第二章 提示原则

如何去使用 Prompt，以充分发挥 LLM 的性能？首先我们需要知道设计 Prompt 的原则，它们是每一个开发者设计 Prompt 所必须知道的基础概念。本章讨论了设计高效 Prompt 的两个关键原则：**编写清晰、具体的指令**和**给予模型充足思考时间**。掌握这两点，对创建可靠的语言模型交互尤为重要。

首先，Prompt 需要清晰明确地表达需求，提供充足上下文，使语言模型准确理解我们的意图，就像向一个外星人详细解释人类世界一样。过于简略的 Prompt 往往使模型难以把握所要完成的具体任务。

其次，让语言模型有充足时间推理也极为关键。就像人类解题一样，匆忙得出的结论多有失误。因此 Prompt 应加入逐步推理的要求，给模型留出充分思考时间，这样生成的结果才更准确可靠。

如果 Prompt 在这两点上都作了优化，语言模型就能够尽可能发挥潜力，完成复杂的推理和生成任务。掌握这些 Prompt 设计原则，是开发者取得语言模型应用成功的重要一步。

一、原则一 编写清晰、具体的指令

亲爱的读者，在与语言模型交互时，您需要牢记一点:以**清晰、具体**的方式表达您的需求。假设您面前坐着一位来自外星球的新朋友，其对人类语言和常识都一无所知。在这种情况下，您需要把想表达的意图讲得非常明确，不要有任何歧义。同样的，在提供 Prompt 的时候，也要以足够详细和容易理解的方式，把您的需求与上下文说清楚。

并不是说 Prompt 就必须非常短小简洁。事实上，在许多情况下，更长、更复杂的 Prompt 反而会让语言模型更容易抓住关键点，给出符合预期的回复。原因在于，复杂的 Prompt 提供了更丰富的上下文和细节，让模型可以更准确地把握所需的操作和响应方式。

所以，记住用清晰、详尽的语言表达 Prompt，就像在给外星人讲解人类世界一样，*“Adding more context helps the model understand you better.”*。

从该原则出发，我们提供几个设计 Prompt 的技巧。

1.1 使用分隔符清晰地表示输入的不同部分

在编写 Prompt 时，我们可以使用各种标点符号作为“分隔符”，将不同的文本部分区分开来。

分隔符就像是 Prompt 中的墙，将不同的指令、上下文、输入隔开，避免意外的混淆。你可以选择用 `` ```，"""，< >，<tag> </tag>，: `` 等做分隔符，只要能明确起到隔断作用即可。

使用分隔符尤其重要的是可以防止 **提示词注入（Prompt Rejection）**。什么是提示词注入？就是用户输入的文本可能包含与你的预设 Prompt 相冲突的内容，如果不加分隔，这些输入就可能“注入”并操纵语言模型，导致模型产生毫无关联的乱七八糟的输出。

在以下的例子中，我们给出一段话并要求 GPT 进行总结，在该示例中我们使用 ``` 来作为分隔符。

```
from tool import get_completion

text = f"""
您应该提供尽可能清晰、具体的指示，以表达您希望模型执行的任务。\
这将引导模型朝向所需的输出，并降低收到无关或不正确响应的可能性。\
不要将写清晰的提示词与写简短的提示词混淆。\
在许多情况下，更长的提示词可以为模型提供更多的清晰度和上下文信息，从而导致更详细和相关的输出。
"""
# 需要总结的文本内容
prompt = f"""
把用三个反引号括起来的文本总结成一句话。
```

```
```{text}```
"""
# 指令内容，使用 ``` 来分隔指令和待总结的内容
response = get_completion(prompt)
print(response)
```

为了获得所需的输出，您应该提供清晰、具体的指示，避免与简短的提示词混淆，并使用更长的提示词来提供更多的清晰度和上下文信息。

1.2 寻求结构化的输出

有时候我们需要语言模型给我们一些**结构化的输出**，而不仅仅是连续的文本。

什么是结构化输出呢？就是按照某种格式组织的内容，例如JSON、HTML等。这种输出非常适合在代码中进一步解析和处理。例如，您可以在 Python 中将其读入字典或列表中。

在以下示例中，我们要求 GPT 生成三本书的标题、作者和类别，并要求 GPT 以 JSON 的格式返回给我们，为便于解析，我们指定了 Json 的键。

```
prompt = f"""
请生成包括书名、作者和类别的三本虚构的、非真实存在的中文书籍清单，\
并以 JSON 格式提供，其中包含以下键:book_id、title、author、genre。
"""
response = get_completion(prompt)
print(response)
```

```
{
  "books": [
    {
      "book_id": 1,
      "title": "迷失的时光",
      "author": "张三",
      "genre": "科幻"
    },
    {
      "book_id": 2,
      "title": "幻境之门",
      "author": "李四",
      "genre": "奇幻"
    },
    {
      "book_id": 3,
      "title": "虚拟现实",
      "author": "王五",
      "genre": "科幻"
    }
  ]
}
```
```

1.3 要求模型检查是否满足条件

如果任务包含不一定能满足的假设（条件），我们可以告诉模型先检查这些假设，如果不满足，则会指出并停止执行后续的完整流程。您还可以考虑可能出现的边缘情况及模型的应对，以避免意外的结果或错误发生。

在如下示例中，我们将分别给模型两段文本，分别是制作茶的步骤以及一段没有明确步骤的文本。我们将要求模型判断其是否包含一系列指令，如果包含则按照给定格式重新编写指令，不包含则回答“未提供步骤”。

```
# 满足条件的输入（text中提供了步骤）
text_1 = f"""
泡一杯茶很容易。首先，需要把水烧开。\
在等待期间，拿一个杯子并把茶包放进去。\
一旦水足够热，就把它倒在茶包上。\
等待一会儿，让茶叶浸泡。几分钟后，取出茶包。\
如果您愿意，可以加一些糖或牛奶调味。\
就这样，您可以享受一杯美味的茶了。
"""
prompt = f"""
您将获得由三个引号括起来的文本。\
如果它包含一系列的指令，则需要按照以下格式重新编写这些指令：

第一步 - ...
第二步 - …
…
第N步 - …

如果文本中不包含一系列的指令，则直接写“未提供步骤”。"
\"\"\"{text_1}\"\"\"
"""
response = get_completion(prompt)
print("Text 1 的总结:")
print(response)
```

```
Text 1 的总结:
第一步 - 把水烧开。
第二步 - 拿一个杯子并把茶包放进去。
第三步 - 把烧开的水倒在茶包上。
第四步 - 等待几分钟，让茶叶浸泡。
第五步 - 取出茶包。
第六步 - 如果需要，加入糖或牛奶调味。
第七步 - 就这样，您可以享受一杯美味的茶了。
```

上述示例中，模型可以很好地识别一系列的指令并进行输出。在接下来一个示例中，我们将提供给模型没有预期指令的输入，模型将判断未提供步骤。

```
# 不满足条件的输入（text中未提供预期指令）
text_2 = f"""
今天阳光明媚，鸟儿在歌唱。\
这是一个去公园散步的美好日子。\
鲜花盛开，树枝在微风中轻轻摇曳。\
人们外出享受着这美好的天气，有些人在野餐，有些人在玩游戏或者在草地上放松。\
这是一个完美的日子，可以在户外度过并欣赏大自然的美景。
```

```
"""
prompt = f"""
您将获得由三个引号括起来的文本。\
如果它包含一系列的指令，则需要按照以下格式重新编写这些指令：

第一步 - ...
第二步 - …
…
第N步 - …

如果文本中不包含一系列的指令，则直接写“未提供步骤”。"
\"\"\"{text_2}\"\"\"
"""
response = get_completion(prompt)
print("Text 2 的总结:")
print(response)
```

```
Text 2 的总结:
未提供步骤。
```

1.4 提供少量示例

"Few-shot" prompting，即在要求模型执行实际任务之前，给模型一两个已完成的样例，让模型了解我们的要求和期望的输出样式。

例如，在以下的样例中，我们先给了一个祖孙对话样例，然后要求模型用同样的隐喻风格回答关于“韧性”的问题。这就是一个少样本样例，它能帮助模型快速抓住我们要的语调和风格。

利用少样本样例，我们可以轻松“预热”语言模型，让它为新的任务做好准备。这是一个让模型快速上手新任务的有效策略。

```
prompt = f"""
您的任务是以一致的风格回答问题。

<孩子>: 请教我何为耐心。

<祖父母>: 挖出最深峡谷的河流源于一处不起眼的泉眼；最宏伟的交响乐从单一的音符开始；最复杂的挂毯以一根孤独的线开始编织。

<孩子>: 请教我何为韧性。
"""
response = get_completion(prompt)
print(response)
```

```
<祖父母>: 韧性是一种坚持不懈的品质，就像一棵顽强的树在风雨中屹立不倒。它是面对困难和挑战时不屈不挠的精神，能够适应变化和克服逆境。韧性是一种内在的力量，让我们能够坚持追求目标，即使面临困难和挫折也能坚持不懈地努力。
```

二、原则二 给模型时间去思考

在设计 Prompt 时，给予语言模型充足的推理时间非常重要。语言模型与人类一样，需要时间来思考并解决复杂问题。如果让语言模型匆忙给出结论，其结果很可能不准确。例如，若要语言模型推断一本书的主题，仅提供简单的书名和一句简介是不足够的。这就像让一个人在极短时间内解决困难的数学题，错误在所难免。

相反，我们应通过 Prompt 指引语言模型进行深入思考。可以要求其先列出对问题的各种看法，说明推理依据，然后再得出最终结论。在 Prompt 中添加逐步推理的要求，能让语言模型投入更多时间逻辑思维，输出结果也将更可靠准确。

综上所述，给予语言模型充足的推理时间，是 Prompt Engineering 中一个非常重要的设计原则。这将大大提高语言模型处理复杂问题的效果，也是构建高质量 Prompt 的关键之处。开发者应注意给模型留出思考空间，以发挥语言模型的最大潜力。

2.1 指定完成任务所需的步骤

接下来我们将通过给定一个复杂任务，给出完成该任务的一系列步骤，来展示这一策略的效果。

首先我们描述了杰克和吉尔的故事，并给出提示词执行以下操作：首先，用一句话概括三个反引号限定的文本。第二，将摘要翻译成英语。第三，在英语摘要中列出每个名称。第四，输出包含以下键的 JSON 对象：英语摘要和人名个数。要求输出以换行符分隔。

```
text = f"""
在一个迷人的村庄里，兄妹杰克和吉尔出发去一个山顶井里打水。\
他们一边唱着欢乐的歌，一边往上爬，\
然而不幸降临——杰克绊了一块石头，从山上滚了下来，吉尔紧随其后。\
虽然略有些摔伤，但他们还是回到了温馨的家中。\
尽管出了这样的意外，他们的冒险精神依然没有减弱，继续充满愉悦地探索。
"""
# example 1
prompt_1 = f"""
执行以下操作：
1-用一句话概括下面用三个反引号括起来的文本。
2-将摘要翻译成英语。
3-在英语摘要中列出每个人名。
4-输出一个 JSON 对象，其中包含以下键：english_summary，num_names。

请用换行符分隔您的答案。

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

```
prompt 1:
1-两个兄妹在山上打水时发生意外，但最终平安回家。
2-In a charming village, siblings Jack and Jill set off to fetch water from a well on top of a hill. While singing joyfully, they climbed up, but unfortunately, Jack tripped on a stone and rolled down the hill, with Jill following closely behind. Despite some minor injuries, they made it back to their cozy home. Despite the mishap, their adventurous spirit remained undiminished as they continued to explore with delight.
3-Jack, Jill
4-{"english_summary": "In a charming village, siblings Jack and Jill set off to fetch water from a well on top of a hill. While singing joyfully, they climbed up, but unfortunately, Jack tripped on a stone and rolled down the hill, with Jill following closely behind. Despite some minor injuries, they made it back to their cozy home. Despite the mishap, their adventurous spirit remained undiminished as they continued to explore with delight.", "num_names": 2}
```

上述输出仍然存在一定问题，例如，键"姓名"会被替换为法语（译注：在英文原版中，要求从英语翻译到法语，对应指令第三步的输出为 'Noms:'，为Name的法语，这种行为难以预测，并可能为导出带来困难）

因此，我们将Prompt加以改进，该 Prompt 前半部分不变，同时**确切指定了输出的格式。**

```
prompt_2 = f"""
1-用一句话概括下面用<>括起来的文本。
2-将摘要翻译成英语。
3-在英语摘要中列出每个名称。
4-输出一个 JSON 对象，其中包含以下键：English_summary，num_names。

请使用以下格式：
文本：<要总结的文本>
摘要：<摘要>
翻译：<摘要的翻译>
名称：<英语摘要中的名称列表>
输出 JSON：<带有 English_summary 和 num_names 的 JSON>

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

```
prompt 2:
Summary: 在一个迷人的村庄里，兄妹杰克和吉尔在山顶井里打水时发生了意外，但他们的冒险精神依然没有减弱，继续充满愉悦地探索。

Translation: In a charming village, siblings Jack and Jill set off to fetch water from a well on top of a hill. Unfortunately, Jack tripped on a rock and tumbled down the hill, with Jill following closely behind. Despite some minor injuries, they made it back home safely. Despite the mishap, their adventurous spirit remained strong as they continued to explore joyfully.

Names: Jack, Jill

JSON Output: {"English_summary": "In a charming village, siblings Jack and Jill set off to fetch water from a well on top of a hill. Unfortunately, Jack tripped on a rock and tumbled down the hill, with Jill following closely behind. Despite some minor injuries, they made it back home safely. Despite the mishap, their adventurous spirit remained strong as they continued to explore joyfully.", "num_names": 2}
```

2.2 指导模型在下结论之前找出一个自己的解法

在设计 Prompt 时，我们还可以通过明确指导语言模型进行自主思考，来获得更好的效果。

举个例子，假设我们要语言模型判断一个数学问题的解答是否正确。仅仅提供问题和解答是不够的，语言模型可能会匆忙做出错误判断。

相反，我们可以在 Prompt 中先要求语言模型自己尝试解决这个问题，思考出自己的解法，然后再与提供的解答进行对比，判断正确性。这种先让语言模型自主思考的方式，能帮助它更深入理解问题，做出更准确的判断。

接下来我们会给出一个问题和一份来自学生的解答，要求模型判断解答是否正确：

```
prompt = f"""
判断学生的解决方案是否正确。

问题：
我正在建造一个太阳能发电站，需要帮助计算财务。

    土地费用为 100美元/平方英尺
    我可以以 250美元/平方英尺的价格购买太阳能电池板
    我已经谈判好了维护合同，每年需要支付固定的10万美元，并额外支付每平方英尺10美元
    作为平方英尺数的函数，首年运营的总费用是多少。

学生的解决方案：
设x为发电站的大小，单位为平方英尺。
费用：

    土地费用：100x
    太阳能电池板费用：250x
    维护费用：100,000美元+100x
    总费用：100x+250x+100,000美元+100x=450x+100,000美元
"""
response = get_completion(prompt)
print(response)
```

学生的解决方案是正确的。他正确地计算了土地费用、太阳能电池板费用和维护费用，并将它们相加得到了总费用。

但是注意，学生的解决方案实际上是错误的。（*维护费用项100x应为10x，总费用450x应为360x*）

我们可以通过指导模型先自行找出一个解法来解决这个问题。

在接下来这个 Prompt 中，我们要求模型先自行解决这个问题，再根据自己的解法与学生的解法进行对比，从而判断学生的解法是否正确。同时，我们给定了输出的格式要求。通过拆分任务、明确步骤，让模型有更多时间思考，有时可以获得更准确的结果。在这个例子中，学生的答案是错误的，但如果我们没有先让模型自己计算，那么可能会被误导以为学生是正确的。

```
prompt = f"""
请判断学生的解决方案是否正确，请通过如下步骤解决这个问题：

步骤：

    首先，自己解决问题。
    然后将您的解决方案与学生的解决方案进行比较，对比计算得到的总费用与学生计算的总费用是否一致，并评估学生的解决方案是否正确。
    在自己完成问题之前，请勿决定学生的解决方案是否正确。

使用以下格式：

    问题：问题文本
    学生的解决方案：学生的解决方案文本
    实际解决方案和步骤：实际解决方案和步骤文本
    学生计算的总费用：学生计算得到的总费用
    实际计算的总费用：实际计算出的总费用
    学生计算的费用和实际计算的费用是否相同：是或否
    学生的解决方案和实际解决方案是否相同：是或否
    学生的成绩：正确或不正确

问题：

    我正在建造一个太阳能发电站，需要帮助计算财务。
    - 土地费用为每平方英尺100美元
    - 我可以以每平方英尺250美元的价格购买太阳能电池板
    - 我已经谈判好了维护合同，每年需要支付固定的10万美元，并额外支付每平方英尺10美元;

    作为平方英尺数的函数，首年运营的总费用是多少。

学生的解决方案：

    设x为发电站的大小，单位为平方英尺。
    费用：
    1. 土地费用：100x美元
    2. 太阳能电池板费用：250x美元
    3. 维护费用：100,000+100x=10万美元+10x美元
    总费用：100x美元+250x美元+10万美元+100x美元=450x+10万美元

实际解决方案和步骤：
"""
response = get_completion(prompt)
```

```
print(response)
```

实际解决方案和步骤：

1. 土地费用：每平方英尺100美元，所以总费用为100x美元。
2. 太阳能电池板费用：每平方英尺250美元，所以总费用为250x美元。
3. 维护费用：固定费用为10万美元，额外费用为每平方英尺10美元，所以总费用为10万美元+10x美元。
4. 总费用：将上述三项费用相加，得到总费用为100x美元+250x美元+10万美元+10x美元=360x+10万美元。

学生计算的总费用：450x+10万美元
实际计算的总费用：360x+10万美元
学生计算的费用和实际计算的费用是否相同：否
学生的解决方案和实际解决方案是否相同：否
学生的成绩：不正确

三、局限性

开发大模型相关应用时请务必铭记：

虚假知识：模型偶尔会生成一些看似真实实则编造的知识

在开发与应用语言模型时，需要注意它们可能生成虚假信息的风险。尽管模型经过大规模预训练，掌握了丰富知识，但它实际上并没有*完全*记住所见的信息，难以准确判断自己的知识边界，可能做出错误推断。若让语言模型描述一个不存在的产品,它可能会自行构造出似是而非的细节。这被称为“幻觉”(Hallucination)，是语言模型的一大缺陷。

如下示例展示了大模型的幻觉。我们要求告诉我们华为公司生产的 *GT Watch 运动手表* 产品的信息，事实上，这个公司是真实存在的，但产品是编造的，而模型一本正经地提供了它编造的知识，而且迷惑性很强。

```
prompt = f"""
告诉我华为公司生产的GT Watch运动手表的相关信息
"""
response = get_completion(prompt)
print(response)
```

华为公司生产的GT Watch运动手表是一款智能手表，具有多种功能和特点。以下是相关信息：

1. 设计和外观：GT Watch采用圆形表盘设计，具有精致的外观和高质量的材料制造。它有多种颜色和表带选择，可以根据个人喜好进行定制。

2. 显示屏：GT Watch配备了1.39英寸的AMOLED显示屏，具有高清分辨率和良好的可视性。用户可以通过触摸屏幕进行操作和导航。

3. 运动追踪：GT Watch具有全天候的运动追踪功能，可以监测用户的步数、跑步距离、卡路里消耗和心率等数据。它还支持多种运动模式，如跑步、骑行、游泳等。

4. 健康监测：GT Watch可以监测用户的心率、血氧饱和度和睡眠质量等健康指标。它还提供健康建议和提醒，帮助用户保持良好的健康状态。

5. 通知和连接：GT Watch可以与用户的手机进行连接，通过蓝牙技术实现通知推送和电话提醒。用户可以在手表上查看短信、电话和社交媒体通知，无需拿出手机。

6. 长续航时间：GT Watch具有较长的续航时间，一次充电可以使用数天。它还支持快速充电技术，可以在短时间内充满电。

7. 其他功能：GT Watch还具有其他功能，如天气预报、闹钟、计时器、计步器等。它还支持NFC支付和音乐控制等便利功能。

总体而言，华为GT Watch是一款功能强大、外观精致的智能运动手表，适合那些注重健康和运动的用户使用。

语言模型生成虚假信息的"幻觉"问题，是使用与开发语言模型时需要高度关注的风险。由于幻觉信息往往令人无法辨别真伪，开发者必须警惕并尽量避免它的产生。

目前 OpenAI 等公司正在积极研究解决语言模型的幻觉问题。在技术得以进一步改进之前，开发者可以通过Prompt设计减少幻觉发生的可能。例如，可以先让语言模型直接引用文本中的原句，然后再进行解答。这可以追踪信息来源，降低虚假内容的风险。

综上，语言模型的幻觉问题事关应用的可靠性与安全性。开发者有必要认识到这一缺陷（注：截至2023年7月），并采取Prompt优化等措施予以缓解，以开发出更加可信赖的语言模型应用。这也将是未来语言模型进化的重要方向之一。

注意:

关于反斜杠使用的说明：在本教程中，我们使用反斜杠 \ 来使文本适应屏幕大小以提高阅读体验，而没有用换行符 \n 。GPT-3 并不受换行符（newline characters）的影响，但在您调用其他大模型时，需额外考虑换行符是否会影响模型性能。

四、英文原版 Prompt

1.1 使用分隔符清晰地表示输入的不同部分

```
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)
```

```
To guide a model towards the desired output and reduce irrelevant or incorrect
responses, it is important to provide clear and specific instructions, which can
be achieved through longer prompts that offer more clarity and context.
```

1.2 寻求结构化的输出
```

```
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)
```

```
{
  "books": [
    {
      "book_id": 1,
      "title": "The Enigma of Elysium",
      "author": "Evelyn Sinclair",
      "genre": "Mystery"
    },
    {
      "book_id": 2,
      "title": "Whispers in the Wind",
      "author": "Nathaniel Blackwood",
      "genre": "Fantasy"
    },
    {
      "book_id": 3,
      "title": "Echoes of the Past",
      "author": "Amelia Hart",
      "genre": "Romance"
    }
  ]
}
```

1.3 要求模型检查是否满足条件

```
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 - Once the water is hot enough, pour it over the tea bag.
Step 4 - Let it sit for a bit so the tea can steep.
Step 5 - After a few minutes, take out the tea bag.
Step 6 - If you like, add some sugar or milk to taste.
Step 7 - Enjoy your delicious cup of tea.
```

```
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.
```

1.4 提供少量示例（少样本提示词，Few-shot prompting）

```
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)
```

```
<grandparent>: Resilience is like a mighty oak tree that withstands the strongest storms, bending but never breaking. It is the unwavering determination to rise again after every fall, and the ability to find strength in the face of adversity. Just as a diamond is formed under immense pressure, resilience is forged through challenges and hardships, making us stronger and more resilient in the process.
```

2.1 指定完成任务所需的步骤

```
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, go on a quest to fetch water from a hilltop well,
but encounter misfortune when Jack trips on a stone and tumbles down the hill,
with Jill following suit, yet they return home and remain undeterred in their
adventurous spirits.

2 - Jack et Jill, frère et sœur, partent en quête d'eau d'un puits au sommet
d'une colline, mais rencontrent un malheur lorsque Jack trébuche sur une pierre
et dévale la colline, suivi par Jill, pourtant ils rentrent chez eux et restent
déterminés dans leur esprit d'aventure.

3 - Jack, Jill

4 - {
  "french_summary": "Jack et Jill, frère et sœur, partent en quête d'eau d'un
puits au sommet d'une colline, mais rencontrent un malheur lorsque Jack trébuche
sur une pierre et dévale la colline, suivi par Jill, pourtant ils rentrent chez
eux et restent déterminés dans leur esprit d'aventure.",
  "num_names": 2
}
```

```
prompt_2 = f"""
Your task is to perform the following actions:
1 - Summarize the following text delimited by <> 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.

Use the following format:
Text: <text to summarize>
Summary: <summary>
Translation: <summary translation>
Names: <list of names in French summary>
Output JSON: <json with summary and num_names>

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

```
Completion for prompt 2:
Summary: Jack and Jill, siblings from a charming village, go on a quest to fetch
water from a hilltop well, but encounter misfortune when Jack trips on a stone
and tumbles down the hill, with Jill following suit, yet they remain undeterred
and continue exploring with delight.

Translation: Jack et Jill, frère et sœur d'un charmant village, partent en quête
d'eau d'un puits au sommet d'une colline, mais rencontrent un malheur lorsque
Jack trébuche sur une pierre et dévale la colline, suivi par Jill, pourtant ils
restent déterminés et continuent à explorer avec joie.

Names: Jack, Jill

Output JSON:
{
  "french_summary": "Jack et Jill, frère et sœur d'un charmant village, partent
en quête d'eau d'un puits au sommet d'une colline, mais rencontrent un malheur
lorsque Jack trébuche sur une pierre et dévale la colline, suivi par Jill,
pourtant ils restent déterminés et continuent à explorer avec joie.",
  "num_names": 2
}
```

2.2 指导模型在下结论之前找出一个自己的解法

```
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. They correctly identified the costs for land,
solar panels, and maintenance, and calculated the total cost for the first year
of operations as a function of the number of square feet.
```

```
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.
- 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)
```

```
To calculate the total cost for the first year of operations, we need to add up
the costs of land, solar panels, and maintenance.

1. Land cost: $100 / square foot
The cost of land is $100 multiplied by the number of square feet.

2. Solar panel cost: $250 / square foot
The cost of solar panels is $250 multiplied by the number of square feet.

3. Maintenance cost: $100,000 + $10 / square foot
The maintenance cost is a flat fee of $100,000 per year, plus $10 multiplied by
the number of square feet.

Total cost: Land cost + Solar panel cost + Maintenance cost

So the actual solution is:
Total cost = (100 * x) + (250 * x) + (100,000 + (10 * x))

Is the student's solution the same as the actual solution just calculated:
No

Student grade:
Incorrect
```

3.1 幻觉

```
prompt = f"""
Tell me about AeroGlide UltraSlim Smart Toothbrush by Boie
"""
response = get_completion(prompt)
print(response)
```
```

The AeroGlide UltraSlim Smart Toothbrush by Boie is a technologically advanced toothbrush designed to provide a superior brushing experience. Boie is a company known for its innovative oral care products, and the AeroGlide UltraSlim Smart Toothbrush is no exception.

One of the standout features of this toothbrush is its ultra-slim design. The brush head is only 2mm thick, making it much thinner than traditional toothbrushes. This slim profile allows for better access to hard-to-reach areas of the mouth, ensuring a thorough and effective clean.

The AeroGlide UltraSlim Smart Toothbrush also incorporates smart technology. It connects to a mobile app via Bluetooth, allowing users to track their brushing habits and receive personalized recommendations for improving their oral hygiene routine. The app provides real-time feedback on brushing technique, duration, and coverage, helping users to achieve optimal oral health.

The toothbrush features soft, antimicrobial bristles made from a durable thermoplastic elastomer. These bristles are gentle on the gums and teeth, while also being effective at removing plaque and debris. The antimicrobial properties help to keep the brush head clean and hygienic between uses.

Another notable feature of the AeroGlide UltraSlim Smart Toothbrush is its long battery life. It can last up to 30 days on a single charge, making it convenient for travel or everyday use without the need for frequent recharging.

Overall, the AeroGlide UltraSlim Smart Toothbrush by Boie offers a combination of advanced technology, slim design, and effective cleaning capabilities. It is a great option for those looking to upgrade their oral care routine and achieve a healthier smile.