Synthesis of Best Practices in Prompt Engineering

Clarity and Context: Effective prompts are **clear, explicit, and provide context**. Ambiguity leads to irrelevant or incorrect answers, so good prompts precisely specify the task and any necessary background. Research shows that adding detailed descriptions of the task and context significantly improves model performance (Prompt Engineering a Prompt Engineer - Microsoft Research). For example, *setting the scene or role* for the model ("You are a financial expert…") can focus the response style. Similarly, stating the intended audience or difficulty level helps the model tailor its answer (e.g. noting *"the audience is an expert in the field"* yields more advanced detail (26 principles for prompt engineering to increase LLM accuracy 57%)). The more specific and well-scoped the prompt, the more effectively the model can "simulate" the desired response ([2312.16171] Principled Instructions Are All You Need for Questioning LLaMA-1/2, GPT-3.5/4). Always **identify what you need from the model** – be it an explanation, a solution, a recommendation – and state that clearly.

Step-by-Step Reasoning (Chain-of-Thought): Encourage the model to **think through problems stepwise**. Large language models can perform complex reasoning if guided to break down the task ([2205.11916] Large Language Models are Zero-Shot Reasoners). A landmark finding by Kojima et al. (2022) showed that simply appending *"Let's think step by step."* to a question prompts the model to generate intermediate reasoning steps, more than *4x increasing* accuracy on challenging reasoning tasks ([2205.11916] Large Language Models are Zero-Shot Reasoners). This **Chain-of-Thought (CoT)** prompting works for math, logic, and multi-hop questions by making the solution process explicit. Even without provided examples, a zero-shot CoT prompt can unlock hidden reasoning capabilities. The general principle is to **prompt critical thinking**: for instance, ask the model to list assumptions, consider alternatives, or systematically justify each step. By having the model articulate its reasoning or "show its work," you reduce errors and increase correctness ([2205.11916] Large Language Models are Zero-Shot Reasoners). This approach narrows the *compositionality gap* – the tendency to fail at combining multiple facts – by forcing the model to reason through each part ([2210.03350] Measuring and Narrowing the Compositionality Gap in Language Models).

Structured Retrieval and Fact-Finding: When a query requires multiple pieces of information or external knowledge, use prompts that **decompose the problem and retrieve facts**. One effective framework is *self-ask prompting*, where the model is guided to ask itself follow-up questions and find answers before giving a final response

([2210.03350] Measuring and Narrowing the Compositionality Gap in Language Models). Press et al. (2023) demonstrated that a *structured prompt* which explicitly has the model generate sub-questions and investigate them (even consulting a search engine) yields higher accuracy on complex questions ([2210.03350] Measuring and Narrowing the Compositionality Gap in Language Models). In practice, you can prompt an LLM to break a query into sub-parts or to *"search for relevant information on X"*. If the platform supports tools or browsing, instruct the model to use them. Otherwise, prompts can simulate retrieval by asking the model to recall known facts or enumerate what *data is needed*. Techniques like the **Cognitive Verifier pattern** formalize this: the LLM first produces a series of sub-questions that cover the problem's facets, answers each, and then synthesizes the overall answer ([2302.11382] A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT). This ensures comprehensive coverage and can mimic a structured research process. **Fact-checking prompts** are also valuable: e.g. asking *"List the facts supporting your answer and their sources"* or instructing the model to verify each key fact. Incorporating such verification steps helps reduce hallucinations and increase factual accuracy ([2302.11382] A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT).

Specificity and Constraints: The prompt should **specify the desired output format, style, or any constraints** on the answer. If you need a list, a step-by-step guide, a JSON output, or a concise answer, tell the model explicitly. For example, *"Provide the answer in bullet points."* If a certain structure is expected (like an essay with introduction and conclusion, or using the IRAC format for legal analysis), mention that. In fact, aligning the prompt with domain-specific frameworks can greatly boost performance. A recent study on legal reasoning found that the best results came from prompts using the **IRAC structure (Issue, Rule, Application, Conclusion)**, surpassing even basic chain-of-thought prompts (Exploring the Effectiveness of Prompt Engineering for Legal Reasoning Tasks - ACL Anthology). This underscores that **tailoring the prompt to the domain's logic** (be it legal IRAC, medical SOAP notes, scientific IMRaD, etc.) is a best practice for complex tasks.

Role and Persona: Defining a **role** for the AI can guide tone and knowledge level. The *Persona Pattern* in prompt engineering involves prefixing the prompt with a role description (e.g. *"You are a helpful librarian with expertise in history..."*). This can constrain the style of responses usefully ([2302.11382] A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT). Roles can be functional ("Act as a translator…") or stylistic ("Adopt the tone of a friendly mentor…"). By **simulating an**

identity, the model's output can be more consistent and contextually appropriate. This technique is widely applicable across domains – whether you need the model to behave like a critical reviewer, a knowledgeable tutor, or a polite customer service agent.

Iterative Refinement and Self-Correction: Don't expect a perfect answer in one shot. It's often effective to use an **iterative prompt strategy**: first ask for a draft or analysis, then refine. For instance, you might prompt: *"Give an initial answer, then critique it and improve it."* One recent approach, **Self-Refine**, has the model generate an answer, then reflect on its own output and refine it based on that feedback (Self-Refine: Iterative Refinement with Self-Feedback for LLMs). This process can be done in a single prompt (by instructing the model to include a self-critique before finalizing) or in multi-turn interactions. Research has shown that Self-Refine **significantly boosts performance** on tasks like code generation and even sentiment analysis by iteratively polishing the response (Self-Refine: Iterative Refinement with Self-Feedback for LLMs). Another refinement tactic is the **Reflection pattern** – prompting the model to *check its answer for errors or possible improvements* ([2302.11382] A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT). This can catch mistakes or omissions that the initial pass missed. In practice, after getting an answer, you can follow up with a prompt like: *"Identify any errors in the above solution and correct them."* The model's ability to critique and fix its output often yields a more accurate and polished result.

Prompt Structure and Length: The **structure** of your prompt matters. Longer prompts can supply more detail and guidance, but there is a balance – too verbose can introduce confusion or unnecessary constraints. A recent set of *26 prompting principles* by Bsharat et al. (2023) highlights some structural tips: use delimiters (like triple backquotes or XML tags) to clearly separate sections of your prompt (instructions, context, examples) (26 principles for prompt engineering to increase LLM accuracy 57%) (26 principles for prompt engineering to increase LLM accuracy 57%). They also suggest using explicit labels like "**Instruction:**" and "**Question:**" in complex prompts to distinguish the task description from the user query (26 principles for prompt engineering to increase LLM accuracy 57%). These formatting cues can help the model parse your prompt correctly. Another finding is that using **direct, affirmative language** yields better compliance: phrasing as "Do X" instead of "Don't do Y" avoids confusion (26 principles for prompt engineering to increase LLM accuracy 57%). It's also usually unnecessary to add polite fluff ("please," "thank you") – models don't require it (26 principles for prompt engineering to increase LLM accuracy 57%). In fact, overly polite or indirect phrasing can dilute clarity. Instead, be **concise and literal** about what you want.

Few-Shot Examples: When possible, giving **examples** in the prompt can hugely help (this is called few-shot prompting). For instance, if you want the model to follow a specific format, show it a small sample input-output pair within the prompt. Few-shot exemplars can lock in a pattern, especially for tasks like math problem solving, Q&A, or classification. Even 1-2 examples can guide the model better than description alone, as it demonstrates the desired behavior. However, examples should be relevant and diverse – avoid ones that are too similar to each other or to the test query, to prevent the model from just mimicking superficially. If no good example is available, sometimes a well-chosen *generic* example can still help ("Q: [some format] A: [some format]"). Combining chain-of-thought with few-shot has been shown to be effective for complex tasks (26 principles for prompt engineering to increase LLM accuracy 57%) – e.g. provide a few examples where the reasoning is spelled out, to encourage the model to do the same.

Innovative Prompting Tricks: Prompt engineering is still evolving, and researchers have found some creative tricks to boost performance. For example, one study suggests that **simulating stakes** for the AI can affect its output quality – telling the model *"You will be penalized if your answer is incorrect"* or even offering an imaginary reward *"I'm going to tip $100 for a perfect answer"* has surprisingly led to more accurate and detailed responses in experiments (26 principles for prompt engineering to increase LLM accuracy 57%) (26 principles for prompt engineering to increase LLM accuracy 57%). The theory is that such statements prompt the model to **prioritize accuracy** (since it has been trained on text where such cues appear). While these hacks are intriguing, they may not always work and could produce overly cautious answers. Use them sparingly and ethically. On the other hand, straightforward instructions like *"Ensure your answer is unbiased and free of stereotypes."* are a good practice to include for sensitive topics (26 principles for prompt engineering to increase LLM accuracy 57%) – they remind the model of ethical considerations and often do reflect in the output tone.

Frameworks and Guidelines: Several frameworks exist to guide prompt engineering systematically. For instance, the **Prompt Pattern Catalog** by White et al. (2023) organizes common techniques into categories like *Output Customization* (e.g. Template prompts, Persona prompts), *Error Identification* (e.g. Fact-check list, Reflection), *Prompt Improvement* (e.g. Question refinement, Cognitive verifier), and so on ([2302.11382] A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT) ([2302.11382] A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT). These patterns are "recipes" that can be combined to craft a complex prompt. Another approach, from Microsoft Research, is **PE2 (Prompt Engineering a Prompt Engineer)**, which is

essentially a meta-prompt that guides an LLM to develop better prompts on its own (Prompt Engineering a Prompt Engineer - Microsoft Research). PE2 emphasizes including a step-by-step reasoning template in the prompt itself, plus very detailed instructions and context – essentially baking the best practices we discussed *into* a single master prompt. All these efforts underline that **prompting is a skill** requiring both creativity and analytical thinking. By following the principles of clarity, context, step-by-step reasoning, structured retrieval, and iterative refinement, you can greatly **optimize prompts for better performance** in search and beyond.

Generic High-Utility Prompt Examples (Organized by Theme)

Below is a list of versatile, high-utility prompts that can be adapted across domains. They are grouped by theme, and each includes an example prompt (in *italics*) along with an explanation of how it helps. These prompts embody the best practices above – such as encouraging critical thinking, guiding structured retrieval of information, or refining the style and persuasiveness of responses.

Critical Thinking & Reasoning Prompts

- **Step-by-Step Reasoning:** *"Let's think step by step."* – Triggers the model to generate a chain of thought before final answers. This simple cue has been shown to significantly improve accuracy on complex problems by having the AI break down the reasoning process ([2205.11916] Large Language Models are Zero-Shot Reasoners). Use this after a question to encourage analytical, methodical answers.
- **Why & How Explanation:** *"Explain why each step or decision is taken."* – Instructs the model to justify its reasoning at each stage. This prompt forces a deeper analysis and helps expose the logic behind answers, improving transparency and quality of the solution.
- **Explore Alternatives:** *"What are alternative approaches to this problem, and which is best?"* – Prompts the model to not tunnel on one solution. By considering multiple approaches or answers, the AI engages in comparative reasoning, which is useful for complex decision-making or creative brainstorming. It reduces the chance of missing a better solution.
- **Assumptions Check:** *"List any assumptions you are making and check if they are valid."* – Has the model explicitly surface its assumptions. This critical-thinking

prompt helps in tasks where hidden assumptions could lead to wrong answers (e.g., problem-solving, legal reasoning). By validating or questioning each assumption, the model can correct its course if needed.

- **Devil's Advocate / Counterargument:** *"What might be the counterarguments or weaknesses in the above answer?"* – Encourages the AI to critically evaluate an answer (whether one it just gave or a provided statement). This is useful in argumentative or analytical writing: the model will identify potential flaws or opposite viewpoints, leading to a more robust final answer. It instills a habit of self-critique akin to a debate, strengthening critical analysis.
- **Evidence and Fact Check:** *"Provide 3 pieces of evidence or facts to support your answer, and verify each."* – Directs the model to back up its statements with evidence. The model will try to recall factual supporting details (dates, statistics, references) and double-check consistency. This not only makes answers more convincing, it helps reduce hallucinations by focusing on verifiable specifics. (If the AI states a fact, it's more likely to be true if you explicitly asked for evidence.)
- **Socratic Questioning:** *"Continually ask yourself: 'Does this make sense and why?' as you solve the problem."* – This meta-prompt has the model adopt a reflective stance, persistently scrutinizing the logic of its own answer. It yields an introspective solution where each step is justified (similar to how a tutor might probe a student's reasoning). The outcome is often a well-argued answer with strong internal consistency.

Structured Retrieval & Fact-Finding Prompts

- **Problem Decomposition:** *"Break the question into smaller sub-questions first."* – Tells the model to perform query decomposition. This approach is powerful for complex or multi-part questions: the AI will identify the key subtopics or steps needed. By answering each sub-question, it effectively builds the components of the final answer (mimicking an outline or research plan) ([2302.11382] A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT). It ensures no aspect of a multifaceted query is overlooked.
- **Ask for Clarification:** *"Ask me any clarifying questions you need before answering."* – Permits the model to engage the user if information is ambiguous or missing. While the AI typically doesn't ask questions on its own, this prompt explicitly gives it license to do so. The model might request more specifics, which

leads to a more accurate and tailored answer. This is useful in interactive settings or when you suspect the query might be underspecified.

- **External Knowledge Search:** *"If needed, search for relevant information on [Topic] and summarize it before answering."* – Although vanilla LLMs can't actually browse the web without tools, this prompt cues the model to emulate a research step. It may respond with something like: "(Searching for X…) I found that…". In tool-enabled environments, this prompt encourages the use of the tool (e.g. a web search plugin). In standalone mode, it makes the model more likely to bring in pertinent facts it "knows" related to the query. Essentially, it nudges the AI to pull in outside knowledge systematically.
- **Cite Sources (Retrieval Confidence):** *"Include references to any sources or facts you mention."* – Instructs the model to produce source attributions. The AI might cite books, papers, or websites from its training data as the sources of facts. Even if the citations aren't perfect, this habit forces the model to be more factual (since it's trying to recall where a fact came from). In a closed-book setting it might say, for example, "According to a 2021 CDC report, …". This is useful for tasks requiring high factual accuracy or where you plan to verify information after.
- **Fact Check List:** *"Before finalizing, list the key facts in your answer that should be verified."* – This prompt is derived from the *Fact Check pattern* ([2302.11382] A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT). It has the model step back and enumerate the claims it made. By listing them, the model (or the user) can then scrutinize each for accuracy. It's a way to inject a last-minute verification stage. The final answer can either include the list or the model may internally check and correct any dubious facts it listed.
- **Context Manager (Provide Relevant Context):** *"Here is some background info: [insert context]. Use this context to answer the question."* – When you have relevant text or data, include it in the prompt and instruct the model to use it. Large models are very good at weaving provided context into their answers (this is essentially how Retrieval-Augmented Generation works). By explicitly saying the above and providing a chunk of text or bullet points of facts, you guide the model to rely on that instead of hallucinating or using only general knowledge. It's a prompt pattern that ensures **context control**, keeping the answer grounded in the given information.

- **Query Refinement:** *"Rephrase the question in a more precise way before answering."* – This uses the **Question Refinement** pattern ([2302.11382] A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT). The model will first restate or improve the query, which often helps clarify intent. After rephrasing (sometimes the model will literally output a new question), it will proceed to answer that. This is valuable for vague questions: the act of re-writing the query can resolve ambiguity and ensure the answer aligns with what's actually needed. It's akin to the model making sure it understood the question correctly by articulating it in its own words.

Persuasive & Refined Response Prompts

- **Audience Tuning (Explain Like I'm 5 / Expert):** *"Explain the concept in simple terms, as if I'm a beginner."* – A classic prompt to adjust the answer's complexity and tone. Telling the model to explain *"like I'm 5"* will simplify the language and use basic analogies, which is great for understanding or broad audiences (26 principles for prompt engineering to increase LLM accuracy 57%). Conversely, "the audience is an expert" or "use technical terms" will do the opposite (26 principles for prompt engineering to increase LLM accuracy 57%). This prompt ensures the **response is pitched at the right level** of detail and persuasiveness for the target reader. It's broadly useful from teaching scenarios to executive summaries.
- **Tone and Style Directive:** *"Respond in a formal and persuasive tone, using compelling evidence and rhetorical devices."* – Guides the model's writing style. You can swap in any tone (formal, casual, humorous, academic, empathetic, etc.) and any style elements needed (e.g. "use metaphors to illustrate points" or "be concise and authoritative"). This direct specification helps in scenarios where the manner of delivery is as important as the content – for example, drafting a persuasive cover letter versus a scientific abstract will require very different tone, and a single phrase in the prompt can signal that.
- **Thesis and Conclusion Structure:** *"Start with a clear thesis statement, then provide supporting arguments, and end with a strong conclusion."* – Instructs the model to adopt a persuasive essay structure. This is useful for any argumentation or position piece. The AI will present a thesis (central claim), follow with organized supporting points (each potentially with evidence or examples if it knows any), and then wrap up conclusively. It ensures the answer isn't just a flat list of points but is structured to persuade or inform coherently.

- **Refinement Request:** *"Now refine the above answer to make it more concise and impactful."* – This is used in a multi-turn setting: after an initial answer, you ask for improvement with specific goals (conciseness, impact, clarity, etc.). The model will edit its previous response, often yielding a tighter and more polished version. This iterative refinement prompt is very practical – you act as an editor guiding the AI. You can replace the adjectives as needed (e.g. "more formal", "more optimistic in tone", "more detailed", "less technical", etc.). It's effectively leveraging the AI's ability to revise text.
- **Unbiased and Factual Reminder:** *"Ensure the answer remains unbiased, uses neutral language, and sticks to facts."* – Emphasizes impartiality and factuality. This is a gentle constraint that is especially important for persuasive writing or sensitive topics, to prevent the model from going on an opinionated or biased tangent. Including this instruction has been recommended to avoid stereotype or bias in outputs (26 principles for prompt engineering to increase LLM accuracy 57%). It makes the response more trustworthy and balanced – a critical aspect of persuasion is maintaining credibility.
- **Role Play for Style:** *"You are a storytelling coach. Turn the information into a compelling story to convince the reader."* – Combines persona with a goal. By role-playing (here as a storytelling coach), the model may infuse narrative elements or emotional appeal into the answer, which can be very persuasive to readers. This type of prompt can be adapted (e.g. "You are a lawyer defending this stance – argue convincingly…") to get a response that follows the techniques of that persona or profession. It's a way to **infuse creative or persuasive techniques** implicitly, by leveraging the model's knowledge of how different roles communicate.
- **Highlight Benefits / Implications:** *"Emphasize why this information is important and what the implications are for the reader."* – In persuasion or communication, it's effective to answer the "so what?" question. This prompt directs the model to not just provide information, but also to highlight significance and benefits, which is key to convincing someone. It ensures the answer doesn't read as a dry fact dump; instead, it connects facts to value or action, strengthening the persuasive impact.

Each of these prompts can be mixed and matched as needed. They are **generic and high-utility** – meaning they can apply to writing code, answering scientific questions,

giving medical advice (with domain knowledge), creating marketing content, and much more. By categorizing them, we can systematically craft prompts: for example, for a given task you might use one from the Critical Thinking set (*step-by-step reasoning*), one from Retrieval (*problem decomposition*), and one from Persuasion (*tone directive*) together. The ability to combine these prompt elements is a powerful way to adapt to virtually any context or domain.

Master Search Prompt Template

Below is a comprehensive **master prompt template** designed to guide an AI assistant through a thorough "search and answer" process. This template is structured to be adaptable to any research or information-gathering context. It incorporates the best practices (step-by-step reasoning, structured retrieval, clarity, etc.) into one long prompt. Following the long version, there is also a shorter version for quick use. **You can use these as a starting point and fill in or adjust details (in brackets)** to suit your specific query.

Long Version (Detailed Template)

[SYSTEM/ROLE:]

You are a highly intelligent research assistant with expertise in [domain/topic]. Your goal is to help answer questions thoroughly and accurately using a step-by-step approach.

[TASK:]

The user's query: "[Insert user's question or topic here]".

Your task is to:

1. **Analyze** the query to understand exactly what is being asked and identify the key components or sub-questions implicit in it.

2. **Plan** an approach to answer: break the problem down into logical parts or topics that need to be investigated or answered.

3. **Research** each part systematically. For each sub-question or topic:

- If you have relevant knowledge, recall the key facts, data, or concepts (cite any known sources or evidence if possible).

- If the answer might require external information, formulate what you *would* search for (e.g., keywords, database lookups) and infer likely findings from a trustworthy source. (Imagine what an expert or a textbook might say on the subject.)

- Critically evaluate the information: check for consistency, credibility, and relevance to the question.

4. **Synthesize** the information from all parts into a coherent, well-structured answer.

[GUIDELINES:]

- **Use a step-by-step reasoning process.** Before giving the final answer, show your reasoning steps for transparency. For example, you might number the steps or write a short "thinking" narrative that leads to the conclusion.

- **Ask clarifying questions** if the query is unclear or broad, *but only* if absolutely necessary. (Prefer to make reasonable assumptions and state them, rather than leaving the user waiting.)

- **Maintain accuracy:** Double-check important facts or calculations. If something is uncertain or controversial, mention that and consider multiple perspectives.

- **Stay organized and focused:** Tackle one aspect of the problem at a time, and keep the explanation of each part clear and concise.

- **Use a neutral, informative tone** (unless a different style is requested). The answer should be impartial and based on evidence. Avoid inserting personal opinions or biases.

- **Structure the final answer** in a logical way, with an introduction, body (which can be broken into sections for each subtopic or step), and a conclusion or summary. Use headings or bullet points if it improves readability for complex answers.

- **Support your answer with evidence.** Where applicable, cite sources, quote definitions, or include data. For instance, you might say "According to [Source], ..." or "A study in 2022 found that ...". If you don't have actual sources, you can refer to general knowledge (e.g., "Historically, ..." or "In biology, it is known that ...") to show the basis of your statements.

- **Be thorough but relevant:** Provide enough detail to fully address the question, including background information if needed to understand the answer. However, do not

digress into unrelated tangents. Every part of the answer should serve the user's query.

- **Conclude with a direct answer** or recommendation (if applicable). After the detailed reasoning, ensure you explicitly answer the question posed or fulfill the task given. The user should have a clear takeaway or final response.

[OUTPUT FORMAT:]

Respond with a clear, step-by-step explanation and the final answer. Where helpful, use bullet points or numbered lists for steps. If the answer involves multiple aspects, you can use sections with brief headers (e.g., "Background", "Analysis of X", "Evidence for Y", "Conclusion").

Begin your answer by acknowledging the question and outlining how you will address it, then proceed with the analysis and findings. Finally, provide the conclusive answer.

(You are now prepared to answer the user's query in a comprehensive way. Proceed with the step-by-step reasoning and final answer.)

How to use the long template: The sections in brackets ([SYSTEM/ROLE:], [TASK:], etc.) are organizational hints – you can omit those labels in practice, but ensure the content is included in your prompt. Fill in the [domain/topic] with the general area of the question (e.g. "medicine" or "history") if relevant, and replace "[Insert user's question or topic here]" with the actual question. This template first sets the role and objective, then enumerates the approach. The **Guidelines** segment encodes best practices (you might tweak or shorten these as needed). The idea is to provide the AI with a *recipe* for how to conduct the search and reasoning process. By explicitly writing out steps 1, 2, 3, 4, you're prompting the model to follow that process. The final output format reminder ensures the answer is well-structured for the user. This long prompt can span ~3 A4 pages if fully written out with the guidelines, and is meant to be comprehensive. It's especially useful when you want a very detailed and reliable answer, and you don't mind the AI taking a methodical, multi-step approach to get there.

Short Version (Quick Template)

For faster interactions or when you need a briefer prompt, here's a condensed template

that still captures the essence of the above:

You are a knowledgeable assistant expert in [domain]. Answer the following question with a step-by-step approach, researching as needed and providing a well-reasoned, evidence-backed response:

Question: [Insert question here]

Instructions:

- Break down the question into sub-parts and address each systematically.

- Reason through each step, and bring in relevant facts or data to support each part of the answer.

- If helpful, ask me for clarification or additional information before answering (otherwise state any assumptions you are making).

- Finally, provide a cohesive answer to the overall question, in a clear and structured format, citing sources or evidence where appropriate.

- Keep the tone informative and neutral, and ensure the answer is complete and accurate.

(Now begin your step-by-step reasoning to answer the question.)

In this short version, everything is compressed into a few concise instructions. You still specify the domain expertise, present the question, and list bullet-point instructions for the process (break down the problem, provide step-by-step reasoning with facts, clarify if needed, then give the final answer). It's about **3 short paragraphs** in length. This template is easier to quickly adjust on the fly – for example, you can add a specific output format ("give the answer in 3 bullet points") if desired, or drop a bullet if it's not needed. Despite being shorter, it retains the key elements: instructing the model to use a structured, evidence-supported approach and to be clear and accurate.

Using and Adapting the Templates: These templates are starting points – you should

feel free to modify the wording and emphasis based on your needs. For instance, if you know the user only wants a summary, you might add *"Keep the answer brief"* to the instructions. If the context is creative rather than factual, you might remove the emphasis on evidence and add something about creativity. The **long version** is meant to cover all bases (ideal for critical research queries where thoroughness matters), while the **short version** is for everyday use where a lighter touch is sufficient.

By following the structure in these templates, you leverage multiple prompt engineering best practices at once: you set a clear role and goal, you guide the model through a reasoning procedure, you remind it to check facts and remain clear, and you specify the desired output style. This greatly increases the chances of getting a high-quality answer, regardless of the domain of the question. The templates are designed to be **universal** – whether the query is about a scientific discovery, a historical analysis, a technical troubleshooting, or a business strategy, the same structured approach can be applied. Just insert the right details for your context, and the prompt will direct the AI to deliver an organized, insightful, and well-supported response.

Finally, remember that prompt engineering is an iterative **experiment**. These best practices and prompts are drawn from recent research and community experience (26 principles for prompt engineering to increase LLM accuracy 57%) (Prompt Engineering a Prompt Engineer - Microsoft Research), but models can behave unpredictably at times. Use the provided examples as building blocks. Over time, by observing how the AI responds, you'll develop an intuition for which prompt tweaks yield the best results for your particular use case. Happy prompting!

Sources:

- Press et al. (2023). *"Measuring and Narrowing the Compositionality Gap in Language Models."* (Introduces **Self-Ask** prompting: model asks and answers sub-questions, with optional search, boosting multi-hop QA accuracy) ([2210.03350] Measuring and Narrowing the Compositionality Gap in Language Models).
- Kojima et al. (2022). *"Large Language Models are Zero-Shot Reasoners."* (Shows that a simple prompt *"Let's think step by step"* enables chain-of-thought reasoning, greatly improving performance on reasoning benchmarks) ([2205.11916] Large Language Models are Zero-Shot Reasoners).
- Bsharat et al. (2023). *"Principled Instructions Are All You Need for Questioning LLaMA and GPT-4."* (Presents 26 prompt engineering principles with ~57%

accuracy gains; e.g. integrate audience, use direct instructions like "Your task is…", *avoid* overly polite language, encourage step-by-step thinking, etc.) (26 principles for prompt engineering to increase LLM accuracy 57%) (26 principles for prompt engineering to increase LLM accuracy 57%).

- White et al. (2023). *"A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT."* (Defines reusable **prompt patterns** – e.g. Persona, Template, Reflection, Question Refinement, Cognitive Verifier – and frameworks for combining them to solve different problems) ([2302.11382] A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT) ([2302.11382] A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT).
- Ye et al. (2023). *"Prompt Engineering a Prompt Engineer (PE2)."* (Proposes a meta-prompt that guides the model to produce better prompts, highlighting the value of detailed instructions, context specification, and stepwise reasoning in prompt design) (Prompt Engineering a Prompt Engineer - Microsoft Research).
- Bhatt (2024). *LearnPrompting - "Self-Refine Prompting."* (Describes the **Self-Refine** technique where the model iteratively critiques and improves its answer, leading to higher quality outputs especially on complex tasks) (Self-Refine: Iterative Refinement with Self-Feedback for LLMs).
- Yu et al. (2023). *"Exploring the Effectiveness of Prompt Engineering for Legal Reasoning Tasks."* (Finds that domain-specific structured prompts like IRAC outperform generic prompts in legal QA, underlining the importance of aligning prompts with context and task structure) (Exploring the Effectiveness of Prompt Engineering for Legal Reasoning Tasks - ACL Anthology).