Guidelines for Prompting

In this lesson, you'll practice two prompting principles and their related tactics in order to write effective prompt for large language models.

Setup

Load the API key and relevant Python libaries.

In this course, we've provided some code that loads the OpenAI API key for you.

In [6]:
```
import openai
import os

from dotenv import load_dotenv, find_dotenv
_ = load_dotenv(find_dotenv())

openai.api_key  = os.getenv('OPENAI_API_KEY')
print("Executed")
```

```
Executed
```

helper function

Throughout this course, we will use OpenAI's `gpt-3.5-turbo` model and the chat completions endpoint (https://platform.openai.com/docs/guides/chat).

This helper function will make it easier to use prompts and look at the generated outputs.
Note: In June 2023, OpenAI updated gpt-3.5-turbo. The results you see in the notebook may be slightly different than those in the video. Some of the prompts have also been slightly modified to product the desired results.

In [7]:
```
def get_completion(prompt, model="gpt-3.5-turbo"):
    messages = [{"role": "user", "content": prompt}]
    response = openai.ChatCompletion.create(
        model=model,
        messages=messages,
        temperature=0, # this is the degree of randomness of the model's ou
    )
    return response.choices[0].message["content"]
```

Note: This and all other lab notebooks of this course use OpenAI library version `0.27.0`.

In order to use the OpenAI library version `1.0.0`, here is the code that you would use instead for the `get_completion` function:

```
client = openai.OpenAI()

def get_completion(prompt, model="gpt-3.5-turbo"):
    messages = [{"role": "user", "content": prompt}]
    response = client.chat.completions.create(
        model=model,
        messages=messages,
        temperature=0
    )
```

Prompting Principles

- **Principle 1: Write clear and specific instructions**
- **Principle 2: Give the model time to "think"**

Tactics

Tactic 1: Use delimiters to clearly indicate distinct parts of the input

- Delimiters can be anything like: ```, """, < >, <tag> </tag> , :

In [8]:
```
text = f"""
You should express what you want a model to do by \
providing instructions that are as clear and \
specific as you can possibly make them. \
This will guide the model towards the desired output, \
and reduce the chances of receiving irrelevant \
or incorrect responses. Don't confuse writing a \
clear prompt with writing a short prompt. \
In many cases, longer prompts provide more clarity \
and context for the model, which can lead to \
more detailed and relevant outputs.
"""
prompt = f"""
Summarize the text delimited by triple backticks \
into a single sentence.
```{text}```
"""
response = get_completion(prompt)
print(response)
```

```
Providing clear and specific instructions to a model will guide it towards t
he desired output and reduce the chances of receiving irrelevant or incorrec
t responses, with longer prompts often providing more clarity and context fo
r more detailed and relevant outputs.
```

Tactic 2: Ask for a structured output

- JSON, HTML
```

In [9]:
```
prompt = f"""
Generate a list of three made-up book titles along \
with their authors and genres.
Provide them in JSON format with the following keys:
book_id, title, author, genre.
"""
response = get_completion(prompt)
print(response)
```

```
[
    {
        "book_id": 1,
        "title": "The Midnight Garden",
        "author": "Elena Rivers",
        "genre": "Fantasy"
    },
    {
        "book_id": 2,
        "title": "Whispers in the Wind",
        "author": "Lucas Blackwood",
        "genre": "Mystery"
    },
    {
        "book_id": 3,
        "title": "Echoes of the Past",
        "author": "Samantha Greene",
        "genre": "Historical Fiction"
    }
]
```

Tactic 3: Ask the model to check whether conditions are satisfied

In [12]:
```
text_1 = f"""
Making a cup of tea is easy! First, you need to get some \
water boiling. While that's happening, \
grab a cup and put a tea bag in it. Once the water is \
hot enough, just pour it over the tea bag. \
Let it sit for a bit so the tea can steep. After a \
few minutes, take out the tea bag. If you \
like, you can add some sugar or milk to taste. \
And that's it! You've got yourself a delicious \
cup of tea to enjoy.
"""
prompt = f"""
You will be provided with text delimited by triple quotes.
If it contains a sequence of instructions, \
re-write those instructions in the following format:

Step 1 - ...
Step 2 - …
…
Step N - …

If the text does not contain a sequence of instructions, \
then simply write \"No steps provided.\"

\"\"\"{text_1}\"\"\"
"""
response = get_completion(prompt)
print("Completion for Text 1:")
print(response)
```

```
Completion for Text 1:
Step 1 - Get some water boiling.
Step 2 - Grab a cup and put a tea bag in it.
Step 3 - Pour the hot water over the tea bag.
Step 4 - Let the tea steep for a bit.
Step 5 - Remove the tea bag.
Step 6 - Add sugar or milk to taste.
Step 7 - Enjoy your delicious cup of tea.
```

In [13]:
```
text_2 = f"""
The sun is shining brightly today, and the birds are \
singing. It's a beautiful day to go for a \
walk in the park. The flowers are blooming, and the \
trees are swaying gently in the breeze. People \
are out and about, enjoying the lovely weather. \
Some are having picnics, while others are playing \
games or simply relaxing on the grass. It's a \
perfect day to spend time outdoors and appreciate the \
beauty of nature.
"""
prompt = f"""
You will be provided with text delimited by triple quotes.
If it contains a sequence of instructions, \
re-write those instructions in the following format:

Step 1 - ...
Step 2 - …
…
Step N - …

If the text does not contain a sequence of instructions, \
then simply write \"No steps provided.\"

\"\"\"{text_2}\"\"\"
"""
response = get_completion(prompt)
print("Completion for Text 2:")
print(response)
```

```
Completion for Text 2:
No steps provided.
```

Tactic 4: "Few-shot" prompting

In [ ]:
```
prompt = f"""
Your task is to answer in a consistent style.

<child>: Teach me about patience.

<grandparent>: The river that carves the deepest \
valley flows from a modest spring; the \
grandest symphony originates from a single note; \
the most intricate tapestry begins with a solitary thread.

<child>: Teach me about resilience.
"""
response = get_completion(prompt)
print(response)
```

Principle 2: Give the model time to “think”

In [14]:
```
text = f"""
In a charming village, siblings Jack and Jill set out on \
a quest to fetch water from a hilltop \
well. As they climbed, singing joyfully, misfortune \
struck—Jack tripped on a stone and tumbled \
down the hill, with Jill following suit. \
Though slightly battered, the pair returned home to \
comforting embraces. Despite the mishap, \
their adventurous spirits remained undimmed, and they \
continued exploring with delight.
"""
# example 1
prompt_1 = f"""
Perform the following actions:
1 - Summarize the following text delimited by triple \
backticks with 1 sentence.
2 - Translate the summary into French.
3 - List each name in the French summary.
4 - Output a json object that contains the following \
keys: french_summary, num_names.

Separate your answers with line breaks.

Text:
```{text}```
"""
response = get_completion(prompt_1)
print("Completion for prompt 1:")
print(response)
```

```
Completion for prompt 1:
1 - Jack and Jill, siblings from a charming village, go on a quest to fetch
water from a hilltop well, but encounter misfortune along the way.

2 - Jack et Jill, frère et sœur d'un charmant village, partent en quête d'ea
u d'un puits au sommet d'une colline, mais rencontrent des malheurs en chemi
n.

3 - Jack, Jill

4 -
{
  "french_summary": "Jack et Jill, frère et sœur d'un charmant village, part
ent en quête d'eau d'un puits au sommet d'une colline, mais rencontrent des
malheurs en chemin.",
  "num_names": 2
}
```

Ask for output in a specified format
```

In [20]:
```
prompt_2 = f"""
Your task is to perform the following actions:
1 - Write lyrics for a song in a movie.
2 - The plot is the protagonist is a reformed an and now he was sworn to ch
and lead a better life.As thy are reluctant and they are very brave. He dec
leave everything and reform.
3 - The song should have 3 verses.
4 - Each verse and line should have poetic and music rythmic touch.
5 - Translate the song into Telugu.
6- Output a json object that contains the
  following keys: song_lyrics,Telugu_translated_song.

Use the following format:
Text: <lyrics>
Translation: < translation>

Text: <{text}>
"""
response = get_completion(prompt_2)
print("\nCompletion for prompt 2:")
print(response)
```

```
Completion for prompt 2:
{
  "song_lyrics": {
    "verse_1": "In the heart of the ocean, a pirate once roamed, \nWith a cr
ew of brave souls, their hearts made of stone. \nBut a change was afoot, a r
eformed man he became, \nTo lead his fellow pirates out of the dark, into th
e flame.",
    "verse_2": "They scoffed and they laughed, at his newfound ways, \nBut d
eep down inside, they longed for better days. \nHe became their fear, the on
e they couldn't ignore, \nTo leave behind their past, and seek something mor
e.",
    "verse_3": "Through storms and through battles, they sailed the rough se
as, \nWith courage and strength, they fought to be free. \nAnd in the end, t
hey found a new life to embrace, \nAll thanks to the pirate who led them to
grace."
  },
  "Telugu_translated_song": {
    "verse_1": "సముద్ర హృదయంలో, ఒక దొరికిన దొరలు పాయిరేట్ ఒకసారి రామ్,
\nతమ హృదయాలు కల్లెదుట తయారుగా ఉండిన ఒక దళం. \nకాని మారిపోయింది, మాన
సికంగా పరివర్తన మనిగా, \nతన సహపాఠులను కారకంగా తెలియజేసేందుకు అందించ
డానికి వెళ్ళాడు.",
    "verse_2": "వాడు అవినీతంగా చేస్తున్నారు, వాడి కొండలో నేటి రహస్యాలు, \nకాని
అందరు అందరు లోపల ఉండారు. \nవాడు అవినీతంగా ఉండిన వాడు, వాడు అ
వినీతంగా ఉండిన వాడు, \nవాడు అవినీతంగా ఉండిన వాడు, వాడు అవినీతంగా ఉండిన
వాడు.",
    "verse_3": "తుఫానులు మరియు యుద్ధాలు దాటి, వారు కఠినమైన సముద్రాలను
పారుగా, \nధైర్యం మరియు శక్తితో, వారు ఉచితంగా ఉండటం కోసం పోరాడారు. \nమరియు
చివరిలో, వారు ఆనందంగా అన్వేషించడం కోసం కొనిస్తున్నారు."
  }
}
```

Tactic 2: Instruct the model to work out its own solution before rushing to a conclusion

In [16]:
```
prompt = f"""
Determine if the student's solution is correct or not.

Question:
I'm building a solar power installation and I need \
 help working out the financials.
- Land costs $100 / square foot
- I can buy solar panels for $250 / square foot
- I negotiated a contract for maintenance that will cost \
me a flat $100k per year, and an additional $10 / square \
foot
What is the total cost for the first year of operations
as a function of the number of square feet.

Student's Solution:
Let x be the size of the installation in square feet.
Costs:
1. Land cost: 100x
2. Solar panel cost: 250x
3. Maintenance cost: 100,000 + 100x
Total cost: 100x + 250x + 100,000 + 100x = 450x + 100,000
"""
response = get_completion(prompt)
print(response)
```

```
The student's solution is correct. The total cost for the first year of oper
ations as a function of the number of square feet is indeed 450x + 100,000.
```

Note that the student's solution is actually not correct.

We can fix this by instructing the model to work out its own solution first.

In [17]:
```
prompt = f"""
Your task is to determine if the student's solution \
is correct or not.
To solve the problem do the following:
- First, work out your own solution to the problem including the final tot
- Then compare your solution to the student's solution \
and evaluate if the student's solution is correct or not.
Don't decide if the student's solution is correct until
you have done the problem yourself.

Use the following format:
Question:
```
question here
```
Student's solution:
```
student's solution here
```
Actual solution:
```
steps to work out the solution and your solution here
```
Is the student's solution the same as actual solution \
just calculated:
```
yes or no
```
Student grade:
```
correct or incorrect
```


Question:
```
I'm building a solar power installation and I need help \
working out the financials.
- Land costs $100 / square foot
- I can buy solar panels for $250 / square foot
- I negotiated a contract for maintenance that will cost \
me a flat $100k per year, and an additional $10 / square \
foot
What is the total cost for the first year of operations \
as a function of the number of square feet.
```
Student's solution:
```
Let x be the size of the installation in square feet.
Costs:
1. Land cost: 100x
2. Solar panel cost: 250x
3. Maintenance cost: 100,000 + 100x
Total cost: 100x + 250x + 100,000 + 100x = 450x + 100,000
```
Actual solution:
"""
response = get_completion(prompt)
```

```
print(response)
```

```
Let x be the size of the installation in square feet.

Costs:
1. Land cost: $100 * x
2. Solar panel cost: $250 * x
3. Maintenance cost: $100,000 + $10 * x

Total cost: $100 * x + $250 * x + $100,000 + $10 * x = $350 * x + $100,000

The total cost for the first year of operations as a function of the number
of square feet is $350x + $100,000.
```

```
Is the student's solution the same as actual solution just calculated:
```

```
Yes
```

```
Student grade:
```

```
Correct
```

Model Limitations: Hallucinations

- Boie is a real company, the product name is not real.

In [19]:
```
prompt = f"""
Teach me how to direct a film
"""
response = get_completion(prompt)
print(response)
```

```
Directing a film is a complex and multi-faceted process that requires a comb
ination of technical skills, creative vision, and strong leadership abilitie
s. Here are some steps to help you learn how to direct a film:

1. Study the craft: Start by watching a wide variety of films from different
genres and time periods to gain an understanding of different styles of film
making. You can also read books on film theory and take classes or workshops
on directing.

2. Learn the technical aspects: Familiarize yourself with the technical aspe
cts of filmmaking, such as camera angles, lighting, sound design, and editin
g. You may want to take classes or workshops on these topics to deepen your
knowledge.

3. Develop a vision: Before you start filming, it's important to have a clea
r vision for your film. This includes deciding on the overall tone, style, a
nd message you want to convey. You may want to create a storyboard or shot l
ist to help you visualize your ideas.

4. Work with a team: Directing a film is a collaborative process, so it's im
portant to work closely with your cast and crew. Communicate your vision cle
arly and listen to their input and ideas. A strong director is able to inspi
re and motivate their team to do their best work.

5. Plan and organize: Before you start filming, create a detailed production
schedule and budget. Make sure you have all the necessary equipment and reso
urces in place. It's also important to plan out each shot and scene in advan
ce to ensure a smooth filming process.

6. Direct the actors: One of the most important aspects of directing is work
ing with the actors to bring your characters to life. Provide clear directio
n and feedback, but also allow them the freedom to explore their characters
and make creative choices.

7. Stay flexible: Filmmaking is a dynamic and unpredictable process, so it's
important to be flexible and adaptable. Be prepared to make changes on the f
ly and problem-solve any issues that arise during filming.

8. Edit and post-production: Once filming is complete, work closely with you
r editor to assemble the footage into a cohesive and engaging film. Pay atte
ntion to pacing, sound design, and visual effects to enhance the overall imp
act of your film.

9. Seek feedback: Show your film to friends, colleagues, and industry profes
sionals to get feedback and constructive criticism. Use this feedback to lea
rn and improve your skills as a director.

10. Keep learning: Directing is a lifelong learning process, so continue to
watch films, study the work of other directors, and seek out opportunities t
o practice and refine your craft. With dedication and hard work, you can bec
ome a skilled and successful film director.
```

Try experimenting on your own!

In [ ]:

Notes on using the OpenAI API outside of this classroom

To install the OpenAI Python library:

```
!pip install openai
```

The library needs to be configured with your account's secret key, which is available on the website (https://platform.openai.com/account/api-keys).

You can either set it as the `OPENAI_API_KEY` environment variable before using the library:

```
!export OPENAI_API_KEY='sk-...'
```

Or, set `openai.api_key` to its value:

```
import openai
openai.api_key = "sk-..."
```

A note about the backslash

- In the course, we are using a backslash `\` to make the text fit on the screen without inserting newline '\n' characters.
- GPT-3 isn't really affected whether you insert newline characters or not. But when working with LLMs in general, you may consider whether newline characters in your prompt may affect the model's performanc

In [ ]: