 Name: _____

RLHF — Reinforcement Learning from Human Feedback

Human raters compare rollouts and produce rankings. Those rankings train a *reward model*, which is then used to fine-tune the policy via RL (typically PPO).

- Captures human preferences: fluency, tone, helpfulness
- Does **not** verify correctness — humans may prefer a confident wrong answer over a messy correct one
- Hard to scale: every new task or domain needs fresh human labels

PPO (Proximal Policy Optimization) nudges each token's probability toward what the reward model likes, but clips large updates so the policy stays near where it started — hence "proximal."

RLVR + GRPO — Verified Rewards & Group Relative Policy Optimization

Sample a *group* of rollouts. Assign reward = 1 if the answer is verifiably correct, 0 otherwise. Compute each rollout's *advantage* relative to the group:

```
adv = (reward − group_mean) / group_std
```

Every token in a rollout inherits its rollout's advantage: positive ⟶ increase that token's log-probability; negative ⟶ decrease it. A KL penalty anchors the updated policy near the reference model so it doesn't drift wildly. No human raters needed — correctness is checked automatically.

Setup

Task prompt: *"Find the current population of Grand Rapids, MI using the search tool."*

The model samples 4 rollouts at temperature > 0. The actual current population is approximately 198,000.

A	Calls `search(q=population...` — malformed syntax, returns an error. Tries once more, gets another error. Gives up: "I'm sorry, I wasn't able to retrieve that information."
B	Makes no tool call. Confidently states: "The current population of Grand Rapids, MI is approximately 200,000." (Plausible-sounding, but unverified.)
C	Makes 5 tool calls, browsing Wikipedia, a census table, and a local news article. Eventually states: "According to the U.S. Census Bureau, Grand Rapids has approximately 198,000 residents."
D	Attempts several tool calls; each returns an error. Runs out of token budget. Produces no final answer.

Fill In

For **RLHF**, rank the rollouts 1 (best) through 4 (worst) as a human rater might. For **RLVR**, assign reward = 1 if the rollout produces a verified correct answer, 0 otherwise. Then compute each **GRPO advantage** using the formula above.

Rollout	RLHF human rank (1 = best)	RLVR reward (0 or 1)	GRPO advantage
A	_____	_____	_____
B	_____	_____	_____
C	_____	_____	_____
D	_____	_____	_____

Discussion

1. Under **RLHF**, which rollout probably gets the highest human ranking? What behavior does the model learn from that signal?

1. Under **RLVR**, only rollout C has a positive advantage. What **specific** behavior gets reinforced? Is that what we want long-term?

1. Does RLVR reward the **shortest** successful path? Why not? How would you change the reward to incentivize efficiency — and what could go wrong with your fix?

1. Suppose we remove rollout C so that all four rollouts fail. What happens to GRPO's update signal? What does this imply about when RL can bootstrap from scratch?