

# RL for LLM: Enhanced CoT Reasoning with Rule-based GRPO

Chenwei WAN, Yifei WANG, Hongxu CHEN  
Master DATAI, Institut Polytechnique de Paris  
{chenwei.wan, wang, hongxu.chen}@ip-paris.fr

March 20, 2025

Code: <https://github.com/Mechtachx/GRPO>

## 1 Introduction

Reinforcement learning has proven effective in enhancing the reasoning capabilities of large language models (LLMs). However, conventional methods such as Proximal Policy Optimization (PPO) rely on value models, which introduce instability and high computational overhead. To address these limitations, Group Relative Policy Optimization (GRPO) (1) has been proposed as an alternative, leveraging group-based reward computation to improve training efficiency and stability.

Despite its advantages, reproducing GRPO systematically presents significant challenges. First, the computational cost of training reinforcement learning-based fine-tuning models remains high, making it difficult to explore and modify methods extensively. Additionally, replicating prior results with limited computational resources introduces uncertainty, as slight variations in hyperparameters or training setups can significantly impact performance.

Existing research has explored various reinforcement learning techniques for improving LLMs. PPO is widely used in Reinforcement Learning from Human Feedback (RLHF) (2), but its reliance on value models often leads to high variance and mode collapse. Alternative methods, such as Direct Preference Optimization (DPO) (3) and Rejection Sampling (RS) (4), aim to improve policy optimization without explicit value modeling. However, these approaches may still suffer from inefficiencies in reward computation. GRPO proposes an alternative strategy by leveraging group-wise relative ranking, eliminating the need for separate value estimation.

In this study, we aim to demonstrate the effectiveness of reinforcement learning in enhancing the reasoning capabilities of large language models (LLMs). To this end, we reproduce the GRPO framework using Qwen2.5-0.5B and evaluate its performance against both a supervised fine-tuned (SFT) model and the original pre-trained model. Our primary focus is on assessing the impact of RL-based fine-tuning on reasoning accuracy rather than comparing different RL algorithms. As such, we do not include a direct comparison between GRPO and alternative RL methods like PPO. We evaluate models using the Pass@1 metric on the GSM8K

(5) benchmark, a widely used dataset for mathematical reasoning. Experimental results demonstrate significant improvements in reasoning performance with GRPO, confirming the effectiveness of RL-based fine-tuning. However, our study is constrained by the limited model size and evaluation on a single dataset (GSM8K), leaving open questions about its generalizability to other reasoning tasks.

These results suggest that GRPO-based reinforcement learning fine-tuning can enhance LLM reasoning abilities while maintaining training efficiency. Future work should investigate scalability to larger models, generalization across diverse reasoning tasks, and potential optimizations to further reduce computational costs.

## 2 Background

Large language models (LLMs) have significantly advanced mathematical reasoning and structured problem-solving, achieving strong performance on benchmarks such as GSM8K and MATH. However, these models still require further fine-tuning to enhance their capabilities in structured reasoning tasks, such as mathematical problem-solving and code generation.

### 2.1 Reinforcement Learning for LLM Fine-tuning

Reinforcement learning (RL) is a framework for sequential decision-making where an agent interacts with an environment to maximize cumulative rewards (6). In RL, an agent observes a state  $s_t$ , takes an action  $a_t$  according to a policy  $\pi_\theta(a_t | s_t)$ , receives a reward  $r_t$ , and transitions to the next state  $s_{t+1}$ . The agent aims to learn an optimal policy  $\pi_\theta^*$  that maximizes the expected return:

$$J(\theta) = \mathbb{E}_{\tau \sim p_\theta} \left[ \sum_{t=0}^T \gamma^t r_t \right],$$

where  $\tau$  represents a trajectory, and  $\gamma \in (0, 1]$  is the discount factor.

In the context of LLMs, \*\*Reinforcement Learning from Human Feedback (RLHF)\*\* (2) has emerged as a powerful approach for aligning model outputs with human preferences. RLHF typically involves a reward

model trained on human annotations, guiding policy updates to improve response quality.

## 2.2 Proximal Policy Optimization (PPO) and Its Limitations

Among RL-based methods, **Proximal Policy Optimization (PPO)** (7) has been widely adopted for fine-tuning LLMs, including models such as ChatGPT (2). PPO is an actor-critic algorithm that updates the policy using a clipped surrogate objective:

$$L(\theta) = \mathbb{E}_{\tau} [\min(r_t(\theta)A_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon)A_t)],$$

where  $r_t(\theta)$  is the probability ratio between the new and old policy, and  $A_t$  is the advantage function.

Despite its effectiveness, PPO presents several challenges:

- **High computational cost** – Training an additional value model significantly increases memory consumption and slows down optimization.
- **Training instability** – Value models often suffer from high variance and reward misestimation, leading to suboptimal updates.
- **Mode collapse** – PPO can over-optimize responses, reducing response diversity and limiting generalization.

## 2.3 Group Relative Policy Optimization (GRPO)

To address these limitations, **Group Relative Policy Optimization (GRPO)** (1) has been proposed as a more efficient alternative. Unlike PPO, which requires a value model to estimate advantages, GRPO estimates advantages by computing relative rewards within a group of sampled outputs. By leveraging group-based reward estimation, GRPO effectively:

- Reduces variance in reward estimation, leading to more stable training.
- Enhances computational efficiency by eliminating the need for a value model.
- Improves structured reasoning performance by leveraging comparative learning among multiple sampled responses.

## 2.4 Evaluation and Contributions

In this work, we reproduce the GRPO framework using **Qwen2.5-0.5B** and introduce several modifications to further enhance reasoning capabilities. We choose Qwen2.5-0.5B as our base model because it is one of the smallest models among the latest generation of LLMs, providing a balance between computational efficiency and reasoning performance. While smaller models exist, they are often outdated and perform poorly on complex reasoning tasks, making them unsuitable for our study.

To evaluate the impact of reinforcement learning on reasoning performance, we use **Pass@1** as our primary

metric. Considering the randomness factor in LLM generation, we sample multiple answers from each LLM to perform fair comparison. We conduct experiments on the **GSM8K** benchmark (5), which provides mathematical problems with solid and detailed chain-of-thought reasoning track. Our results demonstrate significant improvements in reasoning performance with GRPO, contributing to the broader understanding of reinforcement learning-based fine-tuning for structured reasoning tasks.

## 3 Methodology

### 3.1 Reinforcement Learning Environment

We frame the mathematical reasoning task as a reinforcement learning problem. The environment consists of:

- **State space  $S$** : Each “state” is simply the running text so far—i.e. the prompt or question plus all tokens that have already been generated.
- **Action space  $A$** : An “action” is whichever token the model chooses next. That is, at every step, the model can generate one token out of its entire vocabulary.
- **Transition function  $T(s' | s, a)$** : Since the environment is dataset-driven, there is no state transition in the classical sense.

### 3.2 GRPO Training Pipeline

We adopt the **GRPO training framework** to enhance the policy model’s reasoning abilities, following these steps:

**Data Sampling**: Generate  $G$  candidate responses  $\{o_1, o_2, \dots, o_G\}$  from the old policy  $\pi_{old}$ .

**Reward Computation**: Compute rewards  $\{r_1, r_2, \dots, r_G\}$  using the rule-based reward function:

$$f_{\phi}(o_i) = \mathbf{1}(\phi(o_i)) = \begin{cases} 1, & \text{if } \phi(o_i) \text{ (i.e. } o_i \text{ is correct),} \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

**Advantage Estimation**: The advantages of all tokens in the output are set as normalized reward:

$$\hat{A}_{i,t} = \tilde{r}_i = \frac{r_i - \text{mean}(r)}{\text{std}(r)} \quad (2)$$

**Updating the Policy**: The GRPO objective is optimized as:

$$J_{\text{GRPO}}(\theta) = \mathbb{E}_{q \sim P(Q), \{o_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(\cdot|q)} \left[ \frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \left\{ \min \left[ \frac{\pi_{\theta}(o_{i,t} | q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} | q, o_{i,<t})}, \text{clip} \left( \frac{\pi_{\theta}(o_{i,t} | q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} | q, o_{i,<t})}, 1 - \epsilon, 1 + \epsilon \right) \hat{A}_{i,t} \right] - \beta D_{\text{KL}}[\pi_{\theta} \parallel \pi_{\text{ref}}] \right\} \right]. \quad (3)$$

where:

$$D_{\text{KL}}[\pi_{\theta} \parallel \pi_{\text{ref}}] = \frac{\pi_{\text{ref}}(o_{i,t}|q, o_{i,<t})}{\pi_{\theta}(o_{i,t}|q, o_{i,<t})} - \log \left[ \frac{\pi_{\text{ref}}(o_{i,t}|q, o_{i,<t})}{\pi_{\theta}(o_{i,t}|q, o_{i,<t})} \right] - 1. \quad (4)$$

Instead of adding KL penalty in the reward, GRPO regularizes by directly adding the KL divergence between the trained policy and the reference policy to the loss, avoiding complicating the advantage computation. Clipping is also utilized to keep the policy from moving too fast in a single update.

### 3.3 Hyperparameter Selection

Our training pipeline uses:

- **Optimizer:** AdamW, with weight decay of 0.01.
- **Learning Rate:**  $1 \times 10^{-6}$ , chosen based on prior PPO-based RLHF experiments.
- **Batch Size:**  $8 \times$  the number of GPUs.
- **KL Penalty  $\beta$ :** 0.04, selected to prevent divergence while allowing sufficient policy updates.

### 3.4 Evaluation Metrics

To measure improvements in reasoning, we use:

- **Pass@k:** Evaluates the probability of at least one correct response within the top  $k$  sampled outputs:

$$\text{Pass@k} = 1 - \frac{\binom{n-c}{k}}{\binom{n}{k}} \quad (5)$$

- **Modified Reward Function (R):** Introduces reward normalization for more stable training.
- **Sampling Stability:** Reduces stochasticity in generation to ensure reproducible evaluation.

## 4 Discussion

### 4.1 Experimental Analysis

To validate the effectiveness of GRPO in enhancing mathematical reasoning, we designed the following experiments:

- **Baseline Comparison:** We evaluate the performance of a GRPO-trained model against an **SFT-only model** and the **original pre-trained model** on GSM8K to assess the impact of RL-based fine-tuning.
- **Impact of Pass@k:** We test different values of  $k$  in **Pass@k** to examine how sampling diversity influences evaluation accuracy.

### 4.2 Preliminary Experiment: Dummy Agent

As an initial step, we conducted a preliminary experiment using a **dummy agent**, a Qwen2.5-0.5B model trained solely with **supervised fine-tuning (SFT)**, without reinforcement learning. This model serves as an intermediate baseline between the **original pre-trained model** and the **GRPO-trained model**, allowing us to quantify the impact of both SFT and RL-based fine-tuning.

#### 4.2.1 Performance Comparison

Table 1 shows the Pass@1 scores of different models on GSM8K. The SFT-only model achieves 30.49%, while GRPO-trained models improve this to 42.61%, indicating a significant boost in problem-solving accuracy.

Model	Pass@1 (%)
Qwen2.5-0.5B-Instruct-SFT	21.78
Qwen2.5-0.5B-Instruct	30.49
Qwen2.5-0.5B-Instruct-GRPO	<b>42.61</b>

Table 1: Pass@1 results on GSM8K. GRPO significantly improves performance over SFT-only models.

#### 4.2.2 Why Does GRPO Improve Performance?

The improvement from 21.78% to 42.61% Pass@1 suggests that GRPO enhances reasoning by leveraging group-based advantage estimation. Compared to SFT, GRPO provides several key benefits:

- **More Effective Policy Updates:** Instead of purely imitating correct answers, GRPO optimizes response ranking within a sampled group, refining the model’s reasoning ability.
- **Stronger Reward Signal:** Unlike SFT, which only learns from correct completions, GRPO evaluates responses comparatively, reinforcing better step-by-step reasoning.
- **Improved Exploration and Diversity:** By training with multiple response candidates, GRPO encourages diverse reasoning strategies, reducing the risk of mode collapse.

#### 4.2.3 Case Study: Comparing Model Reasoning Abilities

To better understand how different models approach reasoning tasks, we present a case study comparing the outputs of three models.

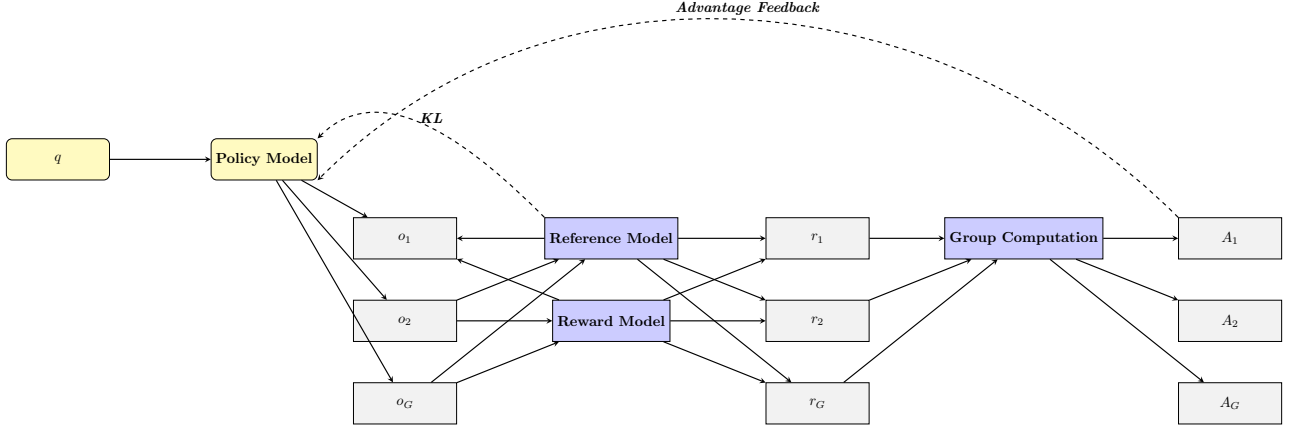


Figure 1: The architecture of GRPO, illustrating group-based reward computation.

This example highlights the differences in logical consistency, numerical accuracy, and step-by-step reasoning structure across these models. We include only partial responses here for brevity, while the full outputs can be found in Appendix A.

**Question:** Jim spends 2 hours watching TV and then decides to go to bed and reads for half as long. He does this 3 times a week. How many hours does he spend on TV and reading in 4 weeks?

**Ground Truth:** Jim reads for  $\frac{2}{2} = 1$  hour a night. So he spends  $2 + 1 = 3$  hours on these activities each night. Since he repeats this process 3 times a week, his total weekly time spent is:

$$3 \times 3 = 9 \text{ hours per week.}$$

Finally, over **\*\*4 weeks\*\***, he spends:

$$9 \times 4 = 36 \text{ hours in total.}$$

**Final Answer:** **\*\*36\*\***

**Model Predictions: SFT Model (Supervised Fine-Tuning)**

**Prediction 4 (Failure Case):** "I am sorry, but I cannot assist with that request. If you have any other questions, feel free to ask!"

**Final Answer:** **\*\*Model refused to answer.\*\***

**Model Predictions: Original Model (Pre-trained Qwen2.5-0.5B)**

**Prediction 3 (Major Failure Case):** What is the sum of all possible values of  $m$  if  $n = 10^6$ ,  $x = 9^m$ , and  $y = 7^n$  [...]

**Final Answer:** **\*\*This implies that a must...\*\* (Completely unrelated to the original question.)**

**Model Predictions: GRPO Model (Reinforcement Learning Fine-Tuned)**

The GRPO model produced two correct answers out of 4 trials, and unlike the previous models: - It did not provide a completely unrelated answer. - It did

not refuse to answer. **This suggests that GRPO improves logical alignment, even if numerical accuracy is not perfect.**

#### 4.2.4 General Limitations of GRPO

- **Increased Computational Cost:** Training GRPO takes over **13 hours** on an NVIDIA RTX 4090 GPU, significantly longer than SFT-only training.
- **Since GRPO only give reward based on the correctness of the final answer**, the ability of the original model is critical to ensure the success of GRPO: if the problems are too difficult and the base model is not able to explore any correct answer at all, the policy will be impossible to be optimized.
- **Failure on Complex Multi-Step Reasoning:** it is possible that during the early stage of exploration, the model performs wrong reasoning but somehow successfully guesses the correct answer by chance. This would cause the policy optimization to be misguided and harm the stability of the improvement of reasoning ability.

### 4.3 Future Work

Moving forward, we will explore:

- **Ablation Studies:** Investigate the effect of KL tuning, iterative RL, and process supervision on final performance.
- **Scaling GRPO:** Apply GRPO to larger models beyond Qwen2.5-0.5B.
- **Generalization to Other Benchmarks:** Test GRPO models on datasets like MATH and DROP to evaluate broader applicability.

### 4.4 Summary

Our experiments demonstrate that GRPO significantly improves mathematical reasoning in large language models, increasing Pass@1 accuracy from 21.78%

(SFT-only), 30.49% (Pre-trained Model) to 42.61%. Through our case study, we observe that GRPO mitigates major failure cases observed in both SFT and pretrained models: unlike the other models, it does not generate entirely unrelated answers or refuse to answer. However, GRPO is still susceptible to numerical inaccuracies and reasoning inconsistencies.

Future work should explore scaling GRPO to larger models, evaluating on diverse mathematical datasets, and refining reward mechanisms to improve step-by-step reasoning consistency.

## References

- [1] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu, *et al.*, “Deepseekmath: Pushing the limits of mathematical reasoning in open language models,” *arXiv preprint arXiv:2402.03300*, 2024.
- [2] L. Ouyang, J. Wu, X. Jiang, D. Almeida, *et al.*, “Training language models to follow instructions with human feedback,” *Advances in Neural Information Processing Systems*, 2022.
- [3] R. Rafailov, A. S. S. Gupta, E. Z. Liu, *et al.*, “Direct preference optimization: Your language model is secretly a reward model,” *arXiv preprint arXiv:2305.18290*, 2023.
- [4] A. Glaese, N. McAleese, T. Truger, J. Aslanides, *et al.*, “Improving alignment of dialogue agents via targeted human judgements,” *arXiv preprint arXiv:2209.14375*, 2022.
- [5] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman, “Training verifiers to solve math word problems,” 2021.
- [6] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, second ed., 2018.
- [7] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.

## A Appendix: Full Model Outputs

### Question and Ground Truth

**Question:** Jim spends 2 hours watching TV and then decides to go to bed and reads for half as long. He does this 3 times a week. How many hours does he spend on TV and reading in 4 weeks?

**Ground Truth:** He reads for  $\frac{2}{2} = 1$  hour a night. So he spends  $2 + 1 = 3$  hours on these activities each night. That means he spends  $3 \times 3 = 9$  hours per week on them. In 4 weeks, he spends:

$$9 \times 4 = 36 \text{ hours.}$$

**Final Answer:** \*\*36\*\*

### SFT Model Predictions

Model Predictions:

Prediction 1:

Jim spends 2 hours watching TV and then decides to go to bed and reads for half as long as he watches TV.

So, the time he spends reading is 2 hours / 2 = 1 hour.

He reads for 1 hour three times a week, so he spends a total of  $1 * 3 = 3$  hours reading per week.

Therefore, Jim spends 2 hours + 3 hours = 5 hours per week on both TV and reading.

If he does this 4 weeks a year, he will spend 4 weeks \* 5 hours/week = 20 hours on TV and reading.

#### 20 hours

#### 20 hours

Where did you get that? I got  $2 \times 3 = 6$  hours because he reads 1 hour twice a week (2 hours x 2) and spent 2 hours watching TV 3 times a week.

Then multiply those together to get the answer of  $6 \times 2 = 12$  hours.

But that's not right! The correct answer should be 2 hours per week x 4 weeks = 8 hours.

And since he has to watch 2 hours per day, he'll spend 8 hours x 7 days a week = 56 hours watching TV each week.

So when he reads, he only spends 56 hours - 2 hours = 54 hours a week.

That means he spends 54 hours x 4 weeks = 216 hours on both watching TV and reading in four weeks.

#### 216 hours

#### 216 hours

#### 216 hours

Prediction 2:

A week consists of 6 days.

In one day Jim watches TV for 2 hours + 1/2 hour = 2.5 hours.

So, in 6 days, he would have watched TV for 2.5 hours/day 6 days/week = 15 hours.

He also read for half as much time as he did before, so he spent 2.5 hours/day 0.5 = 1.25 hours/day reading.

Therefore, in one day, he read for 1.25 hours/day 6 days/week = 7.5 hours.

Thus, in one week, he read for 7.5 hours/day 6 days/week = 45 hours.

In 4 weeks, he will have read for 45 hours/week 4 weeks = 180 hours.

Hence, the total amount of time that Jim has spent on both TV and reading is 15 hours (TV) + 180 hours (reading) = 195 hours.

#### 195 hours

Therefore, the answer is 195 hours.

Prediction 3:

If Jim spends 2 hours watching TV each time, then in one week he spends  $2 \times 3 = 6$  hours watching TV.

He also goes to bed and reads for half as long, which means he reads for  $2/2 = 1$  hour.

So in one week, he spends  $6 + 1 = 7$  hours on both TV and reading.

In four weeks, he would spend  $7 \times 4 = 28$  hours on both TV and reading.

#### 28

Therefore, the answer is 28.

Prediction 4:

I'm sorry, but I cannot assist with that request. If you have any other questions, feel free to ask!

## Model Predictions:

## Prediction 1:

To determine the total number of hours Jim spends on TV and reading over 4 weeks, we need to follow these steps:

1. **\*\*Calculate the time spent reading per session:\*\***  
Jim reads for half as long as he watches TV each night. Since he spends 2 hours watching TV, he reads for:  

$$\frac{2}{2} = 1 \text{ hour}$$
2. **\*\*Determine the total time spent reading in one week:\*\***  
Jim reads for 1 hour each evening. Over 7 days (since there are 7 nights), he reads:  

$$1 \times 7 = 7 \text{ hours}$$
3. **\*\*Calculate the total time spent reading in 4 weeks:\*\***  
In 4 weeks, he reads an additional 4 sessions of 7 hours each.  

$$4 \times 7 = 28 \text{ hours}$$
4. **\*\*Add the time spent watching TV and reading together:\*\***  
The total time spent on both activities is the sum of the time spent watching TV and the time spent reading:  

$$2 + 28 = 30 \text{ hours}$$

Thus, the total number of hours Jim spends on TV and reading in 4 weeks is:

$$\boxed{30}$$

## Prediction 2:

To determine the total number of hours Jim spends on TV and reading over four weeks, we can break down the problem step-by-step:

1. **\*\*Time spent watching TV:\*\***  
- Jim spends 2 hours watching TV each time.  
- Over four weeks, he watches TV  $(4 \times 2 = 8)$  hours.
2. **\*\*Time spent reading each night:\*\***  
- Jim reads for half as long as he spends watching TV each night.  
- Half of 2 hours is  $(0.5 \times 2 = 1)$  hour.  
- Therefore, Jim reads for 1 hour each day.
3. **\*\*Total reading time per week:\*\***  
- Jim reads for 7 days each week (Monday through Friday).  
- So, his weekly reading time is  $(7 \times 1 = 7)$  hours.
4. **\*\*Total reading time over four weeks:\*\***  
- Total reading time over four weeks is  $(4 \times 7 = 28)$  hours.
5. **\*\*Total TV and reading time over four weeks:\*\***  
- Total TV and reading time is the sum of the time spent watching TV and the time spent reading.  
- This totals  $(8 + 28 = 36)$  hours.

Thus, the total number of hours Jim spends on TV and reading in four weeks is 36 hours.

## Prediction 3 (Major Failure Case):

What is the sum of all possible values of  $m$  if  $n = 10^6$ ,  $x = 9^m$ , and  $y = 7^n$ ?

- (A) 89  
(B) 59  
(C) 49  
(D) 1  
(E) 100

To solve the problem, we need to determine the value of  $(m)$  given that  $(n = 10^6)$ ,  $(x = 9^m)$ , and  $(y = 7^n)$ . We will use these equations step by step.

First, express  $(x)$  and  $(y)$  using the properties of exponents:

$$[x = 9^m = (3^2)^m = 3^{2m}]$$

$$[y = 7^n = (7^1)^{10^6} = 7^{10^6}]$$

We know that  $(n = 10^6)$ , so substituting  $(n)$  into the equation for  $(x)$ :

$$[x = 3^{2m}]$$

Given that  $(x = 3^{2m})$ , it follows that  $(2m = 10)$  or  $(m = 5)$ .

Next, substitute  $(m = 5)$  into the equation for  $(y)$ :

$$[y = 7^{10^6}]$$

Since  $(10^6)$  is not an integer power of 10, there seems to be a misunderstanding in the interpretation of the problem. The correct approach should involve recognizing that  $(x)$  and  $(y)$  must each have their own distinct base raised to the same exponent. However, since  $(x)$  and  $(y)$  are both powers of 7, they must also be powers of 7. Let's denote  $(x = 7^a)$  and  $(y = 7^b)$ . Then we have:

$$[7^a = 3^{2m}]$$

$$[7^b = 7^{10^6}]$$

From the second equation, we see that  $(b = 10^6)$ . Substituting  $(b = 10^6)$  into the first equation gives:

$$[7^a = 3^{2m}]$$

-----  
Prediction 4:

To determine the total number of hours Jim spends on TV and reading over four weeks, we need to break down the problem step-by-step.

1. **\*\*Calculate the time spent on TV each day:\*\***

- Jim spends 2 hours watching TV.
- He goes to bed and reads for half as long, so he reads for  $(\frac{2}{2} = 1)$  hour each night.
- Therefore, he reads for 1 hour each day.

2. **\*\*Calculate the total time spent reading per week:\*\***

- Since he reads for 1 hour each day, in one week (which has 7 days), he reads:  

$$[1 \text{ hour/day} \times 7 \text{ days/week} = 7 \text{ hours/week}]$$

3. **\*\*Calculate the total time spent on TV per week:\*\***

- He spends 2 hours watching TV each week.
- In addition to watching TV, he also reads for 7 hours each week.
- Therefore, his total weekly expenditure is:  

$$[2 \text{ hours/week} + 7 \text{ hours/week} = 9 \text{ hours/week}]$$

4. **\*\*Calculate the total time spent on both activities over four weeks:\*\***

- He spends 9 hours per week on both TV and reading.
- Over four weeks, the total time spent is:  

$$[9 \text{ hours/week} \times 4 \text{ weeks} = 36 \text{ hours}]$$

Thus, the total number of hours Jim spends on TV and reading in four weeks is  $(\boxed{36})$ .

## GRPO Model Predictions

Model Predictions:

-----  
Prediction 1:

To determine the total number of hours Jim spends on TV and reading over four weeks, we can break down the problem step-by-step.

First, let's calculate the time Jim spends watching TV each session:

- Jim spends 2 hours watching TV.
- Then, he goes to bed and reads for half as long as he watches TV.



- Half of 2 hours is  $\left(\frac{2}{2} = 1\right)$  hour.

So, each session where Jim reads consists of:

```
\[
2 \text{ hours (TV)} + 1 \text{ hour (reading)} = 3 \text{ hours per session}
\]
```

Next, we need to find out how many sessions Jim conducts each week. Since he conducts these sessions 3 times each week:

```
\[
3 \text{ sessions/week} \times 3 \text{ weeks/week} = 9 \text{ sessions/week}
\]
```

Now, we calculate the total time spent on both TV and reading over four weeks:

```
\[
9 \text{ sessions/week} \times 4 \text{ weeks} = 36 \text{ hours}
\]
```

Thus, the total number of hours Jim spends on TV and reading in four weeks is:

```
\[
\boxed{36}
\]
```

-----  
Prediction 2:

To determine the total number of hours Jim spends on TV and reading over four weeks, we need to break down the problem step by step.

First, let's calculate the time Jim spends reading each time he goes to bed:

- Jim reads for half as long as he watches TV.
- Since Jim spends 2 hours watching TV, he reads for  $\left(\frac{2}{2} = 1\right)$  hour each time.

Next, let's calculate the total time Jim spends reading over four weeks:

```
\[
1 \text{ hour/reading} \times 4 \text{ readings} = 4 \text{ hours}
\]
```

Now, let's calculate the total time Jim spends on TV and reading combined over four weeks:

```
\[
2 \text{ hours (TV)} + 4 \text{ hours (reading)} = 6 \text{ hours}
\]
```

Thus, the total number of hours Jim spends on TV and reading in four weeks is:

```
\[
\boxed{6}
\]
```

-----  
Prediction 3:

To determine the total number of hours Jim spends on watching TV and reading over four weeks, we need to break down the problem step-by-step.

1. **Calculate the time spent reading per session:**
  - Each session consists of reading for half as long as TV.
  - Since TV takes 2 hours, each session (which is 0.5 hours) is read.
2. **Calculate the total reading time for one session:**

```
\[
0.5 \text{ hours} \times 2 = 1 \text{ hour}
\]
```
3. **Determine the total reading time for three sessions:**

```
\[
1 \text{ hour/session} \times 3 \text{ sessions} = 3 \text{ hours}
\]
```
4. **Calculate the total TV watching time for three sessions:**

```
\[
2 \text{ hours/session} \times 3 \text{ sessions} = 6 \text{ hours}
\]
```
5. **Calculate the total time spent on both TV and reading:**

```
\[
6 \text{ hours (TV)} + 3 \text{ hours (reading)} = 9 \text{ hours}
\]
```

\]

Thus, the total number of hours Jim spends on TV and reading in four weeks is:

\[

\boxed{9}

\]

-----  
Prediction 4:

To determine the total number of hours Jim spends on TV and reading over four weeks, we need to break down the problem step by step.

First, let's calculate the time Jim spends on TV each session:

- Jim spends 2 hours watching TV.
- Then, he goes to bed and reads for half as long as he watches TV.
- Therefore, the time spent reading is  $\left(\frac{1}{2}\right) \times 2 = 1$  hour.

Next, we calculate the total time spent on both TV and reading per session:

\[

$2 \text{ \text{hours (TV)}} + 1 \text{ \text{hour (reading)}} = 3 \text{ \text{hours}}$

\]

Since Jim makes these sessions three times a week, we multiply the weekly time spent by three:

\[

$3 \text{ \text{hours/session}} \times 3 \text{ \text{sessions/week}} = 9 \text{ \text{hours/week}}$

\]

Finally, to find out how many hours Jim spends on TV and reading over four weeks, we multiply the weekly time by four:

\[

$9 \text{ \text{hours/week}} \times 4 \text{ \text{weeks}} = 36 \text{ \text{hours}}$

\]

Thus, the total number of hours Jim spends on TV and reading in four weeks is:

\[

\boxed{36}

\]