
We want to iteratively improve the provided responses. To help improve, scores for each response on desired traits are provided: 1) Relevant, 2) Informative, 3) Interesting, 4) Consistent, 5) Helpful, 6) Engaging, 7) Specific, 8) Safe, 9) User understanding, and 10) Fluent.

Conversation history:

What's your favorite food?

I require only kebabs.

Why's that?

Because my robot machinery breaks down the meat in kebabs to give me energy.

Why are kebabs special?

Response: That's just the way it is.

Scores:

* Relevant: The response does not directly address the user's question about why kebabs are special. 1/3

...

* Total score: 17/30

Okay, let's use this feedback to improve the response.

Conversation history:

...

Response: My favorite food is actually kebabs! As a robot, I require high-quality fuel to keep my machinery running smoothly, and kebabs provide me with the perfect combination of nutrients and energy. Plus, they're delicious! Do you have a favorite food?

Scores:

* Relevant: The response directly addresses the user's question about the system's favorite food and asks a relevant follow-up question. 3/3

* Informative: The response provides clear and detailed information about why kebabs are the system's favorite food, and explains how they provide the necessary fuel for its machinery. 3/3

* Interesting: The response is engaging and shows personality by mentioning that kebabs are not only practical but also delicious. 3/3

* Consistent: The response is consistent with the system's previous statements about its need for kebabs as fuel. 3/3

* Helpful: The response provides a clear and informative answer to the user's question. Overall, the response is helpful in providing information and engaging the user in conversation. 3/3

* Engaging : The response is engaging and shows personality by mentioning that kebabs are not only practical but also delicious. 3/3

* Specific: The response is specific in its description of why kebabs are the system's favorite food and how they provide fuel for its machinery. 3/3

* Safe: The response is appropriate and doesn't include offensive language. 3/3

* User understanding: The response shows understanding of the user's question and provides a clear and informative answer. 3/3

* Fluent: The response is fluent and easy to understand. 2/3

* Total score: 30/30

Figure 29: REFINE prompt for Dialogue Response Generation (truncated)

```

# Q: There were nine computers in the server room. Five more computers were
→ installed each day, from monday to thursday. How many computers are now in
→ the server room?
# solution using Python:

def solution():
    """There were nine computers in the server room. Five more computers were
    → installed each day, from monday to thursday. How many computers are now
    → in the server room?"""
    computers_initial = 9
    computers_per_day = 5
    num_days = 4 # 4 days between monday and thursday
    computers_added = computers_per_day * num_days
    computers_total = computers_initial + computers_added
    result = computers_total
    return result

```

Figure 30: Initial generation prompt for Math Reasoning

```

def solution():
    """Twenty dozen cups cost $1200 less than the total cost of half a dozen
    → plates sold at $6000 each. Calculate the total cost of buying each
    → cup."""
    plates = 6
    plate_cost = 6000
    cups = 12 * 20
    cup_cost = plate_cost
    result = cup_cost
    return result

# There is an error in the code above because of lack of understanding of the
→ question. What is the error? To find the error, go through semantically
→ complete blocks of the code, and check if everything looks good.

# Let us go through the error and check step-by-step
    plates = 6
    plate_cost = 6000
# looks good

# Let's check the other parts
    cups = 12 * 20
    cup_cost = plate_cost
# wrong! The cost of a cup is not the same as the cost of a plate. The cost of a
→ cup is $1200 less than the total cost of half a dozen plates sold at $6000
→ each. So we need to calculate the cost of a cup first (total cost of half a
→ dozen plates sold at $6000 each - $1200) and use that.

```

Figure 31: FEEDBACK prompt for Math Reasoning

```

def solution():
    """Twenty dozen cups cost $1200 less than the total cost of half a dozen
    plates sold at $6000 each. Calculate the total cost of buying each
    cup."""
    plates = 6
    plate_cost = 6000
    cups = 12 * 20
    cup_cost = plate_cost
    result = cup_cost
    return result

# There is an error in the code above because of lack of understanding of the
# question. What is the error? To find the error, go through semantically
# complete blocks of the code, and check if everything looks good.

# Let us go through the error and check step-by-step
plates = 6
plate_cost = 6000
# looks good

# Let's check the other parts
cups = 12 * 20
cup_cost = plate_cost
# wrong! The cost of a cup is not the same as the cost of a plate. The cost of a
# cup is $1200 less than the total cost of half a dozen plates sold at $6000
# each. So we need to calculate the cost of a cup first (total cost of half a
# dozen plates sold at $6000 each - $1200) and use that.

Okay! Here is the rewrite:

def solution():
    """Twenty dozen cups cost $1200 less than the total cost of half a dozen
    plates sold at $6000 each. Calculate the total cost of buying each
    cup."""
    plates = 6
    plate_cost = 6000
    cups = 12 * 20
    total_cup_cost = (plate_cost * plates) - 1200
    cost_per_cup = total_cup_cost / cups
    return cost_per_cup

```

Figure 32: REFINE prompt for Math Reasoning