KROK AI

AI for beginners

A practical guide to working with neural networks

16 lessons · 128 questions · a practical guide

Digital edition

Krok AI

KROK AI

AI for beginners

A practical guide to working with neural networks

16 lessons · 128 questions · a practical guide

Table of contents

1 How a neural network thinks: tokens, prediction, context window

2 Your first working prompt: task + format + constraints

3 Iterations instead of perfectionism

4 Roles, audience and tone: shifting perspective

5 A role ≠ magic. The template-instruction: a prompt with blanks

6 Few-shot prompting: teaching the model with examples

7 Structured output: lists, tables, JSON, schemas

8 Chains of steps: decomposition and self-checking

9 Texts: emails, posts, articles — from outline to proofreading

10 Long documents: summaries, fact extraction, reworking

11 Images and presentations: the prompt formula and slide structure

12 AI as a teacher: explanations, a quiz trainer, a personal plan

13 Spreadsheets and data without programming

14 Automating routine: from finding tasks to no-code connections

15 Hallucinations and fact-checking

16 Safety and privacy: a checklist before sending

LESSON 1

How a neural network thinks: tokens, prediction, context window

Let's start with the main thing: a neural network doesn't "know" anything and doesn't "think" like a human. It is a very powerful next-word prediction machine. When you type "The sky over the city was...", the model runs through millions of options and picks the most likely continuation: "gray", "clear", "hazy". All the text it generates is assembled one such prediction at a time.

The model doesn't work with letters but with tokens — chunks of text. One word usually consists of 1–3 tokens: "hi" may be a single token, while "prediction" splits into two or three. That's why the model sometimes fails oddly at counting letters in a word: it doesn't see letters, it sees whole chunks. It's not stupidity — it's just how it's built.

The model has a context window — working memory that holds both your messages and its answers. In modern chats that's tens and hundreds of pages of text. Within one chat, the model "remembers" everything you've discussed. But open a new chat — and it forgets everything: your name, the task, your agreements. So keep important details either in one conversation, or paste them into the prompt again.

The model can't say "I don't know" as honestly as we'd like. Its job is to predict a plausible continuation, not to verify it against reality. If you ask about a nonexistent book, the model will very likely invent a plausible title, author and year. And its tone will be just as confident as when answering a simple question. This is called a hallucination, and we'll look at it closely in lesson 15.

Practical takeaway: treat the neural network like a very well-read but slightly overconfident assistant. It writes, explains, rephrases and structures things excellently. But numbers, dates, quotes, links and titles are worth double-checking — especially if a decision depends on them.

EXAMPLE PROMPT

```
"I have a question about a historical event. If you're not sure about exact dates or details — say so directly, don't make things up. Question: when did the first supermarket open in the US?"
```

EXAMPLE PROMPT

```
"Explain how a neural network's context window works as if I know nothing about technology. Maximum 5 sentences, without jargon like 'embedding' or 'inference'."
```

> 💡 If an answer sounds confident but matters to you, treat it as a draft until you've verified it. The model's confident tone is a style of generation, not a sign of truth.

QUIZ

What is a token to a neural network?

- ○ A whole sentence
- ○ A single letter of text
- ● **A chunk of text that words are made of ✓**

Why: A token is a fragment of a word: a word usually splits into 1–3 tokens, which is why the model can make mistakes, for example, when counting letters.

What happens to the context if you open a new chat?

- ○ The model will remember the conversation and show it later
- ● **The model starts from a blank slate and forgets everything ✓**
- ○ The model carries memory over from the old chat

Why: The model's memory is limited to the context window of one conversation; a new chat is new, empty memory.

Why can a neural network confidently name a nonexistent book?

- ● **Its job is to predict plausible text, not to verify facts ✓**
- ○ It doesn't know a single book at all
- ○ It wants to deceive you on purpose

Why: The model generates a probabilistic continuation. A confident tone is a style of generation, not a sign of accuracy.

What fits into the context window?

- ○ All the user's chats over all time
- ○ Only the user's last message
- ● **Your messages and the model's answers in the current conversation ✓**

Why: The context window is the working memory of one conversation: both your messages and the model's answers.

The model is bad at counting the letter "r" in the word "extraordinarily". Why?

- ○ It always makes mistakes in long words
- ● **It sees token chunks, not individual letters ✓**
- ○ It doesn't know English

Why: The model works with tokens, not letters, so letter-by-letter tasks are hard for it.

The chat has gotten cluttered with old discussions and the answers got worse. What's the sensible move?

- [x] Start a new chat and briefly restate the task ✓
- [ ] Type in caps that the old stuff doesn't matter
- [ ] Ask the model to "forget" the previous messages

Why: A new chat with a brief recap clears the context of junk. "Forget it" doesn't delete messages from the window.

Which conclusion about the model's confident tone is correct?

- [ ] The model always writes unconfidently when it doubts
- [ ] A confident tone means the model verified the fact
- [x] A confident tone is just a result of generation, not a guarantee of truth ✓

Why: The tone comes from training on confident texts; it says nothing about whether the answer is correct.

Which of the following does a neural network usually do well?

- [ ] Accurately name today's date from memory
- [x] Rephrase, explain and structure text ✓
- [ ] Guarantee that links are up to date

Why: Working with text is the model's strong side. Dates, links and current facts are typical failure points.

HOMEWORK

Run two mini-experiments to get a feel for how the model works. Experiment 1: in a new chat, ask "How many letter 'o's are in the word 'onomatopoeia'?" and check the answer manually. Experiment 2: in one chat, set a topic (for example, "I'm planning a trip to Chicago"), write 3 messages, then open a new chat and ask what you were talking about.

CHECKLIST

- ☐ The letter-counting experiment was done, the answer checked manually and written down
- ☐ In the first chat, the model actually used details mentioned earlier
- ☐ In the new chat, the model confirmed it knows nothing about the previous conversation
- ☐ 1–2 of your own observations written down about how the model's "memory" will change your working habits

LESSON 2

Your first working prompt: task + format + constraints

A prompt is simply the assignment you give the model. The difference between "write a text about coffee" and a good prompt is like the difference between "do something" and a proper brief. A working prompt is built from three blocks: task, format, constraints.

Task — what exactly needs to be done and why. Format — what the result should look like: length, structure, type (list, table, email). Constraints — the boundaries: who the text is for, what to avoid, what style. It's also useful to add context — a little information about your situation so the model doesn't guess it on its own.

Build a prompt like this: (1) what to do, (2) in what form, (3) with what boundaries. For example: "Make a list of 7 birthday gift ideas for a coworker. For each idea — a title and one line of explanation. Budget up to $50, the person doesn't like coffee or sports gear". One prompt, three blocks — and the result is immediately several times more accurate.

EXAMPLE PROMPT

```
"Write an email to a client apologizing for a 2-day delivery delay. Length 3–4 sentences, polite and businesslike tone, no excessive excuses. At the end, offer a 5% discount on the next order. The client's name is Olivia".
```

EXAMPLE PROMPT

```
"Explain the difference between a mortgage and renting to someone who has never compared them. Format: a table with 5 rows — criterion, mortgage, renting. Criteria: down payment, monthly payments, flexibility, building equity, risks. No financial advice, just a comparison".
```

First — a vague task: "write something about health". Second — no format: the model chooses on its own, and almost never the way you need. Third — contradictory requirements: "in detail, but in two sentences". Fourth — expecting the model to guess the context: it doesn't say who the text is for or where it will go. Fifth — one attempt instead of iterating: getting the wrong thing, a person gives up, when refining would have been enough.

A good habit: before sending, reread the prompt through the eyes of a performer seeing the text for the first time. Is everything clear? What would you ask in the model's place? Everything you'd want to clarify in the model's place — add to the prompt right away.

> 💡 Keep a personal template at hand: "Task: ... Format: ... Constraints: ... Context: ...". Fill in the fields — the prompt is ready.

QUIZ

What three blocks make up a basic working prompt?

- ○ Greeting, question, farewell
- ● **Task, format, constraints ✓**
- ○ Topic, length, deadline

Why: A working prompt answers three questions: what to do, in what form, within what boundaries. Context is added as needed.

"Write a text about the benefits of exercise" — what's the main problem with this prompt?

- ● **No format or boundaries: length, audience, style are not set ✓**
- ○ The model doesn't know the word "benefits"
- ○ The word "exercise" is too short

Why: Without format and constraints the model picks the length and style itself — and will almost certainly miss.

What belongs to the "format" block?

- ○ "Text for the team's internal chat"
- ○ "Don't use corporate jargon"
- ● **"Make a table with 4 columns" ✓**

Why: Format is the shape of the result: table, list, length. Style bans are constraints; purpose is context.

Which mistake is a contradictory requirement?

- ○ "Use a list"
- ● **"Cover the topic in detail, but fit it into 2 sentences" ✓**
- ○ "Write an email for a client"

Why: Detail and a hard 2-sentence limit conflict; the model is forced to choose, and you won't get a predictable result.

Why include context about the situation in a prompt?

- ● **So the model doesn't guess your situation on its own ✓**
- ○ Context is only for programmers
- ○ So the model answers longer

Why: Without context the model makes assumptions; a couple of lines about the situation sharply improves accuracy.

You got "not quite right" but the direction is correct. What's smartest?

- ○ Accept it and edit everything by hand
- ○ Start a new chat from scratch
- ● **Refine the prompt: what exactly to change in the answer ✓**

Why: If the direction is right, this is a job for iteration: ask it to adjust the specific spots.

Which prompt follows the formula "do → in the form of → taking into account"?

- ○ "Tell me about investments"
- ● **"Explain in simple words what a savings account is, 3 sentences, for a teenager, no bank ads" ✓**
- ○ "Write well and nicely about money"

Why: There's a task (explain a savings account), a format (3 sentences) and constraints (for a teenager, no ads).

What should you do before sending a prompt?

- ○ Scan the prompt with antivirus
- ○ Add as many words as possible so the model definitely understands
- ● **Reread it through the eyes of a performer seeing the task for the first time ✓**

Why: Reading "through the model's eyes" reveals missing details — better to add them right away than later through iterations.

HOMEWORK

Take a real task from your life or work (an email, a shopping list, explaining a topic to a child, a social media post). Write a prompt using the formula "task + format + constraints + context", and get an answer. Then find at least 2 flaws in the result and refine the prompt once.

CHECKLIST

- ☐ The task is taken from your real life, not made up
- ☐ The prompt clearly contains the "task", "format" and "constraints" blocks
- ☐ At least 2 flaws of the first answer were found and written down
- ☐ The second, refined prompt noticeably improved the result — save both versions for comparison

LESSON 3

Iterations instead of perfectionism

A beginner tries to write the perfect prompt on the first try and gets upset when it doesn't work. An experienced user works differently: the first answer is a draft, and the prompt is a dialogue. A typical successful session with the model is 2–4 refinements — that's the norm, not a failure.

Key idea: don't rewrite the prompt from scratch — edit the result. The model remembers the whole dialogue, so short commands are enough: "make it twice as short", "add a concrete example to point two", "rewrite with a warmer tone, no corporate jargon". That way each step brings you closer to the goal instead of starting over.

Useful universal commands: "make it simpler" (complicated words go away), "give me 3 versions" (you get a choice), "more specific, with examples" (fluff goes away), "keep the structure, change only point three" (a targeted edit), "what did you miss from my request?" (the model self-checks). Memorize three or four of them — and 80% of situations are covered.

EXAMPLE PROMPT

```
"Good, but make it shorter: keep only the main idea in each paragraph, maximum 3
sentences per paragraph. Don't change the style or structure".
```

EXAMPLE PROMPT

```
"Give 3 different versions of this email: the first — businesslike and reserved, the
second — warm and personal, the third — very short, 2 sentences. Number them".
```

Another technique — ask the model to be self-critical: "Evaluate your answer: what's weak in it and what should be improved?" Often the model finds the fluff, missed details and weak spots itself — and you decide what to fix.

Iterations are great when the direction is right. But sometimes the answer "drifts": you've been editing a text for 10 messages and it's gotten worse than before. Or the task has fundamentally changed. Or the model is "stuck" on a bad version, and every edit drags along old flaws. Those are signals to open a new chat: restate the task based on what you've learned — and spend 2 prompts instead of 10.

> 💡 If you only need a change in one place — say exactly that: "change only the headline, leave the rest as is". Otherwise the model may rewrite the whole text.

QUIZ

Which working style with the model is usually the most productive?

- [x] Treat the first answer as a draft and refine it ✓
- [ ] Accept any first answer as final
- [ ] Write the perfect prompt on the first try

Why: 2–4 iterations is the norm. The first answer sets the direction; refinements bring it to the goal.

The model remembers previous messages in the chat. What follows from this?

- [ ] The model won't accept a refinement after the first answer
- [ ] You have to rewrite the prompt from scratch every time
- [x] Short refinements like "make it shorter" are enough ✓

Why: As long as the dialogue is in context, short commands to edit the result work without restating the task.

What does the command "make it simpler" do?

- [ ] Deletes half the text at random
- [x] Removes complicated words and simplifies the writing ✓
- [ ] Changes the topic of the text

Why: It's a universal refinement about simplifying language; length and structure are set with separate commands.

You need a choice among several approaches. Which refinement fits?

- [x] "Give 3 versions: reserved, warm and very short" ✓
- [ ] "Write the same thing again"
- [ ] "Answer yes or no"

Why: Asking for numbered versions with different characteristics gives you material for choice and a hybrid.

Why ask the model "evaluate your answer, what's weak"?

- [ ] It never works
- [ ] So it deletes the answer
- [x] It often finds the fluff and weak spots itself, and you decide what to fix ✓

Why: Self-criticism is a quick way to surface an answer's flaws without an external editor.

Which signal says it's time to open a new chat?

- [ ] You didn't like the first answer
- [x] You've been editing the text for 10 messages and it's gotten worse ✓

- ○ You want to ask one more clarifying question

Why: When edits drag along old flaws or the task has changed, a new chat with a brief recap pays off more.

You only need to change the email's headline. How to phrase the edit?

- ○ "Start the dialogue over"
- ○ "Rewrite the email"
- ● **"Change only the headline to ..., leave the rest as is" ✓**

Why: An explicit "don't touch the rest" protects the finished parts of the text from unnecessary rewrites.

What does the command "before answering, ask me 3 clarifying questions" do?

- ○ The model refuses to answer
- ● **The model gathers missing details before doing the work, lowering the risk of a miss ✓**
- ○ The model asks questions after answering

Why: This hands the model the right to collect requirements — it saves several rounds of edits.

HOMEWORK

Take any task (ideally the same one from lesson 2). Do at least three iterations: first answer → a format refinement ("make it shorter/simpler") → a content refinement ("add an example/more specifics") → optionally, ask for three versions. Then compare the final version with the first and write down what changed.

CHECKLIST

- ☐ At least 3 iterations were done in one chat
- ☐ At least 2 different types of refinements were used (about format and about content)
- ☐ The final version is saved next to the first one — the improvement is visible
- ☐ A conclusion is written down: which refinements worked best for you specifically

LESSON 4

Roles, audience and tone: shifting perspective

One of the most powerful techniques is giving the model a role and an audience. A role is "who" it will be: "you are an experienced editor", "you are a polite support manager". An audience is "who" the result is for: a child, a director, a neighbor who knows nothing about the topic. These two lines change the vocabulary, depth and tone of the answer more than ten style refinements.

Why it works: in training texts, every type of writer and reader has their own characteristic language. Role and audience "switch on" the right layer. "You are a tax consultant, explain the deduction to the director of a small firm" and "...explain it to a college freshman" will produce two very different texts.

The same question asked for different audiences is a great understanding tool. First ask "explain it like I'm 10" — you'll get the essence without jargon. Then "explain it like an expert to an expert" — you'll see the real terminology and subtleties. The difference between the two answers by itself shows where the hard parts of the topic are.

EXAMPLE PROMPT

```
"You are the editor of a popular online magazine. Check my text: fix errors, remove corporate jargon, keep my style and meaning. Don't rewrite from scratch; show the edits as a list: before → after".
```

Tone is set with sample phrases: "friendly, like texting a good friend", "official, like a bank's letter", "lively, like a good lecturer". The more concrete the image, the more precise the tone. You can also give a negative constraint: "no exclamation marks", "no words like 'unique' or 'innovative'".

And an important technique from lesson 3 that fits especially well here: "before answering, ask clarifying questions". Once you've given a role ("you are my nutritionist"), the model is entitled to find out the context — goals, constraints, habits. A good "performer" asks first instead of rushing to write.

EXAMPLE PROMPT

```
"You are a nutritionist who works with busy people. Before giving me a week of meal recommendations, ask me 3 clarifying questions about my goals, constraints and habits. Wait for my answers".
```

> 💡 A role is not magic and not a universal amplifier. It tunes style and level of detail, but doesn't give the model new knowledge or access to your information. More on this in lesson 5.

> 💡 Combine: role + audience + tone + format in one prompt. For example: "You are an editor... For beginners... Friendly... A list of 5 points...".

QUIZ

What influences the vocabulary and tone of the answer most?

- ○ Time of day
- ○ Prompt length
- ● **The set role and audience ✓**

Why: The role sets the writer's "voice", the audience sets the reader's language level. Together they rebuild the whole text.

"Explain it like I'm 10" — what's this technique for?

- ○ It shortens the text to save money
- ● **It removes jargon and shows the essence of the topic in simple words ✓**
- ○ It's only needed for children

Why: The child level cuts out terminology and exposes the essence — useful both for understanding and as a base for explaining to others.

What does comparing "for a beginner" and "for an expert" explanations give you?

- ● **You see where the hard parts and terminology of the topic are ✓**
- ○ The second explanation is always more correct
- ○ Nothing, they're the same thing

Why: The difference between the two levels shows which details require preparation and what can be said simply.

Which line sets tone rather than role?

- ○ "Audience: students"
- ○ "You are an experienced lawyer"
- ● **"Friendly, like texting a good friend" ✓**

Why: Tone is the coloring of the text (samples and bans), role is who the model acts as, audience is who it's for.

Is it true that the role "security expert" makes the model better at data protection?

- ○ Yes, a role adds new knowledge
- ● **No: a role tunes style and depth but doesn't add new knowledge ✓**
- ○ Yes, but only on paid plans

Why: A role activates the right layer of texts but doesn't add capabilities or information to the model.

Why ask the model to ask clarifying questions after assigning a role?

- ○ To check your literacy
- ○ To buy time
- ● So the performer gathers context before starting instead of guessing ✓

Why: A good performer clarifies requirements before starting: it replaces several rounds of rework.

Which wording is a negative tone constraint?

- ○ "Write warmly and lively"
- ● "No exclamation marks, no words like 'unique' or 'innovative'" ✓
- ○ "You are the magazine's editor"

Why: Negative constraints directly ban specific elements — they're easy to check by eye after the answer.

The model answers too dry and official. Which refinement is most precise?

- ● "Rewrite it in a friendlier way, like texting a good friend, don't change the meaning" ✓
- ○ "Answer shorter"
- ○ "Write better"

Why: A sample phrase ("like texting a friend") gives the model a concrete style reference.

HOMEWORK

Pick a topic you actually need to explain to someone (a relative, a colleague, a child, a client). Get three versions of the explanation from the model: for a 10-year-old, for an average unprepared adult, and for an expert. From the three versions, assemble one final one (a hybrid is fine) for your specific listener.

CHECKLIST

- ☐ The topic was chosen for a real person in your life
- ☐ All 3 versions with different audience levels were received
- ☐ It's written down how the versions differ (this is a map of the topic's hard spots)
- ☐ The final version was assembled for your listener and tested: is it clear to them

LESSON 5

A role ≠ magic. The template-instruction: a prompt with blanks

After lesson 4 it's easy to conclude: "appoint the model a chef — get dinner". No. A role is a lens, not a magic wand. A model with the "lawyer" role answers in a lawyer's language, but doesn't check current laws and bears no responsibility. A role doesn't give access to your documents, doesn't remember a previous chat, and doesn't replace a specialist in serious matters — money, health, law.

But roles are excellent at a different job: they save you explaining. If you write posts in one style every day, instead of describing the style every time you can fix it in the role: "You write short posts for the Telegram channel of a small bakery. Warm tone, short sentences, always one concrete detail from bakery life".

The "template" pattern is the most practical tool for repetitive tasks. You write an instruction with blanks in curly braces once, then only change the values. The model plugs in the values and executes the instruction. The curly braces matter: they clearly show where a variable is and where regular text is.

EXAMPLE PROMPT

```
"You write support emails for an online store. Template: "Hello, {name}! Your order # {number} {status}. {Customer action}. If you have questions — just reply to this email." Fill in: name = Anna, number = 45128, status = handed to the delivery service today, action = it will arrive on November 12, you can track it via the link in your account".
```

EXAMPLE PROMPT

```
"You prepare posts for a coffee shop's Telegram channel using a template: "Topic: {topic}. Hook: {question or fact}. Benefit: {what the reader learns}. Call to action: {what to do}". Fill in the template for the topic "a new seasonal drink — sea buckthorn riff".
```

A template has three big advantages. First: consistency — the result is built the same way every time. Second: speed — you fill in three fields instead of retelling the task. Third: delegation — you can hand the template to a colleague and they'll get the same result with no training.

Collect templates in a separate file: an email-to-client template, a post template, a task-breakdown template, a review-reply template. This is your personal "first-aid kit" of typical tasks. Over time you'll start seeing the template structure in any repetitive work — that's the most noticeable jump in effectiveness.

> 💡 Don't complicate a template right away: start with 2–3 variables. With more than five, filling it in becomes slow in itself.

💡 Store templates with real values separately from templates with blanks — and never paste clients' personal data into a public bot.

QUIZ

What does an assigned role actually give the model?

- ○ New knowledge and internet access
- ● A switch of style, vocabulary and level of detail ✓
- ○ A guarantee of professional-quality answers

Why: A role is a style lens. It doesn't add knowledge and doesn't replace a real specialist.

In which matters does the role "your lawyer" not replace a real lawyer?

- ● Serious decisions: law, money, health ✓
- ○ Nowhere, the role fully replaces a specialist
- ○ Writing a greeting

Why: A model without fact-checking and accountability shouldn't be the final authority on critical matters.

What does {variable} mean in a template prompt?

- ○ A special command for the internet
- ○ A system error of the model
- ● A place where you plug in a specific value ✓

Why: Curly braces mark a blank: each time you only change the values, the instruction stays the same.

The main advantage of an instruction template?

- ○ The model technically answers faster
- ● A consistent, repeatable result with minimal effort ✓
- ○ Templates only work for programmers

Why: A template gives repeatability and time savings, and is easy to hand to colleagues.

How many variables is it reasonable to have in your first template?

- ○ At least 10 for accuracy
- ○ The number of variables doesn't matter

- [x] 1–3, so filling it in is quick ✓

Why: Too many fields slows you down; start small and add as needed.

Which example is a template prompt?

- [ ] "Write an email"
- [x] "Template: "Hello, {name}! Order #{number} {status}". Fill in: Anna, 45128, being delivered" ✓
- [ ] "You are a lawyer"

Why: There's an instruction with blanks and specific values to plug in.

Why is it convenient to store templates in a separate file?

- [x] It's a personal library: repetitive tasks get done in seconds ✓
- [ ] Without a file templates don't work
- [ ] Files speed up the neural network

Why: Accumulated templates are the main personal asset: they save time and standardize your work.

You write a weekly summary for your manager in the same format. What do you do once to save time?

- [ ] Ask the model to remember the format forever
- [ ] Ask the model "write a summary" every time
- [x] Fix a template with the structure and variables, and fill it in weekly ✓

Why: "Remember forever" doesn't work between chats; a template solves the task reliably.

HOMEWORK

Find a task in your work or life that you do at least once a week in a similar form (an email, a report, a post, an answer to a typical question). Create an instruction template for it with 2–4 variables in curly braces and test it on three different sets of values.

CHECKLIST

- ☐ The task repeats at least once a week
- ☐ The template contains an instruction and 2–4 variables in curly braces
- ☐ The template was tested on 3 different sets of values — the results are structured the same way
- ☐ The template is saved in a separate file with a clear name

LESSON 6

Few-shot prompting: teaching the model with examples

"Few-shot" is when you show the model 2–3 examples of the desired result right in the prompt, and it continues the pattern for a new case. It's the most natural way to show the model "I need something like this": it's hard to describe in words, easy to show. One example — "one-shot", several — "few-shot", zero examples — "zero-shot".

An example is the best way to set a format. Try describing in words what a reply to a customer review should look like: you'll get tired writing rules. But three pairs of "review → reply" the model reads instantly and produces a fourth reply in the same spirit.

EXAMPLE PROMPT

```
"Turn customer reviews into short support replies. Tone is warm, no excuses, always offer
a solution.
Example 1. Review: "The courier was an hour late". Reply: "Sorry for the wait — a delay
is unacceptable. We'll send a promo code for your next order to your account."
Example 2. Review: "The item arrived dented". Reply: "We're sorry — we'll replace it
without hassle. The courier will bring a new one tomorrow and take the old one."
Now the review: "I couldn't get through on the phone for a long time"
```

When examples help the most: a nonstandard or brand-specific format, a specific tone and style, sorting by a scheme ("question → category → cause → action"), tasks with rare terminology.

Bad examples work worse than no examples: the model will copy the mistakes too. If the examples share one structure, the model will blindly repeat it even where it's inappropriate. If all the examples are too similar, the model may narrow the task. And be careful with sensitive data: don't show real customer correspondence in examples.

The format of examples matters more than their number. Two tidy, uniform examples work better than five sloppy ones. In the examples, keep the same look you want in the output: if you want an answer without quotation marks, don't put the example answers in quotation marks. And separate the examples explicitly: "Example 1… Example 2… Now: …".

EXAMPLE PROMPT

```
"Classify support tickets by the scheme: "Ticket → category → urgency". Categories:
delivery, payment, refund, other. Urgency: high/medium/low.
Example 1: "The package didn't arrive, I've waited 5 days" → delivery → high.
Example 2: "A question about future bonuses" → other → low.
Now: "I paid twice, refund my money"".
```

💡 Start with 2 examples. If the model gets confused, add a third or "anchor" a rule in words after the examples: "Always keep the answer length like in the examples".

💡 Examples can be combined with a template: instruction + a template with braces + 2 filled examples — the most reliable construction for regular tasks.

QUIZ

What is a few-shot prompt?

- [x] A prompt with 2–3 examples of the desired result ✓
- [ ] A prompt sent several times
- [ ] A prompt with a limit on answer length

Why: Few-shot is teaching inside the prompt: we show examples, the model continues the pattern.

Why show examples if you can describe the requirements in words?

- [ ] Nothing can be described in words
- [ ] Examples are never needed
- [x] An example conveys format and tone more precisely and faster, especially complex ones ✓

Why: Describing format and tone in words is bulky; the model picks up "input → output" pairs immediately.

What is more dangerous for the result?

- [ ] Two tidy, uniform examples
- [x] Two examples with errors and sloppiness ✓
- [ ] One good example

Why: The model copies patterns from examples, including mistakes; bad examples are worse than no examples.

All your examples are too similar to each other. What can happen?

- [ ] The model will automatically make the examples more diverse
- [ ] The model won't notice anything
- [x] The model may narrow the task and repeat the structure even where it's not needed ✓

Why: Uniform examples set too narrow a pattern of behavior.

You want an answer without quotation marks. How should the examples be formatted?

- ○ Put the answers in quotation marks for looks
- ● Give the answers in the examples without quotation marks too ✓
- ○ It doesn't matter, the model ignores quotation marks

Why: The examples' format is the strongest signal: the model will reproduce the look of the examples, down to small details.

Which element makes examples convenient for the model?

- ● Explicit numbering and separation: "Example 1… Example 2… Now: …" ✓
- ○ The order of letters inside words
- ○ Including customers' personal data

Why: Explicit separation helps the model tell the examples apart from the working case and the instruction.

The model gets confused even with two examples. What's the first step?

- ○ Delete all the examples
- ○ Abandon few-shot
- ● Add a third example or anchor a rule in words after the examples ✓

Why: Strengthening the signal — another example or a verbal rule — usually straightens out the behavior.

Which construction is the most reliable for regular tasks?

- ○ Only a long instruction in words
- ● Instruction + a template with braces + a couple of filled examples ✓
- ○ Only examples without an instruction

Why: The combination sets rules, structure and a sample — minimal freedom for random interpretations.

HOMEWORK

Pick a typical "input" from your practice: reviews, emails, comments, requests. Build a few-shot prompt with 2–3 pairs of "input → desired output" and process 5 real inputs. Then swap one example for a bad one (with sloppiness) and compare — feel the power of example quality.

CHECKLIST

- ☐ The inputs are taken from your real practice (anonymized)

- ☐ The prompt has 2–3 uniform "input → output" pairs
- ☐ At least 5 new inputs were processed, and the outputs match the style of the examples
- ☐ An experiment with a "bad" example was done and a conclusion about the difference written down

LESSON 7

Structured output: lists, tables, JSON, schemas

When the result needs to go somewhere — into a spreadsheet, a form, a program — free-form text won't do. You need structured output: a list, a table, or even JSON — a simple machine-readable "key: value" format. Good news: the model is great at returning structure if you ask for it directly.

Start with the most common kinds. List: "return a numbered list, one item per line". Table: "return a table with columns: name, price, deadline". For tables it's convenient to ask for Markdown — the chat will display it nicely, and you can copy it into Excel without losses.

EXAMPLE PROMPT

```
"Read the shopping list below and return a Markdown table with columns: Item | Price |
Category. No explanations before or after the table.
Items: milk $1.10 (dairy); bread $0.55 (bakery); laundry detergent $4.00 (household);
apples $1.80/lb (produce)".
```

For a strict format, use the technique "return strictly in the format of...". You describe a schema — a set of fields and their type — and ask for only that back. A simple schema example: "Return strictly JSON: {"item": string, "price_usd": number, "in_stock": true|false}. No text before or after". JSON is needed when the result will go further into automation (lesson 14) or into a spreadsheet with formulas.

EXAMPLE PROMPT

```
"From the listing text below, extract the data and return strictly JSON by the schema:
{"type": "apartment|house", "district": string, "area_m2": number, "price_mln": number,
"furniture": true|false}. JSON only, no comments.
Listing: "For sale: 2-bedroom apartment, 54 m2 in the Greenwood district, $820K, some
furniture stays"".
```

What to do if the structure breaks: extra text around it, "floating" commas in JSON, "about 8 million" instead of a number? First, repeat more firmly: "return only JSON, no introductory phrases and no text after". Second, check the schema: maybe you didn't specify the value's type. Third, give the model the broken fragment and ask it to fix it: "here's JSON with an error — fix it so it's valid".

A useful habit — ask for self-checking: "after generating, check that the output strictly matches the schema, and if not — fix it". And if the data is important, spot-check it manually: the structure can be perfect while the value inside is made up.

> 💡 For tables ask for Markdown, for automation — JSON, for human reading — a list with subheadings. Choosing the right form is half the success.

💡 If the model inserts "approximately", add a rule: "If a value is missing from the text — put null, don't invent it". This sharply reduces the amount of fabricated data.

QUIZ

Why is structured output needed?

- ○ Structured output is always more factually accurate
- ○ To make the answer look longer
- ● **So the result can be pasted into a table, a form, or processed programmatically ✓**

Why: Structure is about convenience of further work, not about the reliability of the content.

What is JSON in simple words?

- ○ A programming language
- ● **A machine-readable "field: value" format in curly braces ✓**
- ○ A kind of font

Why: JSON is a text format with strict rules; spreadsheets, automations and programs read it.

Which technique makes the output as strict as possible?

- ○ Asking to answer in caps
- ○ Asking "try hard"
- ● **The wording "return strictly in the format of..." with an explicit field schema ✓**

Why: The word "strictly" plus a schema is an explicit contract: a set of fields, types, a ban on extra text.

In JSON the model wrote "about 8 million" instead of a number. What to do?

- ○ Accept it
- ● **Clarify the field's type in the schema and ask to fix the broken fragment into a valid one ✓**
- ○ Delete the whole field

Why: The "number" type in the schema + a request to fix the fragment is the standard structure repair.

What does the rule "if a value is missing — put null, don't invent it" mean?

- ● **Values missing from the source are marked with an empty value instead of being fabricated ✓**
- ○ The model will stop answering

- ○ The model will delete text

Why: This is the main anti-hallucination technique in data extraction.

Why ask for a Markdown table rather than JSON?

- ○ Markdown is more reliable for programs
- ○ There's no difference at all
- ● **Markdown looks nicer in the chat and copies into Excel easily for a human ✓**

Why: Markdown is for humans and manual work, JSON is for machines and automation.

The structure is perfect but the price value looks suspicious. What does that mean?

- ○ Structure protects the values too
- ● **Structure and facts are different things: content must be checked separately from formatting ✓**
- ○ Values are always correct if the structure is valid

Why: The model can format a fabrication perfectly. Important values are verified against the source.

The output is wrapped in extra phrases like "Sure! Here's your JSON:". What to do?

- ● **Repeat more firmly: "only JSON, no text before or after" ✓**
- ○ Say goodbye to the model
- ○ Always ask for Markdown

Why: An explicit ban on text before and after is the most direct fix for wrapper junk.

HOMEWORK

Take 5–10 similar text snippets from your practice (listings, requests, emails, reviews). Build an extraction schema (4–6 fields) and get a Markdown table. Then repeat the same with strict JSON and the rule "no data → null". Compare both results.

CHECKLIST

- ☐ At least 5 real, similar snippets were used
- ☐ The schema contains 4–6 fields with types specified
- ☐ Both a Markdown table and strict JSON with the null rule were obtained
- ☐ A spot manual check of the values against the sources was done, errors recorded

LESSON 8

Chains of steps: decomposition and self-checking

A complex task "in one leap" fails even for humans. The secret of working with the model is the same as with contractors: cut the task into steps. Decomposition isn't about "getting smarter" — it's about control: at each step you see the result and can fix it before it drags the following steps along.

The "solve step by step" technique (sometimes written as "think step by step") is simple: ask the model to first write out a plan or reasoning, then the answer. When the model "voices" its steps, quality on tasks with logic and calculations noticeably improves. Even better — drive the steps yourself.

EXAMPLE PROMPT

```
"Don't answer right away. First list 5 steps needed to put together a home renovation budget. Wait for my 'next'. Then go through the steps and draft a budget with approximate price ranges".
```

The classic chain for texts and projects: (1) an outline of points → you edit the outline; (2) a draft from the outline → you adjust the approach; (3) the final version with the edits. At step 1 rework costs seconds; at step 3 — minutes. That's how you buy quality cheaply.

EXAMPLE PROMPT

```
"Task: prepare a comparison of three mobile plans for a family of 4. Step 1: make a list of comparison criteria (5–7 of them) and show me. Don't write the comparison itself until I approve the criteria".
```

The second half of the lesson — "check your answer". The model can verify itself: recount the numbers, compare the conclusion with the task, find contradictions. Useful wordings: "solve it, then verify the solution step by step", "reread your answer and find mistakes", "check the answer against every point of the task and list which conditions are covered".

A checklist instead of the vague request "do it well" works great: list the quality criteria for the model ("in the answer: the deadline is stated, there's an amount, there's a signature, no corporate jargon") and ask it to check the answer against each point. That turns fuzzy quality into something measurable.

EXAMPLE PROMPT

```
"First solve the task. Then separately list all the conditions of the task and note how your answer accounts for each one. If any condition isn't covered — fix the answer before sending".
```

> 💡 Split a complex task into several messages rather than one paragraph: "step 1 — only the

> outline". Fixing an outline is 5 times cheaper than fixing a finished document.

QUIZ

What's the main point of task decomposition?

- ○ The task becomes shorter in text
- ● **Control over the result at each step, cheaper edits ✓**
- ○ The model technically works faster

Why: A step-by-step approach lets you catch errors early — fixing an outline is far cheaper than a final document.

What does the "solve step by step" technique do?

- ○ Reduces the number of tokens
- ○ Forbids the model from answering
- ● **Asks to first write out reasoning or a plan, then the answer — logical quality improves ✓**

Why: "Voicing" the steps helps the model on tasks with logic and calculations.

The classic chain for texts is...

- ○ Three random questions in a row
- ● **Outline → edits → draft → edits → final ✓**
- ○ Final first, then the outline

Why: First the outline (cheap to edit), then a draft, then the final version with the edits.

Why ask the model to list the task's conditions and check them against the answer?

- ● **Missed conditions are found before you'd notice them ✓**
- ○ The model counts faster that way
- ○ It's decoration for the answer

Why: Self-checking against a list of conditions is a practical way to catch omissions.

Why is a checklist better than the request "do it well"?

- ○ It turns off hallucinations
- ○ It's shorter
- ● **It turns fuzzy quality into specific checkable points ✓**

Why: Listed criteria give the model (and you) a measurable standard to check against.

At which step are edits cheapest?

- ○ At the final version
- ● At the outline stage ✓
- ○ Edits cost the same

Why: Changing an outline point takes seconds; reworking finished text for a new outline takes minutes.

Which wording is a direct request for self-checking?

- ● "Solve it, then reread, find mistakes and fix them" ✓
- ○ "Think like an expert"
- ○ "Answer in detail"

Why: It's a two-phase prompt: the solution, then a separate checking phase with a fix.

When is it better to spread the steps across different messages?

- ○ Only in lessons
- ○ Never, everything should be written in one message
- ● In complex tasks — so you can insert your edits between the steps ✓

Why: Separate messages give you a control point after each step.

HOMEWORK

Take a task that's complex for you (comparing purchases, planning an event, structuring a report, breaking down a budget). Take it through the chain "outline → draft → final" in three separate messages. After the final, ask the model to check the result against your checklist of 4–5 criteria.

CHECKLIST

- ☐ The task is genuinely complex (it can't be solved head-on with one prompt)
- ☐ The outline was agreed before the draft, and outline edits were made
- ☐ The draft and the final are separate steps, with the differences visible
- ☐ The model checked the final against your checklist, and the found flaws were fixed

LESSON 9

Texts: emails, posts, articles — from outline to proofreading

Text is the most common task for beginners, and here the model shows its full strength. The main rule: every kind of text has its own goal. An email — so the recipient does something or learns the main point in 10 seconds. A post — to hook and hold. An article — to fully answer the reader's question. Tell the model the goal, and it will pick the structure.

A business email: greeting → the point in the first paragraph → details → a clear call to action → signature. A refusal: thanks → a short "no" without lying → optionally an alternative → a warm ending. A request for information: who you are → what exactly you need → why and by when → how their answer will make your life easier.

EXAMPLE PROMPT

```
"Write a polite refusal to a freelancer who proposed working together at a price above our budget. Structure: thanks → an honest "no" because of the budget → an offer to reach out again in six months → a warm tone. Maximum 5 sentences, no corporate jargon".
```

EXAMPLE PROMPT

```
"Write a request email to a supplier: we ask for the current price list and March delivery times. Context: we've worked with them for two years, volume of about 200 units a month. Businesslike, warm tone. At the end — a concrete question with a deadline: we need the answer by Friday".
```

For posts and articles, the chain from lesson 8 works: outline → parts → assembly. First ask for 3–5 headline options and a post outline, choose and adjust. Then generate one section at a time — that way each piece gets attention. The finale is assembly and a single style pass.

Two quality techniques. "3 versions and a hybrid": ask for three different versions of an intro or the whole post and glue your own from the best pieces — it's almost always better than any single version. And "proofreading": give the model your finished text with the editor role — it will find typos, corporate jargon, weak spots and show the edits as a list.

EXAMPLE PROMPT

```
"You are an editor. Here is my text (below). Check it for: correctness, corporate jargon, repetition, logic. Show the edits as a "before → after" list with a brief explanation of each. Don't rewrite the whole text. Text: "…"
```

> 💡 Tell the model where the text will go and who will see it: "a post for a Telegram channel for moms of preschoolers". That changes everything: length, slang, examples.

> 💡 Always ask for the numbers, names and promises to be listed separately after the text — that way you can check the riskiest parts in a minute.

QUIZ

What should you name first when working on a text?

- ○ The font color
- ○ The length in characters
- ● **The text's goal and its audience ✓**

Why: The goal determines the structure, the audience — vocabulary and examples. Length is a consequence.

How is a business email structured?

- ○ Details → the point → greeting
- ● **The point in the first paragraph → details → a clear call to action ✓**
- ○ Reasoning → conclusion → a hint at action

Why: A business email saves the recipient's time: the point right away, the action concrete.

How do you build a polite refusal correctly?

- ● **Thanks → an honest "no" → optionally an alternative ✓**
- ○ Silence is also an answer
- ○ A long apology and promises

Why: An honest "no" without lying and, where possible, an alternative preserve the relationship.

What does the "3 versions and a hybrid" technique give you?

- ○ A guarantee that all three versions are identical
- ○ The model works slower
- ● **The best result: you assemble your own from the strong pieces ✓**

Why: Versions show the forks; a hybrid collects the best — usually stronger than any single one.

Which order of work on an article does the lesson recommend?

- ○ The whole text at once
- ● **Outline → edits → parts one by one → assembly ✓**

- ○ Final first, then the outline

Why: The chain of steps (lesson 8) applied to text: editing the outline is cheap, and parts get attention separately.

Why ask the model to list the numbers and promises separately after the text?

- ● So you can check the riskiest spots of the text in a minute ✓
- ○ The model writes better that way
- ○ For looks

Why: Numbers and promises are the main places for errors and fabrications; a separate list makes checking easier.

The model sent finished text with made-up statistics "according to research". What to do?

- ○ Add more numbers
- ○ Publish it, it sounds solid
- ● Delete or replace it: demand either a source or a wording without made-up numbers ✓

Why: Unverified numbers are a classic hallucination; publishing without checking is dangerous for your reputation.

Which prompt asks for proofreading rather than a rewrite?

- ○ "Rewrite my text better"
- ● "Show the edits as a "before → after" list, don't rewrite the whole thing" ✓
- ○ "Do the same thing but nicer"

Why: The "before → after" format keeps your text and gives targeted, explainable edits.

HOMEWORK

Use the model to write real texts in three different genres: a business email (a request or a refusal), a social media post via the chain "outline → parts → assembly", and a proofread of your existing text in the editor role. For one of the genres, use "3 versions and a hybrid".

CHECKLIST

- ☐ All three genres of texts were created for real tasks
- ☐ The outline → parts → assembly chain was used for the post
- ☐ 3 versions were received for at least one text and a hybrid was assembled
- ☐ Proofreading was done: edits were shown in the "before → after" format

LESSON 10

Long documents: summaries, fact extraction, reworking

Long documents are where AI saves you hours. Three basic operations. Summarize: compress a document to the main points. Extract: pull specific facts into a table. Rewrite: rework it for a different audience or purpose. These are three different prompts to the same text — and three different results.

Summarize works well if you say what the summary is for: "to decide whether to attend the meeting", "for a report to the manager", "to decide whether to read it in full". The purpose determines what makes it into the selection. A useful form is "5 key points + 1 risk + 1 open question".

EXAMPLE PROMPT

```
"Here's the text of a service agreement (below). Make a summary for the owner of a small shop who's deciding whether to sign: 5 main points, separately — 3 items with obligations and penalties, separately — what happens on termination. No legal assessment, only facts from the text".
```

Extract is the most underrated operation. From a contract you can pull amounts and deadlines into a table, from a report — all the metrics by month, from 40 reviews — every mention of problems. The link with lesson 7 works here: set the table's columns explicitly. And the main rule — "only facts from the text": if something isn't there, write "not in the text" instead of guessing.

EXAMPLE PROMPT

```
"From the text below, extract all the parties' obligations into a table: Obligation | Who | Deadline | Consequences of breach. Only information from the text; if a field isn't specified — write "not in the text". Text: "…"
```

Rewrite — reworking for an audience. A technical report → a 5-point summary for a manager. An instruction → a quick-reference card for a newbie. An article → a script for a short video. The prompt sets: the source, who the reader is, the goal, the form of the result. This differs from summarize: here the form and the addressee change, not just the length.

On length and parts. If a document is over 50–100 pages, split it into chunks and process them one by one, then ask the model to combine the partial results into an overall summary. And keep in mind: the longer the document, the easier it is for the model to "lose" a detail from the middle — check important spots by searching the source.

EXAMPLE PROMPT

```
"Rewrite this technical report for a manager without a technical background: 5 points — what was done, what wasn't done, what's needed from them. No jargon, each line is one fact. Report: "…"
```

💡 Always specify: "facts only from the text; if the data isn't there — say so". It's the cheapest protection against fabrications in summaries.

QUIZ

How does extract differ from summarize?

- ○ Not at all
- ● **Extract pulls specific facts into a structure, summarize compresses into the main ideas ✓**
- ○ Summarize is more factually accurate

Why: Summaries are about the essence and meaning; extraction is about specific fields and data, usually in a table.

Why tell the model what you need the summary for?

- ● **The purpose determines which facts make it into the summary ✓**
- ○ It's a formality
- ○ So the model answers faster

Why: "For a decision" and "for a report" select different main points; the purpose drives the selection.

What does the rule "only facts from the text" mean?

- ○ The model must quote the whole text
- ○ The model must answer shorter
- ● **Data is taken only from the document; if absent — the note "not in the text" ✓**

Why: The rule blocks guessing — the main risk in data extraction.

A 120-page document. How to process it?

- ○ Send it whole, once
- ● **Split into parts, process sequentially, then combine into an overall summary ✓**
- ○ Take only the first page

Why: Large documents are split into chunks; partial results are later combined.

Rewrite is...

- ● **Reworking a text for a different audience, purpose and form ✓**
- ○ A spelling check

- ○ The same as summarize

Why: Rewrite changes the addressee and the form (report → summary, instruction → card), not just the length.

In the table of obligations from a contract, an item says "not in the text". This is...

- ○ A reason to redo everything
- ○ A model error
- ● Correct behavior under the missing-data rule ✓

Why: An explicit absence mark is better than a fabricated value — it's easy to spot and fill in by hand.

Why should important spots in long documents be checked by searching the source?

- ○ The model is obligated to lie
- ● The model can "lose" a detail from the middle of a long text ✓
- ○ It's faster to read that way

Why: With long documents, details from the middle more often slip out of the model's attention.

Which first step is useful before summarizing a large document?

- ○ Delete half the text
- ○ Ask for a full retelling right away
- ● Ask for a map: how much of the document is about the topic you need and in which sections ✓

Why: A document map orients you: which parts to process in detail, which to skip.

HOMEWORK

Take a real long document (a contract, a report, an instruction, a 10+ page article). Do all three operations: a summary for a specific decision, a fact-extraction table (5–6 columns), and a reworking of a fragment for a different audience. Verify at least 3 extracted facts against the source.

CHECKLIST

- ☐ The document taken is real and long
- ☐ The summary is framed for a specific decision or task
- ☐ The extraction table has 5–6 columns and the "not in the text" rule
- ☐ At least 3 facts were verified against the source, discrepancies noted

LESSON 11

Images and presentations: the prompt formula and slide structure

Image generation follows the same logic as text: the more specific the assignment, the better the result. The working prompt formula for images: object + style + composition + details. Object — what's shown. Style — how: photo, illustration, watercolor, flat vector, 3D render. Composition — the framing plan: close-up, top view, object centered. Details — light, color palette, mood.

EXAMPLE PROMPT

```
"A photo of a cozy coffee shop early in the morning: a barista behind a wooden counter pouring coffee into a ceramic cup. Style: natural photography, warm morning light from the window. Composition: medium shot, the object slightly right of center. Details: soft shadows, warm palette – beige and brown, an atmosphere of quiet".
```

The key principle of iterating on images: change one variable at a time. Didn't work — change only the style. Wrong mood — only the lighting details. If you change everything at once, you won't know what helped. Keep a mental list: object/style/composition/details — and turn one at a time.

Words in images are a weak spot of generators: letters can be distorted, especially non-Latin scripts. The rule: minimal text in the image, short words in large font, text in quotes in the prompt — "a sign saying "OPEN"". If the lettering is critical — generate a background without text and add the lettering in an editor. That's more reliable.

EXAMPLE PROMPT

```
"A flat vector illustration for a finance presentation cover: a rising chart, coins, a magnifying glass. Style: minimalism, 3 colors – navy blue, teal, white. Composition: elements in the right half, a clean field on the left for the headline. No text in the image".
```

For presentations, AI is first of all a structure builder. It's good at the frame: slide points, narrative logic, questions for the audience. But design and final files are still more reliably done in regular editors, taking the content and structure from the model.

EXAMPLE PROMPT

```
"Make an 8-slide presentation structure on the topic "How we cut delivery time by 20%". For each slide: a headline (up to 6 words), 3 bullet points, an idea for the visual. Audience – department heads, goal – show the results and win support".
```

💡 Save successful image prompts together with the result: the "prompt → image" pairing is your personal style library.

💡 For slides, ask for a "visual idea" for each point — later those ideas turn into image prompts. That way text and graphics are born in one package.

QUIZ

What four parts make up the image prompt formula?

- [x] Object, style, composition, details ✓
- [ ] Color, font, size, format
- [ ] Topic, headline, text, caption

Why: What's shown, how in style, how in framing, and what details — four levers of control.

Why should image iterations change one variable at a time?

- [ ] The model doesn't accept several changes
- [ ] It's technically faster
- [x] Otherwise it's unclear which change had the effect ✓

Why: One variable at a time is a controlled experiment: you see exactly what changed.

Why is it better to add text in an editor?

- [ ] An editor gives the image style
- [x] Generators often distort letters, especially non-Latin ones ✓
- [ ] It's cheaper

Why: The reliable path: generate a background without text, add the lettering in an editor.

How do you specify lettering in an image prompt?

- [x] Put the text in quotes: a sign saying "OPEN", in a short word and large ✓
- [ ] Write the lettering in caps without quotes
- [ ] Describe its meaning

Why: Quotes plus a short word and large size raise the chance of correct rendering.

What is AI most reliable at in presentations?

- ○ Presenting the slides to the audience
- ○ Drawing the final slide design
- ● **Building the frame: structure, key points, visual ideas ✓**

Why: The model is strong in content and structure; final design is more reliably done in an editor.

What should a prompt for a presentation structure contain?

- ○ Only the topic
- ● **The topic, the number of slides, the audience, the goal and the format of the points ✓**
- ○ The participants' names

Why: Audience and goal drive the selection of points; format sets the uniformity of slides.

The cover needs clean space for a headline on the left. How do you order that?

- ○ Nothing, the model will guess
- ○ "Put the headline on the left"
- ● **In the prompt: elements in the right half, a clean field on the left for the headline ✓**

Why: Composition is ordered in words about element placement and free zones.

The image came out close, but the mood is too cold. What to change?

- ○ The whole object
- ● **Only the details: the light and the color palette to warm ones ✓**
- ○ The style to 3D

Why: Mood is the "details" level; changing the object or style would destroy what already works.

HOMEWORK

Create a visual kit for a real task: one image using the "object + style + composition + details" formula (for social media, a cover or an ad) and a 6–8 slide presentation structure with key points and visual ideas. Do at least 3 iterations of the image, changing only one variable each time, and write down exactly what changed.

CHECKLIST

- ☐ The image prompt explicitly contains all 4 parts of the formula

- ☐ At least 3 iterations were done, changing only one variable at a time
- ☐ The presentation structure contains headlines, key points and visual ideas for each slide
- ☐ A conclusion is written down: which variable changes worked best

LESSON 12

AI as a teacher: explanations, a quiz trainer, a personal plan

An underrated way to use a neural network — a personal teacher available around the clock. Unlike an article or a video, the model adapts to you: asks again, simplifies, gives examples from your field. The basic technique is layered explanation: first "in simple words", then "a bit deeper", then "technically precise".

EXAMPLE PROMPT

```
"Explain what inflation is in three levels: (1) in simple words with a real-life example, (2) deeper — with mechanisms and connections, (3) technically precise with terminology. After each level, ask whether to move to the next".
```

The second technique — metaphors from your field. Tell the model your profession: "I'm an accountant. Explain how a VPN works using analogies from accounting and paperwork". A familiar subject area makes new things understandable far faster than abstract examples.

The strongest learning technique is flipping the roles. Instead of "tell me" — "quiz me". The model asks questions on the topic one by one, checks your answers, corrects mistakes and raises the difficulty. This is active recall — a proven way to learn, incomparably better than passive reading.

EXAMPLE PROMPT

```
"You are my teacher of the basics of email layout. Ask me one question at a time, from simple to complex. After each of my answers: right/wrong, a brief explanation, the next question. After 10 questions, give a grade and a list of topics where I'm weak".
```

EXAMPLE PROMPT

```
"Walk me through my mistake. I did it like this: (description). First ask me 3 questions to understand my train of thought, then point out exactly where the mistake is and give me a similar task to practice".
```

The third scenario — a personal learning plan. Describe your starting level, the goal, and how much time per week you can realistically spend. The model will build a week-by-week plan with topics and practice. Then the plan lives on: every week you come back, report what you managed, and adjust.

> 💡 After breaking down a topic, ask: "Give me 3 questions I should be able to answer if I understood the topic". Testing yourself is the best measure of understanding.

QUIZ

What's the main advantage of AI over an article or video when learning a topic?

- ○ The model always cites sources
- ○ The model answers longer
- ● **The model adapts: simplifies, asks again, changes examples for you ✓**

Why: Adaptability is the key: the explanation adjusts to your questions and level.

What is "layered explanation"?

- ○ Several different topics in a row
- ● **One question at different depths: simple → deeper → technical ✓**
- ○ Explanation via tables

Why: Layers let you stop at a comfortable depth and not drown in terminology right away.

Why ask for metaphors from your profession?

- ● **A familiar area makes new things understandable faster ✓**
- ○ The model loves accountants
- ○ The answer gets longer that way

Why: An analogy from your field maps the structure of new knowledge onto what you already know.

What does the "quiz trainer" technique do?

- ○ The model grades you on a scale
- ○ The model gives a lecture
- ● **The model asks you questions, checks the answers and raises the difficulty ✓**

Why: Active recall: questions to you + feedback is stronger than passive reading.

How do you flip the roles in a prompt?

- ○ "Tell me about the topic"
- ● **"Ask me one question at a time, check my answers" ✓**
- ○ "Make an outline of the topic"

Why: The model goes from narrator to questioner — you train instead of reading.

What needs to be specified for a personal learning plan?

- ○ Name and age

- ○ Only the goal
- ● **Starting level, goal and realistic weekly time ✓**

Why: Without level and realistic time, the plan will be either too hard or unfulfillable.

Which request is a good understanding check after breaking down a topic?

- ○ "Give me a summary"
- ● **"Give me 3 questions I should be able to answer if I understood the topic" ✓**
- ○ "Praise me"

Why: Trying to answer control questions immediately shows where understanding is patchy.

A mistake walkthrough via 3 questions about your train of thought is needed to...

- ● **Find exactly where the logic broke, not just show the correct answer ✓**
- ○ Make the dialogue harder
- ○ Buy time

Why: Diagnosing the train of thought is more precise than a ready solution: the cause gets fixed, not the symptom.

HOMEWORK

Pick a topic you've long wanted to understand (finance, a technology, rules, a language). Run a session: (1) a three-layer explanation, (2) a metaphor from your profession, (3) a 10-question quiz trainer with a grade at the end. Save the final grade and the list of weak spots.

CHECKLIST

- ☐ The topic is real and personally relevant to you
- ☐ All 3 layers of explanation and a metaphor from your field were received
- ☐ A quiz trainer of at least 10 questions with feedback was run
- ☐ The grade and 2–3 weak spots are written down — they'll be the topic of the next session

LESSON 13

Spreadsheets and data without programming

You don't need to know how to program to work with data. A neural network can write formulas for you: you describe in words what needs to be calculated, and it gives a ready formula for Excel or Google Sheets with an explanation. Your job is to describe the table's structure correctly and check the result.

EXAMPLE PROMPT

```
"I have a Google Sheets table. A — date, B — manager, C — sale amount, D — refund (yes/no). Write a formula that sums manager Smith's sales for March without refunds. Give the formula and explain part by part what it does".
```

Real spreadsheets are dirty: "Smith" and "smith " are one person, dates in three formats, amounts with spaces and letters. AI is great at helping bring everything to one form. The technique: paste a chunk of dirty data (anonymized!) and ask for cleaning rules — both in words and as a ready formula, if one exists.

EXAMPLE PROMPT

```
"Here's a fragment of a names column: "smith ", "Smith", "SMITH", "Petrov-Vodkin", "petrov-vodkin". Suggest a rule to bring them to a single form and write a Google Sheets formula that does it automatically. Explain which cases the formula might miss".
```

The second scenario — summaries and conclusions. Paste a table (or an anonymized part of it) and ask like an analyst: "Which three months were strongest by revenue and by how many percent from the average?", "Which product drives 80% of sales?", "What's odd in this data — outliers, gaps, duplicates?". The model will calculate and, most importantly, explain the calculation.

EXAMPLE PROMPT

```
"Here's a sales table by month: (paste the data). Calculate: (1) the average annual revenue, (2) the months above average with the deviation in percent, (3) the three biggest consecutive drops. Show the calculations step by step so I can verify. Separately point out where the data is insufficient for conclusions".
```

A set of typical analytical questions worth keeping in mind: "what's the biggest/smallest here and by how much", "what's the dynamic — growth or decline", "are there outliers and what explains them", "which 20% produce 80% of the result", "what data is missing for a confident conclusion". The last question is the most valuable: it protects you from hasty conclusions.

Hygiene rules. Remove personal data before pasting into a chat (lesson 16). Always ask for the calculation to be shown step by step — that's how you check the arithmetic. And remember: the model calculates excellently from data that's in the prompt, but doesn't "see" your file on the computer until you paste it or connect it.

> 💡 If a formula produced an error in the table, copy the error text and paste it to the model: "here's the formula and here's the error — fix it". It's faster than figuring it out yourself.

QUIZ

What do you need to describe to the model for it to write a formula?

- ○ The file name
- ● **The table's structure (what data is in which columns) and what needs to be calculated ✓**
- ○ The Windows version

Why: The model doesn't see your spreadsheet; an exact description of the columns and the task is the basis of a correct formula.

What does "dirty data" mean?

- ● **The same values in different formats: casing, spaces, dates ✓**
- ○ Data about defective goods
- ○ Data with viruses

Why: Inconsistent formats interfere with counting; cleaning brings values to a single form.

Why ask the model "point out where the data is insufficient for conclusions"?

- ○ The model is obligated to do it itself
- ○ To make the answer longer
- ● **It's protection against hasty conclusions on incomplete data ✓**

Why: An explicit question about gaps in the data is the cheapest way not to draw a false conclusion.

The model calculates excellently from data that...

- ○ Is on your computer
- ● **Is pasted into the prompt or connected to the chat ✓**
- ○ You'll think of later

Why: The model works only with what's in the context: the file needs to be pasted or connected.

Which question to a table is a typical analytical one?

- ○ "Rewrite the table nicely"

- ○ "Do you like my table?"
- ● **"Which 20% of items produce 80% of the result?" ✓**

Why: A question about result concentration (the Pareto principle) is a classic of data analysis.

A formula in the table produced an error. What's the first step?

- ○ Rewrite the table from scratch
- ● **Paste the error text and the formula to the model and ask for a fix ✓**
- ○ Delete the broken column

Why: The error text contains the cause; passing it to the model is the fastest repair.

Why ask "show the calculation step by step"?

- ● **So you can check the arithmetic and the logic ✓**
- ○ Without it the model refuses to calculate
- ○ For reporting purposes

Why: A step-by-step calculation turns a black box into a verifiable chain.

Before pasting a client spreadsheet into a public chat you need to...

- ○ Encrypt the table with a password
- ○ Nothing, it's safe
- ● **Remove or anonymize clients' personal data ✓**

Why: Personal data must not end up in public bots — lesson 16 covers this in detail.

HOMEWORK

Take a real spreadsheet (sales, expenses, a task list, a personal budget). Remove personal data. Do three things: (1) ask for a cleaning rule for the "dirtiest" column, (2) get 2–3 formulas from a description in words, (3) ask at least 3 analytical questions with a request to show the calculation step by step.

CHECKLIST

- ☐ The spreadsheet is real, personal data removed or anonymized
- ☐ For the "dirty" column, a cleaning rule was obtained (and a formula, if applicable)
- ☐ At least 2 formulas were obtained from a description and tested in the spreadsheet

- ☐ 3+ analytical questions were asked, calculations shown step by step, conclusions written down

LESSON 14

Automating routine: from finding tasks to no-code connections

Automation starts not with tools but with finding candidates. A simple filter: frequency × time. List the tasks you do regularly and estimate: how many times a week and how many minutes each time. Anything that repeats often and eats 10+ minutes is a candidate. The "two weeks" rule: if a task will come back at least 5 times — a template or a connection will pay off.

An important caveat: not everything is worth automating. Tasks with a high cost of error (payments, legal filings), unique tasks, and those where human contact matters — keep for yourself. Automate drafts and preparation, and leave the decision and the sending to a human: "AI prepares — a human checks and presses the button".

The first level of automation is the templates you already know (lesson 5). An email-to-client template, a review-reply template, a weekly summary template. This is "semi-automatic": you fill in the variables, the model does the rest. That's enough to cut 60–80% of the time on typical texts.

EXAMPLE PROMPT

```
"Assemble a weekly summary for the manager using the template: "Week {number}: highlights — {3 facts with figures}; problems — {what went wrong}; solutions — {what we're doing}; needed from you — {which decisions}." Keep a businesslike tone, maximum 10 lines, no evaluative words".
```

The second level — automation-builder services: n8n, Make, Zapier and local alternatives. The logic is simple: "when A happens — do B". For example: a new request arrives from a form → the request text goes to the AI → the model classifies it and puts a draft reply into a spreadsheet or chat. You assemble this from ready blocks, no programming — like a circuit from building blocks.

EXAMPLE SCHEME FOR A BUILDER

```
Trigger: a new row in a Google Form of requests. Step 2: send the request text to the AI with the prompt: "Classify the request: urgent/regular, return JSON {"category": "...", "urgency": "...", "draft_reply": "..."}". Step 3: write the result to a spreadsheet. Step 4: if urgent — a notification in chat
```

The third scenario — a chatbot for typical questions. Collect the 10–20 most frequent customer (or colleague) questions and the answers to them. This makes a knowledge base for a bot: it answers the typical ones right away and passes the complex ones to you. Start small: one channel, one scenario, manual handling of exceptions.

> 💡 In any connection, keep a human in the loop: let the automation prepare the draft, and the sending stays with you. It protects against mass mistakes.

QUIZ

By which formula do you find candidate tasks for automation?

- [x] **Frequency × time ✓**
- [ ] Price × risk
- [ ] Complexity × beauty

Why: Frequently repeated and long tasks give maximum savings with minimal setup.

What should you NOT automate first?

- [ ] A weekly summary
- [ ] Answers to typical questions
- [x] **Tasks with a high cost of error and unique decisions ✓**

Why: Payments, legal matters and nonstandard decisions are left to a human; draft routine gets automated.

What is a no-code connection like "form → AI → spreadsheet"?

- [ ] A Python program
- [x] **Ready blocks in a builder (n8n, Make, Zapier): trigger → processing → result ✓**
- [ ] A macro in Word

Why: Builders are assembled from blocks: a trigger, an AI step, writing the result — no programming.

The correct order for introducing automation?

- [ ] Bot first, understanding the task later
- [ ] Set up the connection right away
- [x] **A week doing it by hand with notes on the steps → a template → automation ✓**

Why: First learn the task by hand, then a template, then automation — otherwise you automate chaos.

Why keep a "human in the loop" in automation?

- [ ] So the work looks busy

- [x] Automation prepares the draft, and a human does the checking and sending — protection from mass mistakes ✓
- [ ] The law requires it

Why: One prompt error in a mass send multiplies across all messages; a human is the fuse.

For a chatbot of typical questions you first need to...

- [x] Collect 10–20 frequent questions and their answers ✓
- [ ] Learn to program
- [ ] Buy an expensive subscription

Why: A base of typical questions and answers is the heart of the bot; it's built from real inquiries.

Setup takes 4 hours, savings — 20 minutes a week. The conclusion?

- [ ] Automate and make it even more complex
- [ ] Definitely set it up
- [x] Maybe a template is enough — payback takes over a month ✓

Why: Count payback honestly; sometimes a simple template gives 80% of the benefit for 1% of the effort.

What does AI do in a request-classification connection?

- [ ] Answers clients without oversight
- [x] Returns a structure: category, urgency, draft reply — in JSON for the next step ✓
- [ ] Types the request for the client

Why: A link with lesson 7: strict JSON lets the next builder block write the data into a spreadsheet.

HOMEWORK

Pick one regular routine task from your work. Keep a "journal" of it for a week: how many times and how many minutes, what steps, what decisions. Then build an instruction template (lesson 5) with a prompt for this task and calculate the honest payback. Describe in 3–4 steps what a builder connection for this task would look like.

CHECKLIST

- ☐ The task is real and regular, with a journal of repeats and time kept
- ☐ A working instruction template with variables was built

- ☐ Payback was calculated: setup ÷ (minutes × repeats)
- ☐ A hypothetical connection is described: trigger → AI step → result, with a human in the loop

LESSON 15

Hallucinations and fact-checking

A hallucination is when the model confidently reports something incorrect, having invented a plausible version. This isn't a "bug" that will be fixed soon — it's a property of generation: the model builds a probable continuation, not a verified truth. Your job is to know where it errs most often and to check exactly there.

The model lies most often in: numbers and statistics ("73% according to studies..."), dates and chronology, direct quotes and authorship, links (plausible nonexistent addresses exist), law and article numbers, book and movie titles, the current state of the world (prices, exchange rates, news after the model's training). Anything requiring precise memory of identifiers is a risk zone.

Notice the pattern: precise identifier facts (an article number, a URL, a year) — risky; general understanding of a topic and working with your text — reliable. Not because the model "gets smarter", but because predicting general text is easier than recalling a precise identifier.

EXAMPLE PROMPT

```
"Answer the question: what penalties are provided for this violation? For each statement, indicate your confidence level: sure / not sure. If not sure — don't give the article number or exact amounts, but say where to check".
```

EXAMPLE PROMPT

```
"List 5 statements from your previous answer that you're least confident about, and explain why. Don't soften it: I need an honest assessment, not reassurance".
```

Verification techniques. First — another chat or another model: an independent answer lowers the chance of the same fabrication. Second — a source: a search engine or an official site. Third — a "cross-examination": ask the model to find the weak spots of its own answer and list what exactly needs checking by hand.

A practical scale. Can be trusted almost always: rephrasing, explanations, work with your text, structuring, ideas and drafts. Spot-check: calculations on your data, generalizations over tables. Always check: numbers, dates, quotes, links, laws, specific legal and medical advice. Don't trust without a source: "fresh" news, statistics "from memory", exact prices.

> 💡 Ask the model to indicate which fragment of your text each statement comes from: a fabrication finds it harder to hide when every statement must have an address in the source.

QUIZ

What is a neural network hallucination?

- ○ An overly long answer
- ○ A server connection error
- ● **Confidently reporting incorrect but plausible information ✓**

Why: The model generates probable text, not verified text: the plausible may be a fabrication.

Where does the model err most often?

- ○ In rephrasing your text
- ● **In precise identifiers: numbers, dates, links, law numbers ✓**
- ○ In structuring lists

Why: A precise identifier requires precise memory; general text is easier to predict.

Why is rephrasing your text a reliable zone?

- ○ Your text is always correct
- ○ The model turns off fabrications in this mode
- ● **Here the model doesn't generate anything new — it works with the text it was given ✓**

Why: When the source is in context, the model doesn't need to recall: it transforms what it sees.

How does the "another chat" technique work?

- ○ It's a backup copy
- ● **An independent answer from a second model lowers the chance of the same fabrication ✓**
- ○ The second chat answers faster

Why: Two independent generations rarely fabricate the same thing — a divergence signals risk.

Which prompt makes the model more honest?

- ● **"If you're not sure — say so directly, don't make it up" ✓**
- ○ "Answer briefly"
- ○ "Answer more confidently"

Why: Permission to be unsure + a ban on fabrication shift the behavior toward flagging doubtful things.

The model gave a link to a study. What to do?

- ○ Ask for more links

- ○ Paste the link into the document
- ● **Verify: open the link; nonexistent links are a classic hallucination ✓**

Why: Links are a red zone: a plausible address may not exist.

Which zone is "always check"?

- ○ Ideas for posts
- ● **Quotes, links, laws, specific legal and medical advice ✓**
- ○ The style of the text

Why: Where an error is costly and the fact is precise, checking is mandatory regardless of the confident tone.

The model claims: "according to the latest data, the rate grew by 4%". Risk category?

- ● **Red zone: fresh figures "from memory" — always check ✓**
- ○ It's a calculation on your data
- ○ Can be trusted

Why: Current figures without a source are a typical fabrication: the model doesn't know the present moment.

HOMEWORK

Run a "trust audit". Ask the model to answer 5 factual questions from your field (dates, figures, laws, links). Then: ask the model to mark a confidence level for each answer, verify all the answers against independent sources, and write up a table: the model's answer → the model's confidence → the truth per the source.

CHECKLIST

- ☐ 5+ factual questions from your real field were asked
- ☐ The model marked a confidence level for each answer
- ☐ All answers were verified against independent sources
- ☐ A comparison table was made and a conclusion written down: which types of questions your model handles more reliably

LESSON 16

Safety and privacy: a checklist before sending

Everything you send to a public chatbot goes to the developer company's servers. By default, don't send there: customers' passport data and documents, financial details, medical information, trade secrets (prices, client databases, non-public plans), passwords and access credentials, and third parties' personal data without their consent.

Good news: the model almost never needs real names to do its job. Replace "Irene Smith, tax ID 123456789, salary 145,000" with "client A., tax ID — 10 digits, salary — about 150K". The model will do the task just as well, and there's no risk. Anonymization is a one-minute habit that removes 90% of the risks.

EXAMPLE PROMPT

```
"Client A., 34 years old, self-employed, income about 120K a month, two children. I need to draft a letter about changes to the terms of service. (All data anonymized — real names, tax IDs and contacts were not pasted into the prompt)"
```

The second layer of protection — plans and settings. Most services have data-usage settings: in free versions conversations are more often used to train models; in paid ones you can usually turn off saving and training (in history or privacy settings). It's worth going into the settings once and toggling this deliberately. Corporate plans give additional guarantees — for work tasks, check the rules with your company.

The third layer — tool discipline. Don't paste into the chat files you don't have the rights to transfer. Don't mix work and personal accounts without understanding the rules. And be careful with "connect data": before connecting a cloud drive or email to a bot, understand what the service will get access to.

Four questions in ten seconds: (1) Is there personal data here — anyone's names, documents, contacts? (2) Is there commercially sensitive stuff — prices, databases, plans, account details? (3) What happens if a stranger sees this text? (4) Can the real thing be replaced with an anonymized version while keeping the task? If the first two are "yes" — anonymize or don't send.

EXAMPLE PROMPT

```
"Check my draft before sending to a client: find in it any personal data, account details and commercially sensitive information, and suggest how to anonymize or remove them without losing the meaning. Draft: "…"
```

> 💡 Make anonymization an automatic habit: before Ctrl+V — one glance at the text. One eye-run of "is anyone else's data here?" — and you're fine.

QUIZ

Which of these should you NOT send to a public chatbot?

- ○ An anonymized description of the task
- ● **Customers' passport data ✓**
- ○ An abstract question about text style

Why: Customers' personal data is a forbidden category for public bots: it's both a leak risk and a legal violation.

Why does anonymization almost never hurt task quality?

- ○ Anonymization speeds up the answer
- ○ The model doesn't notice the replacements
- ● **The model usually doesn't need real names and details — the structure of the situation is what matters ✓**

Why: "Client A., income about 120K" carries the same meaning as the real data.

Where is the use of your data for training usually configured?

- ○ In the browser settings
- ● **In the service's privacy/history settings ✓**
- ○ It's impossible to turn off anywhere

Why: Major services have switches for history and training on data; they should be set deliberately.

How do paid plans usually differ in terms of privacy?

- ● **You can more often turn off conversation saving and use for training ✓**
- ○ Paid users' data is sold at a higher price
- ○ No difference

Why: Paid plans usually offer privacy options and corporate guarantees.

The second question of the pre-send checklist:

- ○ Is the prompt long enough?
- ○ Does the prompt look nice?
- ● **Is there anything commercially sensitive — prices, databases, plans, account details? ✓**

Why: Trade secrets are the second forbidden category after personal data.

What to do before connecting email or a cloud drive to an AI service?

- ○ Nothing, it's always safe
- ● **Understand exactly what the service will get access to and why it needs it ✓**
- ○ Connect all the folders at once

Why: Access rights are a risk too; connect the minimum necessary.

A colleague asks you to paste a client database with phone numbers into the chat "just for analysis". Your action?

- ● **Anonymize it or process it with company-approved tools; remove the phone numbers ✓**
- ○ Paste only half the database
- ○ Paste it, it's convenient

Why: Third parties' personal data must not end up in a public bot in any volume.

Why ask "what if a stranger sees this text"?

- ○ It makes writing prompts easier
- ○ It's paranoia
- ● **It instantly assesses the cost of a leak of the prompt's contents ✓**

Why: The thought experiment with a stranger reading is the fastest test of content sensitivity.

HOMEWORK

Run a "privacy audit" of your AI use. Recall the last 5–10 prompts you sent (or look through the history). Note whether there was personal data, account details or commercially sensitive content. Anonymize 2 real past prompts and compare: did the answer quality suffer? Check and configure privacy in your AI service.

CHECKLIST

- ☐ At least 5 past prompts were reviewed, sensitive spots noted
- ☐ 2 prompts were anonymized and sent again — answer quality was compared
- ☐ Privacy in the AI service was checked: history/training configured deliberately
- ☐ A personal checklist of 4 pre-send questions was written down — and placed somewhere visible

What's next

This book is the print edition of the interactive course “AI for beginners” by Krok AI. In the interactive version you answer questions right in your browser, track your progress lesson by lesson and get instant feedback on every answer.

Take the course online — it's free and takes about an hour:

Take the interactive course →

