions $f_1$ and $f_2$.
Ensure the areas are equal:

$$\int_0^1 f_1(x)\,dx = \int_0^1 f_2(x)\,dx.$$

Show that the volumes about the $x$-axis are different by computing and comparing

$$V_1 = \pi \int_0^1 \bigl(f_1(x)\bigr)^2\,dx, \qquad V_2 = \pi \int_0^1 \bigl(f_2(x)\bigr)^2\,dx,$$

and verifying that $V_1 \neq V_2$.
Provide explicit formulas for $f_1$ and $f_2$ (simple choices such as piecewise-constant or piecewise-linear functions are acceptable), and include all necessary area and volume integrals in your justification.

### Ukrainian

Побудуйте контрприклад до твердження:
*Якщо дві плоскі області $R_1$ і $R_2$ лежать у смузі $0 \le x \le 1$ і мають однакову площу, то тіла, отримані обертанням їх навколо осі $x$, мають однаковий об'єм.*
**Ваше завдання:**
Задайте дві області $R_1$ і $R_2$ на площині, які обидві лежать між вертикальними прямими $x = 0$ та $x = 1$, і кожну з яких можна описати як

$$R_i = \{(x, y) : 0 \le x \le 1,\ 0 \le y \le f_i(x)\}$$

для деяких невід'ємних, кусочно-неперервних функцій $f_1$ і $f_2$.
Забезпечте рівність площ:

$$\int_0^1 f_1(x)\,dx = \int_0^1 f_2(x)\,dx.$$

Покажіть, що об'єми при обертанні навколо осі $x$ різні, обчисливши та порівнявши

$$V_1 = \pi \int_0^1 \bigl(f_1(x)\bigr)^2\,dx, \qquad V_2 = \pi \int_0^1 \bigl(f_2(x)\bigr)^2\,dx,$$

і перевіривши, що $V_1 \neq V_2$.
Надайте явні формули для $f_1$ і $f_2$ (припустимі прості варіанти, такі як кусочно-сталі або кусочно-лінійні функції), і включіть усі необхідні інтеграли площі та об'єму у ваше обґрунтування.

### Metadata

*Generation Tokens:* Input: 1753 | Output: 2177 | Total: 3930
*Errors:* `[Problem statement looks like it includes solution/answer-like content.]`
*CPU times:* user 81.7 ms, sys: 9 ms, total: 90.7 ms
*Wall time:* 20.2 s

## 5.b.2 Solution

### English

**Counterexample.** Define two nonnegative, piecewise-continuous functions on $[0, 1]$:

$$f_1(x) = 1 \quad (0 \le x \le 1), \qquad f_2(x) = \begin{cases} 2, & 0 \le x \le \frac{1}{2}, \\ 0, & \frac{1}{2} < x \le 1. \end{cases}$$

**1) Define the planar regions in the strip** $0 \le x \le 1$. For $i = 1, 2$, let

$$R_i = \{(x, y) : 0 \le x \le 1,\ 0 \le y \le f_i(x)\}.$$

Then $R_1$ is the rectangle under $y = 1$ on $[0, 1]$, and $R_2$ is the rectangle under $y = 2$ on $\left[0, \frac{1}{2}\right]$ (and zero height on $\left(\frac{1}{2}, 1\right]$). Both lie entirely in the strip $0 \le x \le 1$.
**2) Check that the areas are equal.** The area of $R_1$ is

$$A_1 = \int_0^1 f_1(x)\,dx = \int_0^1 1\,dx = 1.$$

The area of $R_2$ is

$$A_2 = \int_0^1 f_2(x)\,dx = \int_0^{1/2} 2\,dx + \int_{1/2}^1 0\,dx = 2 \cdot \frac{1}{2} + 0 = 1.$$

Hence,

$$\int_0^1 f_1(x)\,dx = \int_0^1 f_2(x)\,dx = 1,$$

so the regions have the same area.
**3) Compute and compare the volumes upon revolving about the $x$-axis.** Using the disk method, the volume from $R_1$ is

$$V_1 = \pi \int_0^1 \big(f_1(x)\big)^2\,dx = \pi \int_0^1 1^2\,dx = \pi.$$

For $R_2$,

$$V_2 = \pi \int_0^1 \big(f_2(x)\big)^2\,dx = \pi \left( \int_0^{1/2} 2^2\,dx + \int_{1/2}^1 0^2\,dx \right) = \pi \left( \int_0^{1/2} 4\,dx \right) = \pi \left( 4 \cdot \frac{1}{2} \right) = 2\pi.$$

Since

$$V_1 = \pi \ne 2\pi = V_2,$$

the solids have different volumes despite the regions having the same area.
**Final conclusion.** The claim is false: the regions under $f_1(x) = 1$ on $[0, 1]$ and under $f_2(x) = 2$ on $\left[0, \frac{1}{2}\right]$ (and 0 on $\left(\frac{1}{2}, 1\right]$) have equal area, but their volumes of revolution about the $x$-axis are $\pi$ and $2\pi$, respectively.

## Ukrainian

**Контрприклад.** Визначимо дві невід'ємні, кусочно-неперервні функції на $[0, 1]$:

$$f_1(x) = 1 \quad (0 \le x \le 1), \qquad f_2(x) = \begin{cases} 2, & 0 \le x \le \frac{1}{2}, \\ 0, & \frac{1}{2} < x \le 1. \end{cases}$$

**1) Визначимо плоскі області в смузі** $0 \le x \le 1$. Для $i = 1, 2$ покладемо

$$R_i = \{(x, y) : 0 \le x \le 1,\ 0 \le y \le f_i(x)\}.$$

Тоді $R_1$ --- це прямокутник під $y = 1$ на $[0, 1]$, а $R_2$ --- це прямокутник під $y = 2$ на $\left[0, \frac{1}{2}\right]$ (і нульова висота на $\left(\frac{1}{2}, 1\right]$). Обидві області повністю лежать у смузі $0 \le x \le 1$.
**2) Перевіримо, що площі рівні.** Площа $R_1$ дорівнює

$$A_1 = \int_0^1 f_1(x)\,dx = \int_0^1 1\,dx = 1.$$

Площа $R_2$ дорівнює

$$A_2 = \int_0^1 f_2(x)\,dx = \int_0^{1/2} 2\,dx + \int_{1/2}^1 0\,dx = 2 \cdot \frac{1}{2} + 0 = 1.$$

Отже,

$$\int_0^1 f_1(x)\,dx = \int_0^1 f_2(x)\,dx = 1,$$

тому області мають однакову площу.

**3) Обчислимо та порівняємо об'єми при обертанні навколо осі $x$.** За методом дисків об'єм, отриманий з $R_1$, дорівнює

$$V_1 = \pi \int_0^1 \big(f_1(x)\big)^2\,dx = \pi \int_0^1 1^2\,dx = \pi.$$

Для $R_2$ маємо

$$V_2 = \pi \int_0^1 \big(f_2(x)\big)^2\,dx = \pi \left( \int_0^{1/2} 2^2\,dx + \int_{1/2}^1 0^2\,dx \right) = \pi \left( \int_0^{1/2} 4\,dx \right) = \pi \left( 4 \cdot \frac{1}{2} \right) = 2\pi.$$

Оскільки

$$V_1 = \pi \neq 2\pi = V_2,$$

тіла мають різні об'єми, попри те що області мають однакову площу.

**Остаточний висновок.** Твердження є хибним: області під $f_1(x) = 1$ на $[0, 1]$ та під $f_2(x) = 2$ на $\left[0, \frac{1}{2}\right]$ (і 0 на $\left(\frac{1}{2}, 1\right]$) мають однакову площу, але їхні об'єми обертання навколо осі $x$ дорівнюють відповідно $\pi$ та $2\pi$.

## Metadata

*Solution Tokens:* Input: 35504 | Output: 11499 | Total: 47003
*WA calls:* 12 | Errors: 1
*Solution errors:* `[Solution lacks mathematical notation.]`
*CPU times:* user 309 ms, sys: 24.2 ms, total: 333 ms
*Wall time:* 2 min 3 s