```
###############################################################################
#####
# Copyright 2025 Google LLC
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#     https://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
# See the License for the specific language governing permissions and
# limitations under the License.
###############################################################################
#####
```

- `Copyright 2025 Google LLC`
- Apache 라이선스 버전 2.0(이하 **라이선스**)에 따라
- 이 파일을 사용해야 합니다.
- 라이선스 사본은 다음에서 얻을 수 있습니다.

```
-  `https://www.apache.org/licenses/LICENSE-2.0`
```

- 법률이 요구하거나 서면으로 동의하지 않는 한, 소프트웨어는
- 어떠한 종류의 명시적 또는 묵시적 보증이나 조건 없이 "있는 그대로" 배포됩니다.
- 라이선스에 따른 특정 언어 권한 및 제한 사항을 참조하십시오.

Prompt Design - Best Practices

Share to:

Authors

Polong Lin

Karl Weinmeister

1. Overview

- 이 노트북은 프롬프트 엔지니어링의 필수 요소와 몇 가지 모범 사례를 다룹니다.
- 프롬프트 디자인에 대한 자세한 내용은 공식 가이드에서 확인하세요.
- 이 노트북에서는 프롬프트 엔지니어링의 모범 사례, 즉 응답의 품질을 개선하기 위해 프롬프트를 설계하는 방법을 배웁니다.
- 프롬프트 엔지니어링의 best practice:
  - `Be concise` = 간결하게 작성하기
  - `Be specific and well-defined` = 구체적이고 명확하게 정의하기
  - `Ask one task at a time` = 한 번에 하나의 작업 요청하기
  - `Turn generative tasks into classification tasks` = 생성 작업을 분류 작업으로 전환하기
  - `Improve response quality by including examples` = 예시를 포함하여 응답 품질 향상시키기

---

2. Getting Started

1) Install Google Gen AI SDK

```
# ! %pip install --upgrade --quiet google-genai

# ! pip install -U -q "google-genai"
```

a. 두 명령어의 차이점

- `pip install` 명령어 차이점: 주피터 vs. 터미널
  - `! %pip install --upgrade --quiet google-genai` 와 `pip install -U -q "google-genai"` 두 명령어는 `google-genai` 패키지를 설치/업그레이드하는 동일한 목적을 가집니다. 차이점은 주로 **실행 환경과 명령어 사용 방식**에 있습니다.

.....
.....

- Jupyter or Colab 환경용 (`! %pip install ...`)

```
! %pip install --upgrade --quiet google-genai
```

- `!`: 주피터 노트북이나 구글 코랩과 같은 **셀 기반 환경**에서 파이썬 코드 셀 내부에 쉘(운영체제) 명령어를 실행할 때 사용하는 접두사
- `%pip`
  - 역시 주피터/코랩에서 제공하는 **매직 명령어** → 현재 파이썬 커널에 `pip`를 호출하여 패키지 설치
  - `!pip`보다 더 안정적으로 현재 환경에 설치되도록 도움
- `--upgrade` **(`-U`)**: 이미 설치된 패키지가 있다면 **최신 버전으로 업그레이드**하라는 옵션
- `--quiet` **(`-q`)**: 설치 과정을 **조용히** 진행하여 불필요한 메시지를 최소화

.....
.....

- 일반 터미널/명령 프롬프트용 (`pip install ...`)

```
pip install -U -q "google-genai"
```

  - **`pip install`**:
    - 파이썬 패키지를 설치하는 **표준 명령어**
    - 터미널(리눅스/macOS)이나 명령 프롬프트(윈도우)에서 직접 실행할 때 사용
  - **`-U`**: `--upgrade`와 동일하게 이미 설치된 패키지를 **최신 버전으로 업그레이드**
  - **`-q`**: `--quiet`와 동일하게 설치 과정을 **조용히** 진행
  - **`"google-genai"`**:
    - 설치하려는 패키지 이름
    - 큰따옴표는 필수는 아니지만 명확성을 위해 사용될 수 있음 → 구글 공식 가이드 새 SDK 라이브러리 설치 방법 참고

b. 실습 환경에 따른 설치 안내

- **구글 클라우드 실습 환경**
  - `%pip install --upgrade --quiet google-genai`으로 설치
- **로컬**
  - `gemini API key` 발급 → `.env`에 환경변수로 저장
    - 참고 예시 : `.env.example`
  - 터미널 or VSCode 터미널 등 → `pip install -U -q "google-genai"`으로 설치
    - 현 프로젝트에서는 `requirements.txt`에 포함시켜둠

2) Import libraries

a. Google Colab 라이브러리 및 로컬 환경 설정

- `Colab`에서 사용하는 라이브러리
  - **`IPython.display` (Markdown, display)**:

        - **역할**: 주피터/코랩 환경에서 마크다운 텍스트 렌더링 및 다양한 객체(텍스트, 이미지 등)를 시각적으로 표시하는 데 사용
    - **`google.generativeai` (이전 `google.genai`)**:
        - **역할**
            - Google 의 생성형 AI 모델(Gemini)과 통신하기 위한 공식 Python SDK
            - 모델 호출 및 설정에 사용
    - **`google.generativeai.types.GenerateContentConfig`** (이전 `google.genai.types.GenerateContentConfig`):
        - **역할**: 생성형 AI 모델 호출 시 `temperature`, `top_k`, `safety_settings` 등 추가적인 응답 설정 옵션을 지정하는 데 사용

b. 로컬 환경에서 필요한 설치 및 고려사항

- **필수 설치**:
    - **`google-generativeai` SDK**:
        - **`pip install -U google-generativeai`** 명령어로 설치
        - `Gemini` 모델과 통신하는 데 핵심적인 역할
- **선택적 설치**:
    - **`jupyter` 또는 `IPython`**:
        - `pip install jupyter` 로 설치하며, Colab 과 유사한 대화형 노트북 환경을 로컬에서 사용하고 싶을 때 필요
        - 일반 `.py` 스크립트 실행 시에는 불필요
- **가장 중요한 설정 (인증)**:
    - **API 키 확보**: Google AI Studio 에서 `Gemini API 키` 생성
    - **API 키 설정**:
        - 생성된 API 키를 `.env` 파일 환경 변수로 설정
        - `SDK` 가 이 환경 변수를 자동으로 감지하여 사용

```
! pip install -U -q "google-genai"

"""
셀 출력
- [notice] A new release of pip is available: 25.1.1 -> 25.2
- [notice] To update, run: pip install --upgrade pip
"""

! pip install --upgrade pip

"""
셀 출력:
    Requirement already satisfied: pip in /Users/경로... (25.1.1)
    Collecting pip
    Using cached pip-25.2-py3-none-any.whl.metadata (4.7 kB)
    Using cached pip-25.2-py3-none-any.whl (1.8 MB)
    Installing collected packages: pip
    Attempting uninstall: pip
        Found existing installation: pip 25.1.1
        Uninstalling pip-25.1.1:
```

```
        Successfully uninstalled pip-25.1.1
    Successfully installed pip-25.2
"""

"""
# 구글 클라우드 실습 환경or Colab 실습 환경 실행 시
from IPython.display import Markdown, display
from google import genai
from google.genai.types import GenerateContentConfig
"""

# 로컬에서 실행 시
from IPython.display import Markdown, display
from google import genai
from google.genai import types
```

3) Set Google Cloud project information and create client

a. 구글 클라우드 프로젝트 환경 설정

- Vertex AI 사용을 시작하려면 기존 Google Cloud 프로젝트가 있어야 하며, Vertex AI API 를 사용 설정해야 합니다.
- 프로젝트 및 개발 환경 설정에 대해 자세히 알아보세요.

b. 클라이언트 설정

- **구글 클라우드 실습 환경**
  - `PROJECT_ID` = `Google Cloud` 프로젝트를 식별하는 고유한 `ID`
    - `qwiklabs-**********` = Qwiklabs 실습 환경에서 임시로 부여된 `ID`
    - 실제 사용 시에는 본인의 GCP 프로젝트 ID 로 대체해야 함
    - `LOCATION` = `Google Cloud region`(예: us-central1, asia-east1) 지정
      - `Vertex AI` 및 `Gemini` 모델은 특정 리전에 배포 → 해당 리전을 명시해야 함
    - `client = genai.Client(...)`
      - `google-genai` 라이브러리가 `Vertex AI` 엔드포인트를 통해 `Google Generative AI` 모델에 접근하도록 **클라이언트 초기화**
      - 주로 `Google Cloud` 환경 내에서 `Vertex AI` 서비스를 활용할 때 사용

```
import os
```

```
# Google Cloud 프로젝트 ID 설정
PROJECT_ID = "qwiklabs-**********"
#   가상의 값로 대신함
             # 파라미터 = str


# 프로젝트 ID 가 설정되지 않았거나 기본값인 경우, 환경 변수에서 가져옴
(Colab/GCP 환경에서 주로 사용)
if not PROJECT_ID or PROJECT_ID == "qwiklabs-**********":
    PROJECT_ID = str(os.environ.get("GOOGLE_CLOUD_PROJECT",
""))

LOCATION = "europe-****"
# 모델이 배포된 Google Cloud 리전 설정

# 확인용으로 추가
print(f"Project ID: {PROJECT_ID}")
# Project ID: qwiklabs-**********
print(f"Location: {LOCATION}")
# Location: europe-****
```

- **로컬 실습 환경**
  - 새 SDK 라이브러리 코드 참조
  - Client 객체를 통해 모든 API 메서드에 대한 액세스를 제공
    - Client 객체 자체가 만능 키 역할
      - 모델 호출 (`client.models.generate_content()`), 채팅 (`client.chats.create()`), 파일 업로드 (`client.files.upload()`), 모델 튜닝 (`client.tunings.tune()`) 등을 모두 이 **`client` 객체 하나로 가능**
    - 대부분 **상태 비저장(`Stateless`)** 기능
      - **스테이트리스(`Stateless`) 함수** = 각 함수 호출이 독립적으로 작동한다는 의미 → 하나의 함수를 호출할 때 이전 호출의 정보나 상태를 기억하거나 영향을 받지 않음
      - 예를 들어, `client.models.generate_content("질문 1")`을 호출하고, 다음에 `client.models.generate_content("질문 2")`를 호출하면, 두 번째 호출은 첫 번째 호출의 내용을 전혀 모른 채 **독립적으로** '질문 2'에 대한 응답을 생성
      - 대부분의 API 기능은 이렇게 독립적인 요청-응답 형태로 설계 → 사용하기 간편하고 예측 가능
    - **상태 저장(`Stateful`)** 특수 사례 (`Chat`, `Live API Session`)

- **스테이트풀(Stateful) 특수 사례** = 함수 호출 간에 상태나 맥락을 유지해야 하는 특별한 경우
- **독립적인 함수 호출 X** : **세션** or **객체** 먼저 생성 → 그 객체를 통해 **연속적인 상호작용**을 하는 방식으로 작동
- 대표
  - **`chat` (채팅)** → 채팅은 이전 대화 내용을 기억하고 그 맥락 위에서 다음 응답을 생성해야 하므로, **상태**를 가지고 있어야 함
    - 채팅 **세션 객체** 생성: `client.chats.create()`

```
import google.generativeai as genai
from dotenv import load_dotenv
import os

load_dotenv() # .env 파일 로드
client = genai.Client() # Client 객체 생성

# 채팅 모델 초기화 (예: 'gemini-pro')
chat_model =
client.models.GenerativeModel('gemini-pro')

# 채팅 세션 생성
# chat = client.chats.create() # 이렇게 직접
chat 객체를 생성하지 않고,
# 보통 모델 객체에서 start_chat() 메서드를 사용합니
다.

chat_session =
chat_model.start_chat(history=[]) # 빈 대화 기
록으로 시작
print("채팅 세션이 시작되었습니다.")
```

- 채팅 세션 객체 통한 대화 진행: `chat_session = chat_model.start_chat()` = **연속적 상호작용 가능**

```
response1 = chat_session.send_message("안
녕하세요? 자기소개 부탁해요.")
print("첫 번째 응답:", response1.text)

response2 = chat_session.send_message("그
럼, 인공지능에 대해 더 설명해줄 수 있나요?")
```

```
  print("두 번째 응답:", response2.text)

  # 대화 기록 확인
  for message in chat_session.history:
      print(f"{message.role}:
{message.parts[0].text}")
```

- **라이브 API 세션(session)** → **실시간**으로 **연속적인 데이터 교환**이나 **상태 유지가 필요한 경우** 의미

- 반환되는 객체 = `pydantic` 클래스:
  - `API` 호출 결과로 반환되는 데이터들은 `pydantic` 이라는 라이브러리의 클래스 형태로 제공
  - `pydantic` 은 **데이터 유효성 검사** 및 **설정 관리**에 **유용**한 라이브러리입니다.
  - `API` 응답 데이터가 **일관된 구조**와 **타입**을 가지도록 보장하며, 개발자가 **데이터**를 다루기 더 **쉽고 안전하게** 함

```
! pip install python-dotenv

from dotenv import load_dotenv
import os

# .env 파일 로드
load_dotenv()                                                    # true

# 클라이언트 초기화
# 단일 클라이언트 객체 생성하기
# API 키는GEMINI_API_KEY 환경 변수에서 자동으로 로드

client = genai.Client()

"""
- 제미나이SDK 클라이언트 초기화 생성 방법 예시들
# chat = client.chats.create(...)                    # 채팅 세션 생성
# my_file = client.files.upload(...)                 # 파일 업로드 (나중에 튜닝 등
에 사용)
# tuning_job = client.tunings.tune(...)              # 모델 튜닝 작업 생성
"""
```

4) Load model

여기에서 Vertex AI Gemini models 들을 더 알아보세요.

a. 구글 클라우드 실습 환경

- 정해진 모델을 골라 ID 에 설정 → **모델 호출**

```
MODEL_ID = "gemini-2.5-flash"  # @param {type: "string"}
# 다른 모델도 호출 가능
```

b. 로컬

- 사용할 모델 지정 = 호출 (같은 작업)
  - `gemini-pro` = 텍스트 기반 일반 작업에 적합
  - `-flash` 기반 모델 = 더 빠르고 비용 효율적인 최신 모델

```
# --- client 객체를 사용하여 "헬로우" 호출해보기---

# 1. 사용할 모델 지정 (gemini-2.5-flash-lite)


# 2. '헬로우' 프롬프트로 콘텐츠 생성 요청
try:
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',                    # 사용할 모델 지정 (gemini-2.5-flash-lite)
        contents='hello'                                  # 요청하신 프롬프트 '헬로우'
    )

    # 응답 텍스트 출력
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)

    # 응답 전체 내용 (JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    print("\n--- 모델 응답 전체 JSON ---")
    print(response.model_dump_json(
        exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력

```
--- 모델 응답 텍스트 ---
Hello! How can I help you today?
```

```
  --- 모델 응답 전체 JSON ---
      {
          "sdk_http_response": {
              "headers": {
                  "content-type": "application/json; charset=UTF-8",
                  "vary": "Origin, X-Origin, Referer",
                  "content-encoding": "gzip",
                  "date": "Mon, 04 Aug 2025 08:46:18 GMT",
                  "server": "scaffolding on HTTPServer2",
                  "x-xss-protection": "0",
                  "x-frame-options": "SAMEORIGIN",
                  "x-content-type-options": "nosniff",
                  "server-timing": "gfet4t7; dur=539",
                  "alt-svc": "h3=\":443\"; ma=2592000,h3-29=\":443\";
ma=2592000",
                  "transfer-encoding": "chunked"
              }
          },
          "candidates": [
              {
                  "content": {
                      "parts": [
                          {
                              "text": "Hello! How can I help you
today?"
                          }
                      ],
                      "role": "model"
                  },
                  "finish_reason": "STOP",
                  "index": 0
              }
          ],
          "model_version": "gemini-2.5-flash-lite",
          "usage_metadata": {
              "candidates_token_count": 9,
              "prompt_token_count": 2,
              "prompt_tokens_details": [
                  {
                      "modality": "TEXT",
                      "token_count": 2
                  }
              ],
              "total_token_count": 11
          },
          "automatic_function_calling_history": []
      }
```

3. Prompt engineering best practices

- 프롬프트 엔지니어링은 응답이 당신이 진정으로 보고 싶었던 것이 되도록 프롬프트를 설계하는 방법에 관한 모든 것입니다.
- **화려하지 않은(`unfancy`)** 프롬프트를 사용한다는 생각은 `LLM`(대규모 언어 모델)이 프롬프트의 의도를 잘못 해석할 가능성을 줄이기 위해 **프롬프트 내의 불필요한 요소를 최소화**하는 것입니다. 아래는 **화려하지 않은** 프롬프트를 설계하는 방법에 대한 몇 가지 지침입니다.
- 이 섹션에서는 프롬프트 엔지니어링 시 다음 **`best practices`** 을 다룰 것입니다:
  - `Be concise` - **간결하게 작성하기**
  - `Be specific, and well-defined` - **구체적이고 명확하게 정의하기**
  - `Ask one task at a time` - **한 번에 하나의 작업만 요청하기**
  - `Improve response quality by including examples` - **예시를 포함하여 응답 품질 향상시키기**
  - `Turn generative tasks to classification tasks to improve safety` - **생성 작업을 분류 작업으로 전환하여 안전성 향상시키기**

1) Be concise

 비추천 아래 프롬프트는 불필요하게 장황합니다.

a. 구글 클라우드 실습 환경

- 위 안내문은 구글 클라우드 실습 과정에서의 안내문으로 그 이유는 다음과 같음
  - 이 프롬프트는 불필요하게 많은 단어를 사용하거나 복잡하게 작성되어 있기 때문
  - 이로 인해 LLM(대규모 언어 모델)이 핵심 의도를 파악하기 어렵거나, 엉뚱한 부분에 집중하여 원하지 않는 답변을 생성할 가능성이 높아짐
- 실습 코드

```
    prompt = "What do you think could be a good name for a flower
shop that specializes in selling bouquets of dried flowers more
than fresh flowers?"

    response = client.models.generate_content(model=MODEL_ID,
contents=prompt)
    display(Markdown(response.text))
```

- 답변

```
    This is a fun challenge! Dried flowers evoke a sense of
timelessness, nostalgia, rustic charm, and enduring beauty. Here
are some name ideas, playing on different vibes:

    **Evocative & Poetic:**
```

1. **The Everlasting Bouquet:** Simple, classic, and direct.
2. **Timeless Botanicals:** Suggests enduring beauty and a curated collection.
3. **Whispering Blooms:** Implies a soft, delicate, and lasting presence.
4. **Aura Flora:** Combines a sense of atmosphere with flowers.
5. **Echo & Bloom:** Suggests lasting memories and beauty.
6. **Solis & Bloom:** "Solis" means sun, hinting at sun-dried flowers.
7. **The Petal Archive:** Implies a collection of preserved beauty, like a treasured library.
8. **Bloom & Memory:** Connects the flowers to sentiment and lasting keepsakes.

Charming & Whimsical:

9. **Dusted Blooms:** A playful, slightly vintage feel, but in a good way.
10. **Willow & Bloom:** Willow can evoke natural forms and grace.
11. **The Quiet Bloom:** Understated, elegant, and serene.
12. **The Gathered Stem:** Suggests a handcrafted, natural aesthetic.
13. **Fable & Bloom:** Adds a touch of storytelling and magic.
14. **The Floral Keepsake:** Emphasizes the lasting, sentimental value.

Modern & Chic:

15. **Bloom & Dust:** Minimalist and sophisticated, playing on the dried aspect without being negative.
16. **Kinfolk Botanicals:** Implies a natural, curated, and community-focused brand.
17. **Curated Dried:** Direct and highlights a refined selection.
18. **The Bloom Loft:** Suggests an airy, stylish studio space.
19. **Terra Bloom:** Combines earthiness with floral beauty.
20. **Root & Bloom (Dried):** Grounds the concept while specifying the product.

Direct & Descriptive (but still elegant):

21. **The Dried Bloom:** Clear, concise, and elegant.
22. **Everlasting Petals:** Highlights longevity and beauty.
23. **Preserved Posies:** Charming alliteration, "posies" adds a quaint touch.

```
    24. **Dried & True:** A clever play on "tried and true."
    25. **Forever Florals:** Emphasizes the lasting nature.

    **Tips for Choosing:**

    *   **Say it out loud:** Does it roll off the tongue? Is it
easy to remember?
    *   **Check availability:** Is the name available as a domain
name, social media handle, and business registration?
    *   **Consider your target audience:** Are they looking for
rustic charm, modern elegance, or something whimsical?
    *   **Think about your branding:** Does the name align with
the visual aesthetic you envision for your shop?
    *   **Future growth:** Does the name allow for slight
expansion if you decide to introduce other lasting botanical
products (e.g., pressed flowers, botanical art)?

    Good luck!
```

 추천하는 문장입니다. 아래 프롬프트는 간결하고 핵심을 잘 담고 있습니다.

-
  - 추천하는 문장의 코드

```
  prompt = "Suggest a name for a flower shop that sells bouquets
of dried flowers"

  response = client.models.generate_content(model=MODEL_ID,
contents=prompt)
  display(Markdown(response.text))
```

  - 대답

```
  Here are some name suggestions for a flower shop
specializing in dried flower bouquets, categorized by the
vibe they evoke:

  **Elegant & Timeless:**
  1.  **The Everlasting Bloom:** Evokes longevity and
beauty.
  2.  **Unfading Petals:** Suggests lasting beauty and
delicate charm.
  3.  **Timeless Bloom Co.:** Modern, yet speaks to
enduring nature.
  4.  **The Botanical Archive:** Implies a curated
collection of preserved beauty.
  5.  **Heirloom Blooms:** Suggests cherished, passed-down
beauty.
```

```
  6.  **Petal & Preservation:** Clear, sophisticated, and
descriptive.
  7.  **Moonpetal Collective:** Ethereal, gentle, and a bit
mystical.
  8.  **Apothecary Blooms:** Connects to natural remedies,
dried herbs, and artisanal craft.

  **Whimsical & Charming:**
  9.  **Whispering Blooms:** Delicate, gentle, and
enchanting.
  10. **Thistle & Twine:** Rustic, natural, and a little
wild.
  11. **Fable & Fern:** Conjures images of enchanted
forests and stories.
  12. **The Quiet Bloom:** Peaceful, serene, and
understated beauty.
  13. **Dusty Rose Dried:** References a classic dried
flower color and feel.
  14. **Sunstone Blooms:** Warm, lasting, and unique like a
gem.

  **Rustic & Earthy:**
  15. **Field & Forage Dried:** Suggests natural sourcing
and a rustic aesthetic.
  16. **Root & Petal:** Highlights the foundational
elements of plants.
  17. **Earthbound Blooms:** Connects to the earth and
lasting presence.
  18. **Harvest & Haze:** Evokes autumn, natural bounty,
and a soft, muted feel.
  19. **Wild & Dried:** Simple, direct, and speaks to
natural, untamed beauty.

  **Modern & Minimalist:**
  20. **The Dried Bloom:** Simple, direct, and
sophisticated.
  21. **Ever Bloom Co.:** Short, memorable, and focuses on
longevity.
  22. **Preserve & Petal:** Clean, active, and highlights
the unique product.
  23. **Bloom & Thyme:** (Thyme as in 'time' and the herb)
Clever and elegant.
  24. **Botanica Dried:** Clear, professional, and
contemporary.

  **Tips for Choosing:**
  *   **Say it out loud:** Does it roll off the tongue?
  *   **Check availability:** Is the name (and domain name)
available?
  *   **Consider your target audience:** Who are you trying
```

```
to attract?
  *   **Reflect your brand's aesthetic:** Does the name
match the look and feel of your shop?

  Good luck!
```

---

b. 로컬

- 간결하게 작성해보기

- 좋은 간단한 입력 프롬프트의 예시와 예상되는 출력값

입력 (프롬프트)	출력 (예상 응답)
대한민국의 수도는?	서울
"인공지능" 정의	인공지능(AI)은 인간의 학습 능력, 문제 해결 능력, 인지 능력 등을 모방하여 컴퓨터 프로그램으로 구현한 기술입니다.
파이썬 장점 3 가지	1. 배우기 쉽고 문법이 간결합니다.2. 다양한 라이브러리와 프레임워크를 지원합니다.3. 여러 운영체제에서 호환됩니다.
이 문장을 존댓말로 바꿔줘: "반말 쓰지 마."	존댓말을 사용해 주세요.
다음 중 채소는?: 사과, 감자, 바나나	감자

```
# 1_1. 구글 클라우드 실습 질문과 같은 질문으로 입력해보기

prompt = "Suggest a name for a flower shop that sells bouquets of
dried flowers"
prompt_ko = "드라이 플라워 부케를 판매하는 꽃집에 어울리는 이름을 추천해 주세요."

try:
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',                    # 사용할 모델
지정
        #contents=prompt                                   # 같은 프롬프
트 입력
        contents=prompt_ko                                 # 한글로 입력
해보기
    )

    # 응답 텍스트 출력
    print("\n--- 모델 응답 텍스트 ---")
```

```
    print(response.text)

    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    #print("\n--- 모델 응답 전체 JSON ---")
    #print(response.model_dump_json(
    #    exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력
  - 영어로 질문

```
    --- 모델 응답 텍스트 ---
    Here are some name suggestions for a flower shop specializing
in dried flower bouquets, categorized for different vibes:

    **Evocative & Romantic:**

    *   **Whispering Blooms:** Suggests softness and lasting
memories.
    *   **Eternal Petals:** Highlights the longevity of dried
flowers.
    *   **Faded Elegance:** Captures the beauty in the aging
process.
    *   **Moonlit Meadow:** Creates a mystical and dreamy
atmosphere.
    *   **Harvested Whispers:** Implies gathered beauty from
nature.
    *   **Autumnal Alchemy:** Suggests a magical transformation.
    *   **Gilded Gardens:** Hints at preserved, precious beauty.
    *   **Timeless Treasures:** Emphasizes the enduring quality.
    *   **Fireside Flora:** Conveys warmth and coziness.
    *   **Golden Hour Bouquets:** Captures the warm, rich colors
of dried flowers.

    **Modern & Chic:**

    *   **Dried & Styled:** Simple, direct, and emphasizes
curation.
    *   **Preserve & Co.:** Modern, professional, and suggests a
collective of artistry.
```

```
    *   **The Dried Edit:** Implies a carefully selected
collection.
    *   **Petal Preservation:** Clean, descriptive, and elegant.
    *   **Botanical Archive:** Suggests a curated collection of
preserved beauty.
    *   **The Dry Bloom Bar:** Trendy and inviting, like a
specialty beverage bar.
    *   **Terra Flora:** Earthy and sophisticated.
    *   **Aura Botanics:** Focuses on the atmosphere and
aesthetic.
    *   **Sculpted Stems:** Highlights the artistic arrangement.
    *   **The Modern Wreath:** If wreaths are a focus.

    **Natural & Earthy:**

    *   **Earth's Embrace:** Suggests natural beauty and comfort.
    *   **The Sun-Kissed Stem:** Evokes the drying process.
    *   **Wildwood Wreaths & Blooms:** Classic, natural, and
descriptive.
    *   **Rooted & Dried:** Emphasizes the origin and
preservation.
    *   **The Harvested Home:** Suggests bringing nature indoors.
    *   **Bramble & Bloom:** Rustic and charming.
    *   **Sage & Thistle:** Evokes natural textures and colors.
    *   **The Wildcrafted Collection:** Highlights natural
sourcing.
    *   **Stone & Stem:** Simple, grounded, and natural.
    *   **Prairie Petals:** Evokes open fields and natural
beauty.

    **Creative & Playful:**

    *   **Dusty Petals, Happy Hearts:** Charming and a bit
whimsical.
    *   **The Dried Flower Fairy:** Magical and approachable.
    *   **Unfading Friends:** A cute way to describe dried
flowers.
    *   **Perennial Pleasures:** A clever play on words.
    *   **Bloom & Decay Boutique:** Bold and artistic.
    *   **The Wilted Wonder:** Embraces the unique beauty of
dried flowers.
    *   **Charmed Charms:** Suggests the decorative appeal.
    *   **The Dried Dandelion:** Simple, memorable, and natural.
    *   **Petal Ponderings:** Evokes thought and appreciation.
    *   **The Dried Bouquet Bar:** Simple and to the point.

    **Tips for Choosing:**

    *   **Say it Out Loud:** Does it roll off the tongue? Is it
easy to remember?
```

```
    *   **Check for Availability:** Is the domain name and social
media handle available?
    *   **Consider Your Target Audience:** Who are you trying to
attract?
    *   **Reflect Your Brand:** What is the overall feeling you
want your shop to evoke?
    *   **Get Feedback:** Ask friends, family, or potential
customers what they think.

    Good luck with your new dried flower shop!

    --- 모델 응답 전체 JSON ---
    {
        "sdk_http_response": {
            "headers": {
                "content-type": "application/json; charset=UTF-
8",
                "vary": "Origin, X-Origin, Referer",
                "content-encoding": "gzip",
                "date": "Mon, 04 Aug 2025 09:21:32 GMT",
                "server": "scaffolding on HTTPServer2",
                "x-xss-protection": "0",
                "x-frame-options": "SAMEORIGIN",
                "x-content-type-options": "nosniff",
                "server-timing": "gfet4t7; dur=3788",
                "alt-svc": "h3=\":443\"; ma=2592000,h3-
29=\":443\"; ma=2592000",
                "transfer-encoding": "chunked"
            }
        },
        "candidates": [
            {
                "content": {
                    "parts": [
                        {
                            "text": "Here are some name
suggestions for a flower shop specializing in dried flower
bouquets, categorized for different vibes:\n\n**Evocative &
Romantic:**\n\n*   **Whispering Blooms:** Suggests softness and
lasting memories.\n*   **Eternal Petals:** Highlights the
longevity of dried flowers.\n*   **Faded Elegance:** Captures the
beauty in the aging process.\n*   **Moonlit Meadow:** Creates a
mystical and dreamy atmosphere.\n*   **Harvested Whispers:**
Implies gathered beauty from nature.\n*   **Autumnal Alchemy:**
Suggests a magical transformation.\n*   **Gilded Gardens:** Hints
at preserved, precious beauty.\n*   **Timeless Treasures:**
Emphasizes the enduring quality.\n*   **Fireside Flora:** Conveys
warmth and coziness.\n*   **Golden Hour Bouquets:** Captures the
warm, rich colors of dried flowers.\n\n**Modern & Chic:**\n\n*
```

```
**Dried & Styled:** Simple, direct, and emphasizes curation.\n*
**Preserve & Co.:** Modern, professional, and suggests a
collective of artistry.\n*   **The Dried Edit:** Implies a
carefully selected collection.\n*   **Petal Preservation:**
Clean, descriptive, and elegant.\n*   **Botanical Archive:**
Suggests a curated collection of preserved beauty.\n*   **The Dry
Bloom Bar:** Trendy and inviting, like a specialty beverage bar.\
n*   **Terra Flora:** Earthy and sophisticated.\n*   **Aura
Botanics:** Focuses on the atmosphere and aesthetic.\n*
**Sculpted Stems:** Highlights the artistic arrangement.\n*
**The Modern Wreath:** If wreaths are a focus.\n\n**Natural &
Earthy:**\n\n*   **Earth's Embrace:** Suggests natural beauty and
comfort.\n*   **The Sun-Kissed Stem:** Evokes the drying
process.\n*   **Wildwood Wreaths & Blooms:** Classic, natural,
and descriptive.\n*   **Rooted & Dried:** Emphasizes the origin
and preservation.\n*   **The Harvested Home:** Suggests bringing
nature indoors.\n*   **Bramble & Bloom:** Rustic and charming.\n*
**Sage & Thistle:** Evokes natural textures and colors.\n*
**The Wildcrafted Collection:** Highlights natural sourcing.\n*
**Stone & Stem:** Simple, grounded, and natural.\n*   **Prairie
Petals:** Evokes open fields and natural beauty.\n\n**Creative &
Playful:**\n\n*   **Dusty Petals, Happy Hearts:** Charming and a
bit whimsical.\n*   **The Dried Flower Fairy:** Magical and
approachable.\n*   **Unfading Friends:** A cute way to describe
dried flowers.\n*   **Perennial Pleasures:** A clever play on
words.\n*   **Bloom & Decay Boutique:** Bold and artistic.\n*
**The Wilted Wonder:** Embraces the unique beauty of dried
flowers.\n*   **Charmed Charms:** Suggests the decorative
appeal.\n*   **The Dried Dandelion:** Simple, memorable, and
natural.\n*   **Petal Ponderings:** Evokes thought and
appreciation.\n*   **The Dried Bouquet Bar:** Simple and to the
point.\n\n**Tips for Choosing:**\n\n*   **Say it Out Loud:** Does
it roll off the tongue? Is it easy to remember?\n*   **Check for
Availability:** Is the domain name and social media handle
available?\n*   **Consider Your Target Audience:** Who are you
trying to attract?\n*   **Reflect Your Brand:** What is the
overall feeling you want your shop to evoke?\n*   **Get
Feedback:** Ask friends, family, or potential customers what they
think.\n\nGood luck with your new dried flower shop!"
                        }
                    ],
                    "role": "model"
                },
                "finish_reason": "STOP",
                "index": 0
            }
        ],
        "model_version": "gemini-2.5-flash-lite",
        "usage_metadata": {
```

```
            "candidates_token_count": 796,
            "prompt_token_count": 14,
            "prompt_tokens_details": [
                {
                    "modality": "TEXT",
                    "token_count": 14
                }
            ],
            "total_token_count": 810
        },
        "automatic_function_calling_history": []
    }
```

```python
# 1_2. 한글로 입력해보기

#prompt = "Suggest a name for a flower shop that sells bouquets of dried flowers"
prompt_ko = "드라이 플라워 부케를 판매하는 꽃집에 어울리는 이름을 추천해 주세요."

try:
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',                      # 사용할 모델 지정
        #contents=prompt                                     # 같은 프롬프트 입력
        contents=prompt_ko                                  # 한글로 입력해보기
    )

    # 응답 텍스트 출력
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)

    # ------------ 디버깅용은 출력하지 않기----------------------
    # 응답 전체 내용(JSON 형식) 출력 ( 디버깅이나 상세 정보 확인용)
    #print("\n--- 모델 응답 전체JSON ---")
    #print(response.model_dump_json(
    #    exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력

```
--- 모델 응답 텍스트 ---
드라이 플라워 부케를 판매하는 꽃집에 어울리는 이름은 고객에게 어떤 분위기와 느낌을 전달하고 싶은지에 따라 달라질 수 있습니다. 몇 가지 카테고리로 나누어 다양한 스타일의 이름을 추천해 드릴게요.

---

### 1. 감성적이고 시적인 이름

오랫동안 변치 않는 아름다움, 추억, 그리고 잔잔한 감성을 담고 싶을 때 어울립니다.

*   **시간의 정원:** 시간이 흘러도 변치 않는 아름다움을 표현
*   **영원한 조각:** 드라이 플라워의 영속성과 예술성을 담은 이름
*   **꽃숨:** 꽃이 숨 쉬는 듯한 은유적인 표현
*   **말 없는 언어:** 꽃이 전하는 이야기와 감성을 의미
*   **추억의 향기:** 드라이 플라워가 불러일으키는 추억을 강조
*   **햇살 조각:** 따뜻하고 부드러운 드라이 플라워의 이미지를 표현
*   **바람의 노래:** 자연스럽고 싱그러운 느낌을 주는 이름
*   **별을 담은 꽃:** 몽환적이고 로맨틱한 느낌
*   **마음의 정원:** 소중한 사람에게 마음을 전하는 공간
*   **기억의 조각들:** 아름다운 순간들을 담은 드라이 플라워

---

### 2. 모던하고 세련된 이름

심플함, 트렌디함, 그리고 현대적인 감각을 강조하고 싶을 때 좋습니다.

*   **DRYISM (드라이이즘):** 'Dry'와 '-ism'을 결합하여 드라이 플라워의 가치를 나타냄
*   **The Dried Flower Lab:** 실험적이고 창의적인 디자인을 연상
*   **Atelier DRY (아뜰리에 드라이):** 프랑스어로 '작업실'을 뜻하며 전문성과 예술성을 더함
*   **Flow.R (플로알):** 'Flow'와 'Flower'를 결합하여 부드러운 느낌, 끝에 'R'을 넣어 고급스러움
*   **Bloom & Dry:** 꽃의 피어남과 드라이된 상태를 동시에 표현
*   **Verve Dry (버브 드라이):** 'Verve'는 활력, 열정을 뜻하며 생명력을 잃지 않는 드라이 플라워를 의미
*   **Nuance Dry (뉴앙스 드라이):** 색감과 질감의 미묘한 차이를 담은 드라이 플라워
*   **Lumi Dry (루미 드라이):** 'Lumi'는 빛을 의미하며, 빛나는 듯한 드라이 플라워
*   **The Preservation (더 프레저베이션):** 보존이라는 의미로 드라이 플라워의 특성을 직관적으로 전달
*   **Aura DRY (아우라 드라이):** 드라이 플라워가 가진 고유의 분위기와 매력

---
```

```
### 3. 친근하고 따뜻한 이름

이름만 들어도 포근하고 정겨운 느낌을 주어 단골 고객을 만들기 좋습니다.

*   **꽃갈피:** 꽃처럼 아름다운 기억을 간직하는 곳
*   **소소한 꽃집:** 일상 속 작은 행복을 주는 드라이 플라워
*   **향기로운 순간:** 드라이 플라워가 주는 편안한 향기와 시간
*   **꽃이랑:** '꽃'과 '이랑'(친근한 표현)을 결합
*   **우리집 꽃:** 마치 내 집처럼 편안한 느낌
*   **오늘의 꽃:** 매일 새로운 드라이 플라워를 만나는 설렘
*   **마음을 드려요:** 선물하는 기쁨을 강조
*   **꽃 단지:** 자연스럽고 정겨운 느낌
*   **봄날의 드라이:** 봄처럼 따뜻하고 생기 넘치는 드라이 플라워
*   **햇살꽃방:** 햇살처럼 따뜻하고 밝은 느낌

---

### 4. 개성 있고 유니크한 이름

기억에 남는 독특한 이름으로 브랜드 아이덴티티를 구축하고 싶을 때 좋습니다.

*   **꽃의 안식처:** 시들지 않는 아름다움이 영원히 머무는 곳
*   **시간의 숲:** 시간이 멈춘 듯한 신비로운 분위기
*   **비밀의 화원:** 숨겨진 아름다움을 발견하는 공간
*   **달빛 정원:** 은은하고 신비로운 느낌
*   **결의 꽃:** 꽃잎의 질감과 아름다움을 강조
*   **잊혀지지 않을:** 강렬한 인상을 주는 이름
*   **그 계절의 꽃:** 특정 계절의 감성을 담은 드라이 플라워
*   **꽃의 속삭임:** 조용히 감동을 주는 드라이 플라워
*   **꽃 언덕:** 풍성하고 아름다운 드라이 플라워의 이미지
*   **기억의 가지:** 추억을 매달아 놓은 듯한 감성

---

**이름을 선택하실 때 고려할 점:**

1. **타겟 고객:** 어떤 연령층과 스타일에 관심 있는 고객에게 어필하고 싶은가요?
2. **꽃집의 컨셉:** 모던, 빈티지, 내추럴, 로맨틱 등 어떤 분위기의 꽃집을 만들고 싶으신가요?
3. **기억하기 쉬운가:** 발음하기 쉽고 기억에 잘 남는 이름이 좋습니다.
4. **독창적인가:** 다른 꽃집과 차별화되는 독특한 이름이 좋습니다.
5. **의미 전달:** 드라이 플라워의 어떤 매력을 강조하고 싶은가요?

이 중에서 마음에 드는 이름을 고르시거나, 이 아이디어들을 바탕으로 더욱 멋진 이름을 만들어나가시길 바랍니다!
```

```python
# 2_ 프롬프트 작성 - 간결하게 입력하기
# 입력값 리스트
prompt_list = [
```

```
    "대한민국의 수도는?",
    '"인공지능" 정의',
    '파이썬 장점 3 가지',
    '이 문장을 존댓말로 바꿔줘: "반말 쓰지 마."',
    '다음 중 채소는?: 사과, 감자, 바나나'
]

# 모델 호출 및 콘텐츠 생성 요청
try:
    # 리스트에 있는 프롬프트를 반복문으로 하나씩 호출합니다.
    for prompt in prompt_list:
        print(f"\n--- 요청된 프롬프트 ---")
        print(f"입력: {prompt}")

        response = client.models.generate_content(
            model='gemini-2.5-flash-lite',                          # 모델
호출
            contents=prompt                                         # 반복
문으로 가져온 프롬프트 하나를 입력값으로 사용
        )

        # 응답 텍스트 출력
        print("\n--- 모델 응답 텍스트 ---")
        print(response.text)
        print("-" * 30)

    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    print("\n--- 모델 응답 전체 JSON ---")
    print(response.model_dump_json(
        exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력

```
--- 요청된 프롬프트 ---
입력: 대한민국의 수도는?

--- 모델 응답 텍스트 ---
대한민국의 수도는 **서울**입니다.
------------------------------
```

--- 요청된 프롬프트 ---
입력: "인공지능" 정의

--- 모델 응답 텍스트 ---
인공지능(Artificial Intelligence, AI) 정의

인공지능(AI)은 **인간의 학습 능력, 추론 능력, 지각 능력, 자연어 이해 능력 등을 컴퓨터 프로그램으로 실현한 기술**을 말합니다. 즉, **인간처럼 생각하고, 배우고, 문제를 해결하는 능력**을 가진 기계나 시스템을 만드는 것을 목표로 하는 분야입니다.

좀 더 자세히 설명하자면, 인공지능은 다음과 같은 특징들을 포함합니다.

*   **학습(Learning):** 경험을 통해 지식을 습득하고, 성능을 향상시킵니다. 데이터를 분석하고 패턴을 인식하여 새로운 상황에 적응할 수 있습니다.
*   **추론(Reasoning):** 주어진 정보를 바탕으로 논리적인 결론을 도출합니다. 규칙이나 논리를 사용하여 문제를 해결합니다.
*   **문제 해결(Problem Solving):** 복잡한 문제를 분석하고 최적의 해결책을 찾습니다. 목표를 달성하기 위한 전략을 수립합니다.
*   **지각(Perception):** 시각, 청각, 촉각 등 감각 정보를 인식하고 이해합니다. 이미지를 인식하거나 음성을 이해하는 능력이 포함됩니다.
*   **자연어 이해(Natural Language Understanding, NLU):** 인간의 언어를 이해하고 처리하며, 생성하는 능력입니다. 챗봇이나 번역 프로그램 등이 이에 해당합니다.

다양한 관점에서 인공지능을 정의할 수 있습니다.

*   **인간처럼 생각하는 시스템(Thinking Humanly):** 인간의 사고 과정을 모델링하고 이를 컴퓨터 시스템에 구현하려는 접근 방식입니다.
*   **인간처럼 행동하는 시스템(Acting Humanly):** 인간과 같이 행동하는 시스템을 만드는 것을 목표로 합니다. 튜링 테스트를 통과하는 것을 예로 들 수 있습니다.
*   **이성적으로 생각하는 시스템(Thinking Rationally):** 논리적 사고나 최적의 의사결정을 하는 시스템을 만드는 데 초점을 맞춥니다.
*   **이성적으로 행동하는 시스템(Acting Rationally):** 합리적인 판단과 행동을 통해 주어진 목표를 효율적으로 달성하는 시스템을 만드는 데 초점을 맞춥니다.

결론적으로 인공지능은 인간의 지능적인 행동을 모방하거나 능가하는 시스템을 구축하는 광범위한 분야이며, 현재에도 끊임없이 발전하고 확장되고 있습니다.

--- 요청된 프롬프트 ---
입력: 파이썬 장점 3 가지

--- 모델 응답 텍스트 ---
파이썬은 매우 인기 있는 프로그래밍 언어이며, 그 이유는 다양하지만 주요 장점 3 가지를 꼽자면 다음과 같습니다.

1. **쉬운 학습 곡선과 높은 가독성:**
    *   **쉬운 문법:** 파이썬은 문법이 간결하고 사람이 읽고 이해하기 쉬운 방식으로 설계되었습니다. 영어와 유사한 키워드를 많이 사용하며, 들여쓰기를 통해 코드 블록을 구분

하기 때문에 다른 언어에 비해 배우기가 훨씬 수월합니다.
    *   **높은 가독성:** 명확한 들여쓰기 규칙 덕분에 코드를 읽는 것이 직관적입니다. 이는 협업 시 다른 개발자가 코드를 이해하고 수정하는 데 큰 도움이 되며, 유지보수성 또한 높입니다. 초보자도 빠르게 프로그래밍의 재미를 느낄 수 있게 해줍니다.

2. **광범위한 라이브러리와 프레임워크 생태계:**
    *   **풍부한 표준 라이브러리:** 파이썬은 기본적으로 다양한 기능을 제공하는 방대한 표준 라이브러리를 가지고 있습니다. 파일 처리, 네트워크 통신, 데이터베이스 연동 등 일상적인 프로그래밍 작업에 필요한 대부분의 기능이 내장되어 있습니다.
    *   **강력한 서드파티 라이브러리:** 또한, NumPy, Pandas, SciPy (과학 및 데이터 분석), TensorFlow, PyTorch (머신러닝), Django, Flask (웹 개발), Matplotlib, Seaborn (데이터 시각화) 등 수많은 고품질의 서드파티 라이브러리와 프레임워크가 존재합니다. 이러한 라이브러리 덕분에 복잡한 기능을 처음부터 직접 구현할 필요 없이 기존의 검증된 코드를 활용하여 개발 생산성을 극대화할 수 있습니다.

3. **다양한 분야에서의 활용성 (다목적성):**
    *   **높은 범용성:** 파이썬은 특정 분야에 국한되지 않고 매우 다양한 용도로 활용될 수 있는 다목적 언어입니다.
    *   **주요 활용 분야:**
        *   **웹 개발:** Django, Flask 등의 프레임워크를 이용해 빠르고 효율적인 웹 애플리케이션 개발이 가능합니다.
        *   **데이터 과학 및 분석:** NumPy, Pandas, SciPy, Scikit-learn 등의 라이브러리를 통해 데이터 처리, 분석, 시각화, 머신러닝 등 전문적인 작업을 수행할 수 있습니다.
        *   **인공지능(AI) 및 머신러닝:** TensorFlow, PyTorch와 같은 강력한 라이브러리를 통해 AI 및 머신러닝 모델 개발에 핵심적인 역할을 합니다.
        *   **자동화 및 스크립팅:** 반복적인 작업을 자동화하는 스크립트 작성에 매우 뛰어나며, 시스템 관리, 테스트 자동화 등 다양한 곳에 활용됩니다.
        *   **데스크톱 애플리케이션 개발, 게임 개발, 교육 등** 다양한 분야에서도 파이썬이 사용됩니다.

이 세 가지 장점 덕분에 파이썬은 초보 개발자부터 숙련된 개발자까지 폭넓은 사용자층에게 사랑받고 있으며, IT 산업 전반에서 중요한 위치를 차지하고 있습니다.

--- 요청된 프롬프트 ---
입력: 이 문장을 존댓말로 바꿔줘: "반말 쓰지 마."

--- 모델 응답 텍스트 ---
"반말 쓰지 마."를 존댓말로 바꾸면 다음과 같은 표현들을 사용할 수 있습니다.

* **"반말은 사용하지 말아 주세요."** (가장 일반적이고 정중한 표현)
* **"존댓말을 써 주시면 감사하겠습니다."** (부드럽고 간곡한 표현)
* **"말씀하실 때 존댓말을 써 주시면 좋겠습니다."** (상황에 따라 사용할 수 있는 표현)
* **"함부로 반말하지 마세요."** (조금 더 직접적인 표현이지만 존댓말)

어떤 상황에서 사용하느냐에 따라 가장 자연스러운 표현을 선택하시면 됩니다.

```
--- 요청된 프롬프트 ---
입력: 다음 중 채소는?: 사과, 감자, 바나나

--- 모델 응답 텍스트 ---
제시하신 과일 및 채소 목록에서 채소는 **감자**입니다.

*   **사과:** 과일
*   **감자:** 채소 (구체적으로는 덩이줄기 채소)
*   **바나나:** 과일
------------------------------

--- 모델 응답 전체 JSON ---

{
    "sdk_http_response": {
        "headers": {
            "content-type": "application/json; charset=UTF-8",
            "vary": "Origin, X-Origin, Referer",
            "content-encoding": "gzip",
            "date": "Mon, 04 Aug 2025 09:11:00 GMT",
            "server": "scaffolding on HTTPServer2",
            "x-xss-protection": "0",
            "x-frame-options": "SAMEORIGIN",
            "x-content-type-options": "nosniff",
            "server-timing": "gfet4t7; dur=771",
            "alt-svc": "h3=\":443\"; ma=2592000,h3-29=\":443\";
ma=2592000",
            "transfer-encoding": "chunked"
        }
    },
    "candidates": [
        {
            "content": {
                "parts": [
                    {
                        "text": "제시하신 과일 및 채소 목록에서 채소는 **감
자**입니다.\n\n*   **사과:** 과일\n*   **감자:** 채소 (구체적으로는 덩이줄기
채소)\n*   **바나나:** 과일"
                    }
                ],
                "role": "model"
            },
            "finish_reason": "STOP",
            "index": 0
        }
    ],
    "model_version": "gemini-2.5-flash-lite",
    "usage_metadata": {
        "candidates_token_count": 60,
        "prompt_token_count": 16,
```

```
        "prompt_tokens_details": [
            {
                "modality": "TEXT",
                "token_count": 16
            }
        ],
        "total_token_count": 76
    },
    "automatic_function_calling_history": []
}
```

---

2) Be specific, and well-defined (구체적, 명확하게 정의하기)

지구를 창의적으로 묘사할 방법을 브레인스토밍하고 싶다고 가정해 봅시다.

a. 구글 클라우드 실습 환경에서의 코드

- **비추천** - 아래 프롬프트는 다소 일반적일 수 있습니다 (물론 일반적인 질문을 하고 싶다면 전혀 문제 없습니다!).
- 실습 코드

```
    prompt = "Tell me about Earth"

    response = client.models.generate_content(model=MODEL_ID,
contents=prompt)
    display(Markdown(response.text))
```

- 대답

```
Earth is an extraordinary and unique planet, often called "The
Blue Planet" or "Terra." It's our home, and so far, the only
known place in the universe to harbor life.

Here's a breakdown of what makes Earth so special:

1.  **Third Planet from the Sun:** Earth orbits the Sun at an
average distance of about 150 million kilometers (93 million
miles), a distance that allows for liquid water to exist on its
surface.

2.  **Home to Life:** This is Earth's most defining
characteristic. Its unique combination of factors has allowed
life to not only emerge but thrive and diversify into millions of
species over billions of years, from microscopic organisms to
towering trees and complex animals.
```

3. **Abundance of Liquid Water:** Approximately 71% of Earth's surface is covered by water, primarily in its vast oceans. Water is absolutely essential for all known life forms and plays a crucial role in regulating Earth's climate.

4. **Dynamic Atmosphere:** Earth's atmosphere is a thin, life-sustaining layer of gases held in place by gravity. It's composed primarily of nitrogen (78%), oxygen (21%), argon (0.9%), and carbon dioxide (0.04%), along with trace amounts of other gases. It:
    * **Provides breathable air:** For most living things.
    * **Protects from harmful radiation:** The ozone layer within the stratosphere absorbs most of the Sun's harmful ultraviolet (UV) radiation.
    * **Regulates temperature:** Traps heat, preventing extreme temperature swings between day and night (the greenhouse effect).
    * **Drives weather:** The movement of air and water in the atmosphere creates our weather systems.

5. **Geologically Active:** Earth is not a static ball of rock. Its interior is still hot and active, driving processes like:
    * **Plate Tectonics:** The outer layer (lithosphere) is broken into large plates that constantly move, causing earthquakes, volcanic eruptions, and the formation of mountains and ocean trenches. This continuous recycling of Earth's crust is vital for many geochemical cycles.
    * **Volcanism and Earthquakes:** These phenomena are direct results of plate movement and internal heat, shaping the planet's surface.

6. **Protective Magnetic Field (Magnetosphere):** Generated by the molten iron and nickel in Earth's outer core, this magnetic field extends far into space, deflecting harmful charged particles from the Sun (the solar wind) and cosmic rays. Without it, our atmosphere would likely be stripped away over time.

7. **Single Natural Satellite: The Moon:** Earth has one large moon, which plays a significant role in stabilizing Earth's axial tilt (preventing drastic climate shifts over long periods) and creating ocean tides.

8. **Varied Climates and Ecosystems:** Due to its tilt on its axis (causing seasons), rotation, and diverse topography, Earth features an incredible range of climates (polar, temperate, tropical) and ecosystems (forests, deserts, grasslands, oceans, tundras) that support a vast array of life.

In essence, Earth is a complex, interconnected system where geology, atmosphere, water, and life constantly interact and

```
influence each other. It's a testament to the delicate balance of conditions required for a planet to become a vibrant, living world.
```

- 추천합니다. 아래 프롬프트는 구체적이고 명확하게 정의되어 있습니다.
- 실습 코드

```
    prompt = "Generate a list of ways that makes Earth unique compared to other planets"

    response = client.models.generate_content(model=MODEL_ID, contents=prompt)
    display(Markdown(response.text))
```

- 대답

```
Earth stands out in our solar system, and likely many exoplanet systems discovered so far, due to a unique combination of factors that collectively make it a haven for life. Here's a list of ways Earth is unique:

1.  **Abundant and Stable Liquid Water:** Earth is the only known planet to have stable bodies of liquid water on its surface. While other planets and moons have ice (Mars, Europa, Enceladus) or evidence of past liquid water (Mars), only Earth maintains a vast, dynamic ocean and surface water cycle crucial for life.

2.  **Robust and Diverse Life:** Earth is the only known planet teeming with life – from microscopic organisms to complex, multi-cellular beings, including intelligent life. This biodiversity and the intricate ecosystems it forms are unparalleled.

3.  **Just-Right Atmosphere:**
    *   **Composition:** Earth's atmosphere is unique for its high concentration of free oxygen (around 21%), which is a byproduct of photosynthesis and essential for most complex life forms. It also has a significant amount of nitrogen (78%) and trace gases like argon and carbon dioxide.
    *   **Greenhouse Effect:** It has a natural greenhouse effect that traps enough heat to keep the planet warm enough for liquid water, without being runaway like Venus.
    *   **Ozone Layer:** A vital layer within the stratosphere protects surface life from harmful solar ultraviolet (UV) radiation.

4.  **Active Plate Tectonics:** Earth's crust is broken into
```

```
large plates that are constantly moving, colliding, and
subducting. This process drives volcanism, mountain building, and
earthquake activity, and is crucial for:
    *   **Climate Regulation:** It recycles carbon dioxide back
into the atmosphere, helping to regulate Earth's long-term
climate.
    *   **Nutrient Cycling:** Brings fresh minerals to the
surface.
    *   **Creating Diverse Habitats:** Contributes to varied
topography and ecosystems.

5.  **Strong Global Magnetic Field (Magnetosphere):** Generated
by the convection of molten iron in its outer core, Earth's
magnetic field acts as a protective shield. It deflects harmful
charged particles from the solar wind and cosmic rays, preventing
them from stripping away our atmosphere or directly damaging life
on the surface.

6.  **"Goldilocks Zone" Orbit:** Earth orbits the Sun at just the
right distance – not too hot, not too cold – allowing liquid
water to exist on its surface. This "habitable zone" is critical
for life as we know it.

7.  **Large, Stabilizing Moon:** Earth has an unusually large
moon relative to its size. This large moon:
    *   **Stabilizes Axial Tilt:** Helps stabilize Earth's axial
tilt, preventing dramatic wobbles that would lead to extreme
climate variations. This stability contributes to predictable
seasons and a relatively stable climate over long periods.
    *   **Tides:** Creates significant ocean tides, which may
have played a role in the emergence of life from the oceans.

8.  **Internal Heat and Dynamic Core:** Earth retains significant
internal heat from its formation and radioactive decay, which
drives plate tectonics and generates the magnetic field. This
dynamic interior is unique compared to colder, geologically dead
planets like Mars or the inert cores of gas giants.

9.  **Presence of Intelligent, Self-Aware Life (Humanity):**
While this is a consequence of the unique conditions, the
existence of a species capable of science, art, philosophy, and
technology is, as far as we know, unique to Earth in the
observable universe.

It's the *combination* and *interplay* of these factors that make
Earth truly unique, creating a dynamic and incredibly hospitable
environment for life.
```

b. 로컬

- 구글 클라우드 실습 때와 같은 질문 해보기
  - ver_en / ver_ko

구	프롬프트 내용	이유
비	- Tell me about Earth - 지구에 대해 말해 주세요	너무 일반적이고 범위가 넓어서 응답이 모호하거나 너무 길어질 수 있음. 목적이 불명확함.
추	- Generate a list of ways that makes Earth unique compared to other planets. - 지구가 다른 행성과 비교해 독특한 점들을 목록으로 생성해 주세요.	비교 기준(다른 행성)과 주제(지구의 독특함), 출력 형태(목록)이 명확해서, 더 구체적이고 유용한 답변을 받을 수 있음.

```
# 2_1. 구글 클라우드 실습 질문과 같은 질문으로 입력해보기

prompt = "Tell me about Earth"
prompt_ko = "지구에 대해 말해 주세요"

# --- 첫 번째 프롬프트 호출 (영어) ---
try:
    print("\n--- 첫 번째 요청 (영어) ---")
    response_en = client.models.generate_content(
        model='gemini-2.5-flash-lite',                                  #
사용할 모델 지정
        contents=prompt                                                 #
영어 프롬프트 입력
    )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response_en.text)
    print("-" * 30)

    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    #print("\n--- 모델 응답 전체 JSON ---")
    #print(response.model_dump_json(
    #    exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
```

```
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")

# --- 두 번째 프롬프트 호출 (한국어) ---
try:
    print("\n--- 두 번째 요청 (한국어) ---")
    response_ko = client.models.generate_content(
        model='gemini-2.5-flash-lite', # 사용할 모델 지정
        contents=prompt_ko            # 한국어 프롬프트 입력
    )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_ko}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response_ko.text)
    print("-" * 30)

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력

```
    --- 첫 번째 요청 (영어) ---
    입력 프롬프트: Tell me about Earth

    --- 모델 응답 텍스트 ---
    Earth is a planet of incredible complexity, beauty, and
dynamism. Here's a breakdown of what makes it so special:

    **Our Cosmic Home:**

    *   **Third Planet from the Sun:** Earth is the third planet
in our solar system, orbiting a star we call the Sun. It's part
of a rocky, inner solar system along with Mercury, Venus, and
Mars.
    *   **The Only Known Habitable Planet:** So far, Earth is the
only place in the universe where we know for sure that life
exists. This is a truly remarkable distinction.
    *   **Size and Shape:** It's the fifth largest planet in the
solar system by diameter and the most massive of the terrestrial
```

```
(rocky) planets. While we often call it round, it's actually an
**oblate spheroid**, slightly bulging at the equator due to its
rotation.

    **Key Characteristics and Features:**

    *   **Atmosphere:** Earth is enveloped in a thick blanket of
gases known as the atmosphere. This is crucial for life,
providing:
        *   **Oxygen:** Essential for respiration for most living
organisms.
        *   **Nitrogen:** The most abundant gas, playing a role
in biological processes.
        *   **Carbon Dioxide:** A greenhouse gas that helps
regulate Earth's temperature, though increased levels are causing
climate change.
        *   **Ozone Layer:** Protects life from harmful
ultraviolet (UV) radiation from the Sun.
        *   **Weather:** The atmosphere is where weather
phenomena like clouds, rain, wind, and storms occur.
    *   **Water:** Perhaps Earth's most defining feature is the
abundance of **liquid water**. Covering about 71% of the surface,
oceans, rivers, lakes, and ice are vital for life and shape our
planet's climate and geography.
    *   **Geology:**
        *   **Layers:** Earth has distinct layers:
            *   **Crust:** The thin, outermost rocky shell where
we live.
            *   **Mantle:** A thick layer of hot, semi-molten
rock.
            *   **Outer Core:** A liquid layer of iron and
nickel.
            *   **Inner Core:** A solid ball of iron and nickel,
incredibly hot and under immense pressure.
        *   **Plate Tectonics:** The Earth's crust is broken into
large pieces called tectonic plates that move slowly over the
mantle. This movement causes earthquakes, volcanic eruptions, and
the formation of mountains and ocean trenches.
        *   **Volcanoes and Mountains:** These are dramatic
geological features formed by internal Earth processes.
    *   **Biodiversity:** Earth teems with an astonishing variety
of life, from microscopic bacteria to giant whales. This
biodiversity is interconnected and forms complex ecosystems.
    *   **Magnetosphere:** Earth has a powerful magnetic field
generated by its molten outer core. This magnetosphere shields us
from harmful solar wind and cosmic radiation.
    *   **Rotation and Revolution:**
        *   **Rotation:** Earth spins on its axis, completing one
rotation approximately every 24 hours, which gives us day and
```

night.
        *   **Revolution:** Earth orbits the Sun, completing one revolution approximately every 365.25 days, which we define as a year. This orbit, combined with Earth's axial tilt, causes the seasons.

    Why Earth is Habitable (The "Goldilocks Zone"):

    Earth is in the **habitable zone** (also known as the "Goldilocks zone") of our solar system, meaning it's at a distance from the Sun where liquid water can exist on its surface. This is crucial because liquid water is considered essential for life as we know it.

    Challenges and Threats:

    Despite its resilience, Earth faces significant challenges, many of which are caused by human activities:

    *   **Climate Change:** Rising global temperatures due to greenhouse gas emissions are altering weather patterns, melting ice caps, and threatening ecosystems.
    *   **Pollution:** Air, water, and soil pollution degrade habitats and harm living organisms.
    *   **Deforestation and Habitat Loss:** The destruction of natural environments leads to biodiversity loss.
    *   **Resource Depletion:** Overconsumption of natural resources puts a strain on the planet's ability to sustain life.

    In Summary:

    Earth is a dynamic, living planet with a unique combination of factors that make it suitable for life. It's a place of breathtaking natural beauty, intricate ecosystems, and continuous geological activity. Understanding and protecting our planet is of paramount importance for the future of all life.

    --- 두 번째 요청 (한국어) ---
    입력 프롬프트: 지구에 대해 말해 주세요

    --- 모델 응답 텍스트 ---
    지구는 태양계에서 다섯 번째로 큰 행성이자, 유일하게 생명체가 존재하는 것으로 알려진 행성입니다. 매우 특별하고 아름다운 행성이지요. 지구에 대해 몇 가지 흥미로운 사실들을 알려드릴게요.

    지구의 기본적인 정보:

    *   **나이:** 약 45 억 4 천만 년
    *   **태양으로부터의 거리:** 평균 약 1 억 5 천만 킬로미터 (1 AU - 천문 단위)

```
    *   **지름:** 약 12,742 킬로미터
    *   **표면:** 약 71%가 물 (바다, 강, 호수 등)로 덮여 있고, 나머지 29%는 육지입니다.
    *   **대기:** 질소(약 78%), 산소(약 21%)를 주성분으로 하는 대기로 둘러싸여 있어 생명체가 숨 쉴 수 있습니다.
    *   **자전:** 약 24 시간마다 한 바퀴 자전하여 낮과 밤이 생깁니다.
    *   **공전:** 약 365.25 일마다 태양 주위를 한 바퀴 공전하여 1 년이 됩니다.
    *   **위성:** 달이 하나 있습니다.

    **지구의 특징:**

    *   **생명체의 보고:** 지구는 우리가 아는 한 유일하게 생명체가 살고 있는 행성입니다. 엄청나게 다양한 생물들이 서식하며, 복잡한 생태계를 이루고 있습니다.
    *   **물의 행성:** 액체 상태의 물이 풍부하다는 점이 지구의 가장 큰 특징 중 하나입니다. 이 물 덕분에 생명체가 탄생하고 유지될 수 있었습니다.
    *   **다양한 기후:** 극지방의 추운 날씨부터 적도 부근의 더운 날씨까지, 지구에는 매우 다양한 기후대가 존재합니다. 이는 다양한 생태계와 문명의 발달을 가능하게 했습니다.
    *   **판 구조론:** 지구의 표면은 여러 개의 거대한 판으로 이루어져 있으며, 이 판들이 움직이면서 지진, 화산 활동, 산맥 형성 등 다양한 지질 현상을 일으킵니다.
    *   **자전축 기울기:** 지구의 자전축은 약 23.5 도 기울어져 있습니다. 이 기울기 때문에 계절의 변화가 나타납니다.

    **지구를 보호하는 것의 중요성:**

    지구는 우리에게 생명과 모든 것을 제공하는 소중한 터전입니다. 하지만 인간의 활동으로 인해 환경 오염, 기후 변화, 생물 다양성 감소 등 다양한 문제에 직면하고 있습니다. 따라서 지구를 보호하고 지속 가능한 미래를 만들기 위한 노력이 매우 중요합니다.

    지구에 대해 더 궁금한 점이 있으신가요? 어떤 특정 부분에 대해 더 자세히 알고 싶으시면 언제든지 물어보세요!
    ------------------------------
```

- 비슷한 유형으로 질문 해보기
  - ver_en

비추천	추천
Tell me about the ocean	List 5 ways the ocean supports human life
Explain AI	Compare machine learning and deep learning with examples
What is history?	Summarize the causes of World War I in 3 bullet points

- ver_ko

	비추천	추천
	바다에 대해 말해 주세요	바다가 인간의 삶을 지탱하는 5 가지 방법을 나열해 주세요.
	인공지능에 대해 설명해 주세요	머신러닝과 딥러닝을 예시와 함께 비교해 주세요.
	역사가 뭐예요?	제 1 차 세계대전의 원인을 세 가지로 요약해 주세요.

```
# 2.2 - 비추천/추천 프롬프트 작성 (유사한 유형)

# ver_en 비추천 리스트
prompt_list_en_not_rec = [
    "Tell me about the ocean",
    "Explain AI",
    "What is history?"
]

# ver_ko 비추천 리스트
prompt_list_ko_not_rec = [
    "바다에 대해 말해 주세요.",
    "인공지능에 대해 설명해 주세요.",
    "역사가 뭐예요?"
]

# ver_en 추천 리스트
prompt_list_en_rec = [
    "List 5 ways the ocean supports human life",
    "Compare machine learning and deep learning with examples",
    "Summarize the causes of World War I in 3 bullet points"
]

# ver_ko 추천 리스트
prompt_list_ko_rec = [
    "바다가 인간의 삶을 지탱하는 5 가지 방법을 나열해 주세요.",
    "머신러닝과 딥러닝을 예시와 함께 비교해 주세요.",
    "제 1 차 세계대전의 원인을 세 가지로 요약해 주세요."
]

# 비추천 리스트 프롬프트 질문하기

# --- 첫 번째 비추천 리스트 프롬프트 호출 (영어) ---
# prompt_list_en_not_rec
try:
    # 리스트에 있는 프롬프트를 반복문으로 하나씩 호출
    for prompt in prompt_list_en_not_rec:
        print(f"\n--- 요청된 프롬프트 ---")
        print(f"입력: {prompt}")

        response = client.models.generate_content(
            model='gemini-2.5-flash-lite',                    # 모델
```

```
호출
            contents=prompt                                                    # 반복
문으로 가져온 프롬프트 하나를 입력값으로 사용
        )

        # 응답 텍스트 출력
        print("\n--- 모델 응답 텍스트 ---")
        print(response.text)
        print("-" * 30)

    # ---------- 디버깅용 출력하지 않기 -------------
    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    # print("\n--- 모델 응답 전체JSON ---")
    # print(response.model_dump_json(
    #     exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력

```
--- 요청된 프롬프트 ---
입력: Tell me about the ocean

--- 모델 응답 텍스트 ---
The ocean is a vast, interconnected body of saltwater that covers
approximately **71% of the Earth's surface**. It's a truly awe-
inspiring and essential part of our planet, teeming with life,
driving global weather patterns, and holding immense geological
and chemical significance.

Here's a breakdown of what makes the ocean so incredible:

**1. Immense Scale and Depth:**

* **Vastness:** Imagine a body of water so large it encompasses
all the continents. The Pacific Ocean alone is larger than all
the landmasses combined!
* **Depth:** The average depth is around 3,688 meters (12,100
feet), but the deepest known point is the **Challenger Deep** in
the Mariana Trench, reaching nearly 11,000 meters (36,000 feet) –
deeper than Mount Everest is tall!
```

```
**2. A World of Life:**

The ocean is Earth's largest habitat, supporting an incredible diversity of life, from microscopic plankton to the colossal blue whale.

* **Marine Ecosystems:** These are incredibly varied and include:
    * **Coral Reefs:** Vibrant, biodiverse "rainforests of the sea" providing habitats for countless species.
    * **Kelp Forests:** Underwater forests of large brown algae, offering shelter and food.
    * **Deep Sea:** An alien world of perpetual darkness, with unique creatures adapted to extreme pressure and cold.
    * **Estuaries:** Where freshwater rivers meet saltwater oceans, creating highly productive transitional zones.
    * **Open Ocean (Pelagic Zone):** The vast, sunlit surface waters and the darker depths beyond the continental shelves.
* **Biodiversity:** Scientists estimate that 50-80% of all life on Earth is found in the ocean. This includes:
    * **Plankton:** The foundation of most marine food webs, including phytoplankton (algae) and zooplankton (tiny animals).
    * **Fish:** An astonishing array of species, from tiny seahorses to massive sharks.
    * **Mammals:** Whales, dolphins, seals, and sea otters.
    * **Invertebrates:** Jellyfish, squid, octopuses, crabs, lobsters, sea stars, and countless others.
    * **Seaweeds and Algae:** The "plants" of the ocean, playing a crucial role in oxygen production.

**3. Powerful Forces at Play:**

The ocean is a dynamic system, constantly shaped by powerful natural forces.

* **Tides:** The rhythmic rise and fall of sea levels caused by the gravitational pull of the Moon and Sun.
* **Ocean Currents:** Massive, continuous movements of seawater that circulate heat, nutrients, and marine life around the globe. These currents are vital for regulating climate.
* **Waves:** Generated by wind, waves are a visible manifestation of the ocean's energy.
* **Storms:** Hurricanes, typhoons, and cyclones are powerful weather events that form over warm ocean waters, releasing immense energy.

**4. Essential for Earth's Systems:**

The ocean plays a critical role in maintaining the health and stability of our planet.
```

```
* **Climate Regulation:** The ocean absorbs a significant amount of heat from the atmosphere, buffering temperature changes and distributing heat around the globe through currents. It also absorbs a large portion of human-emitted carbon dioxide, though this comes at a cost to ocean chemistry.
* **Oxygen Production:** Phytoplankton in the ocean produce roughly half of the oxygen we breathe.
* **Water Cycle:** The ocean is the primary source of evaporation, driving the global water cycle.
* **Nutrient Cycling:** Ocean currents transport essential nutrients that support marine ecosystems and, indirectly, terrestrial life.

**5. Geological Significance:**

The ocean floor is not a flat, featureless expanse. It's a dynamic landscape shaped by geological processes.

* **Mid-Ocean Ridges:** Underwater mountain ranges where new oceanic crust is formed through volcanic activity.
* **Trenches:** Deep valleys formed where one tectonic plate subducts beneath another.
* **Seamounts:** Underwater mountains, often volcanic in origin.
* **Continental Shelves and Slopes:** The underwater extensions of continents.

**6. Human Connection and Impact:**

Humans have always relied on and interacted with the ocean.

* **Food Source:** The ocean provides a significant portion of the world's protein through fishing and aquaculture.
* **Transportation:** Oceans are vital for global trade and travel.
* **Recreation:** Millions of people enjoy activities like swimming, surfing, sailing, and diving.
* **Resources:** The ocean is a source of minerals, oil, and gas.
* **Threats:** Human activities pose significant threats to the ocean's health, including:
    * **Pollution:** Plastic, chemical, and nutrient pollution degrade habitats and harm marine life.
    * **Overfishing:** Depleting fish stocks and disrupting food webs.
    * **Climate Change:** Ocean acidification and warming threaten coral reefs and other ecosystems.
    * **Habitat Destruction:** Coastal development and destructive fishing practices damage marine environments.

**The ocean is a complex, beautiful, and vital system that we are still learning about.** Understanding and protecting it is
```

```
crucial for the well-being of our planet and all life on it.

What aspects of the ocean are you most interested in? We can dive
deeper into specific topics like marine life, oceanography, or
conservation efforts!
------------------------------

--- 요청된 프롬프트 ---
입력: Explain AI

--- 모델 응답 텍스트 ---
**Artificial Intelligence (AI)** is a broad field of computer
science dedicated to creating systems that can perform tasks that
typically require human intelligence. In essence, AI aims to
build machines that can **think, learn, and act** in ways we
associate with humans.

Let's break down what that means:

**Core Concepts of AI:**

*   **Learning:** This is a fundamental aspect of AI. Instead of
being explicitly programmed for every scenario, AI systems can
learn from data. This learning can take various forms:
    *   **Machine Learning (ML):** A subset of AI where
algorithms are trained on data to identify patterns and make
predictions or decisions without being explicitly programmed for
each specific task. Think of it as teaching a computer by showing
it many examples.
    *   **Deep Learning (DL):** A subfield of ML that uses
artificial neural networks with multiple layers (hence "deep").
These networks are inspired by the structure and function of the
human brain and are particularly good at processing complex data
like images, audio, and text.
*   **Reasoning and Problem-Solving:** AI systems can analyze
information, draw conclusions, and find solutions to problems.
This can range from simple logical deductions to complex
strategic planning.
*   **Perception:** AI can process sensory information from the
environment, such as images (computer vision), sounds (speech
recognition), and even touch.
*   **Understanding and Generation:** AI can understand natural
language (Natural Language Processing or NLP) and even generate
human-like text, speech, or other forms of content.
*   **Action and Robotics:** AI can be used to control physical
robots, enabling them to perform tasks in the real world, from
manufacturing to exploration.

**How AI Works (Simplified):**
```

1. **Data:** AI systems, especially ML and DL, require vast amounts of data to learn. This data can be anything from images of cats to customer purchase histories to historical stock market prices.
2. **Algorithms:** These are the sets of rules and instructions that the AI uses to process data, learn patterns, and make decisions. Different algorithms are suited for different tasks.
3. **Training:** During the training phase, the AI algorithm is fed the data, and it adjusts its internal parameters to minimize errors and improve its performance on the task it's designed for.
4. **Inference/Prediction:** Once trained, the AI can be presented with new, unseen data and use its learned knowledge to make predictions, classify information, or take actions.

Types of AI:

AI is often categorized based on its capabilities:

* **Narrow AI (or Weak AI):** This is the AI we have today. It's designed and trained for a specific task or a narrow set of tasks. Examples include:
    * Virtual assistants like Siri and Alexa
    * Image recognition software
    * Recommendation engines (like on Netflix or Amazon)
    * Spam filters
    * Self-driving car systems (though very complex, they are still focused on driving)
* **General AI (or Strong AI):** This is hypothetical AI that would possess human-level intelligence across a wide range of tasks. It would be able to understand, learn, and apply knowledge to any intellectual task that a human can. This type of AI does not currently exist.
* **Super AI:** This is also hypothetical AI that would surpass human intelligence in virtually every field, including scientific creativity, general wisdom, and social skills.

Why is AI Important?

AI has the potential to revolutionize almost every aspect of our lives and industries:

* **Automation:** Automating repetitive or dangerous tasks, freeing up humans for more creative or strategic work.
* **Efficiency and Productivity:** Optimizing processes, improving decision-making, and boosting output.
* **Personalization:** Tailoring experiences for individuals, from education to healthcare.
* **Discovery:** Accelerating scientific research, drug discovery, and understanding complex phenomena.
* **New Capabilities:** Enabling technologies that were

previously impossible, like sophisticated language translation or truly intelligent robotics.

Potential Challenges and Ethical Considerations:

As AI becomes more powerful, it also raises important questions:

*   **Bias:** AI systems can inherit biases present in the data they are trained on, leading to unfair or discriminatory outcomes.
*   **Job Displacement:** Automation powered by AI could lead to significant changes in the job market.
*   **Privacy:** The collection and use of vast amounts of data for AI raise privacy concerns.
*   **Security:** AI systems can be vulnerable to malicious attacks.
*   **Accountability and Ethics:** Determining responsibility when AI systems make mistakes or cause harm.
*   **The "Black Box" Problem:** Sometimes, it's difficult to understand exactly *why* an AI made a particular decision, especially with complex deep learning models.

In summary, AI is the pursuit of creating intelligent machines. It's a rapidly evolving field with immense potential to transform our world, but it also requires careful consideration of its ethical and societal implications.

--- 요청된 프롬프트 ---
입력: What is history?

--- 모델 응답 텍스트 ---
History is a vast and multifaceted concept, and there's no single, universally accepted definition. However, we can break it down into several key components:

At its core, history is the:

*   **Study of the past:** This is the most fundamental aspect. History involves investigating, interpreting, and narrating events, people, societies, and cultures that have occurred before the present.

More specifically, history is:

*   **A record of human experience:** It's about understanding what it means to be human across different times and places. This includes our triumphs, failures, innovations, conflicts, beliefs, and aspirations.
*   **An interpretation of evidence:** Historians don't just

```
*know* what happened; they *reconstruct* it. This is done by
examining a variety of sources, such as:
    *   **Primary sources:** These are materials created by
people who directly experienced the event or time period (e.g.,
letters, diaries, photographs, government documents, artifacts).
    *   **Secondary sources:** These are works written by
historians who have analyzed primary sources and other secondary
sources (e.g., academic books, journal articles).
*   **A process of inquiry and analysis:** History is not just
memorizing dates and names. It involves asking critical
questions, forming hypotheses, evaluating evidence, and
developing arguments.
*   **A narrative and a story:** While grounded in evidence,
history is also presented as a narrative. The way events are
sequenced, explained, and connected creates a story that helps us
make sense of the past.
*   **A way of understanding the present:** By studying the past,
we can gain insights into why the world is the way it is today.
It helps us understand the origins of our institutions, cultures,
conflicts, and opportunities.
*   **A discipline with methodologies:** Historians follow
established methods for research, source criticism, and
argumentation. This ensures a degree of rigor and scholarly
accountability.
*   **A constantly evolving field:** New evidence is discovered,
and new perspectives emerge, meaning our understanding of the
past is always being revised and enriched.

**In essence, history is:**

*   **What happened:** The actual events, people, and processes
of the past.
*   **How we understand what happened:** The process of research,
interpretation, and writing that creates our knowledge of the
past.

**Why is history important?**

*   **To understand ourselves:** Our identities are shaped by our
past, both individually and collectively.
*   **To learn from mistakes:** History offers valuable lessons
about what works and what doesn't, helping us avoid repeating
errors.
*   **To appreciate diversity:** It exposes us to different
cultures, societies, and ways of life, fostering empathy and
understanding.
*   **To critically evaluate information:** By understanding how
historical narratives are constructed, we can become more
discerning consumers of information in the present.
*   **To inform the future:** A strong understanding of the past
```

```
    is crucial for making informed decisions about the future.

    So, history is more than just a collection of facts; it's a
    dynamic, interpretive, and essential endeavor that helps us
    understand the human journey and our place within it.
    ------------------------------
```

```
# --- 두 번째 비추천 리스트 프롬프트 호출 (한국어) ---
# prompt_list_ko_not_rec
try:
    # 리스트에 있는 프롬프트를 반복문으로 하나씩 호출
    for prompt in prompt_list_ko_not_rec:
        print(f"\n--- 요청된 프롬프트 ---")
        print(f"입력: {prompt}")

        response = client.models.generate_content(
            model='gemini-2.5-flash-lite',                          # 모델
호출
            contents=prompt                                         # 반복
문으로 가져온 프롬프트 하나를 입력값으로 사용
        )

        # 응답 텍스트 출력
        print("\n--- 모델 응답 텍스트 ---")
        print(response.text)
        print("-" * 30)

    # ---------- 디버깅용 출력하지 않기 -------------
    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    # print("\n--- 모델 응답 전체 JSON ---")
    # print(response.model_dump_json(
    #     exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력

```
    --- 요청된 프롬프트 ---
    입력: 바다에 대해 말해 주세요.

    --- 모델 응답 텍스트 ---
```

```
바다는 지구 표면의 약 71%를 덮고 있는 광대한 수역입니다. 우리 행성에서 가장 중요한 생태계 중 하나이며, 지구의 기후를 조절하고 수많은 생물에게 서식지를 제공합니다.

**바다의 몇 가지 흥미로운 사실:**

*   **깊이:** 지구상에서 가장 깊은 지점은 마리아나 해구로, 약 11,000 미터 깊이에 달합니다.
*   **염분:** 바닷물은 평균적으로 35%의 염분을 함유하고 있습니다.
*   **산소:** 바다는 지구 대기 산소의 약 50%를 생산합니다.
*   **생물 다양성:** 바다는 지구 생물 다양성의 약 70%를 차지합니다. 여기에는 다양한 종류의 어류, 해양 포유류, 조류, 식물 등이 포함됩니다.
*   **조류:** 바다는 조석, 해류, 파도와 같은 다양한 종류의 조류를 가지고 있습니다. 이 조류는 해양 생태계에 중요한 역할을 합니다.
*   **색깔:** 바다의 색깔은 빛의 산란과 물 속에 포함된 플랑크톤의 종류에 따라 달라집니다. 맑은 날에는 푸르게 보이지만, 탁한 날에는 회색이나 갈색으로 보일 수도 있습니다.
*   **기후 조절:** 바다는 열을 흡수하고 방출하며, 대기와 열을 교환하여 지구의 기후를 조절하는 중요한 역할을 합니다.

**바다의 중요성:**

*   **식량 공급:** 바다는 전 세계 인구에게 단백질의 주요 공급원입니다.
*   **경제:** 어업, 관광, 해상 운송 등 많은 산업이 바다에 의존하고 있습니다.
*   **기후 조절:** 앞서 언급했듯이 바다는 지구의 기후를 안정시키는 데 필수적입니다.
*   **생물 다양성:** 바다는 우리가 아직 완전히 이해하지 못하는 놀라운 생물 다양성의 보고입니다.

**바다와 관련된 문제점:**

*   **오염:** 플라스틱, 화학 물질, 하수 등이 바다를 오염시켜 해양 생태계와 인간 건강에 심각한 위협이 되고 있습니다.
*   **남획:** 과도한 어획은 일부 어종의 개체 수를 감소시키고 해양 생태계의 균형을 깨뜨립니다.
*   **기후 변화:** 지구 온난화로 인한 해수면 상승, 해양 산성화, 산호초 백화 현상 등은 바다에 큰 영향을 미치고 있습니다.

바다는 우리에게 많은 것을 제공하지만, 동시에 우리가 보호해야 할 소중한 자원이기도 합니다. 바다를 깨끗하게 유지하고 지속 가능한 방식으로 이용하기 위한 노력이 매우 중요합니다.

바다에 대해 더 궁금한 점이 있으시면 언제든지 물어보세요!
------------------------------

--- 요청된 프롬프트 ---
입력: 인공지능에 대해 설명해 주세요.

--- 모델 응답 텍스트 ---
## 인공지능(AI)에 대한 설명

인공지능(AI)은 **인간의 학습 능력, 추론 능력, 지각 능력, 자연어 이해 능력 등을 컴퓨터 프로그램으로 실현한 기술**을 의미합니다. 쉽게 말해, **컴퓨터가 사람처럼 생각하고 행동하도록 만드는 것**이라고 할 수 있습니다.
```

AI 는 단순히 프로그래밍된 대로만 움직이는 것이 아니라, **스스로 학습하고 발전하며 복잡한 문제를 해결**하는 능력을 갖추는 것을 목표로 합니다.

AI 의 주요 구성 요소 및 기술:

AI 를 구현하기 위해서는 다양한 기술들이 필요하며, 그 중 핵심적인 요소들은 다음과 같습니다.

*   **머신러닝 (Machine Learning, ML):** AI 의 가장 중요한 분야 중 하나로, **데이터를 통해 컴퓨터가 스스로 학습하고 예측하거나 결정을 내릴 수 있도록 하는 기술**입니다. 명시적으로 프로그래밍되지 않아도, 수많은 데이터를 분석하여 패턴을 파악하고 새로운 데이터에 대한 결과를 도출해냅니다.
    *   **지도 학습 (Supervised Learning):** 정답이 있는 데이터를 사용하여 학습합니다. (예: 스팸 메일 분류)
    *   **비지도 학습 (Unsupervised Learning):** 정답이 없는 데이터를 사용하여 데이터의 숨겨진 패턴이나 구조를 찾습니다. (예: 고객 세분화)
    *   **강화 학습 (Reinforcement Learning):** 시행착오를 통해 보상을 최대화하는 방향으로 학습합니다. (예: 게임 플레이 AI)

*   **딥러닝 (Deep Learning, DL):** 머신러닝의 한 분야로, **인간의 신경망 구조를 모방한 인공 신경망(Artificial Neural Network)을 여러 층으로 쌓아** 더욱 복잡하고 추상적인 특징을 학습할 수 있도록 하는 기술입니다. 이미지 인식, 음성 인식, 자연어 처리 등에서 뛰어난 성능을 보입니다.

*   **자연어 처리 (Natural Language Processing, NLP):** **컴퓨터가 인간의 언어(자연어)를 이해하고, 생성하며, 처리하는 기술**입니다. 챗봇, 번역, 텍스트 분석 등에 활용됩니다.

*   **컴퓨터 비전 (Computer Vision):** **컴퓨터가 이미지를 인식하고 분석하여 의미를 파악하는 기술**입니다. 자율 주행차의 사물 인식, 의료 영상 분석 등에 사용됩니다.

*   **추론 (Reasoning):** **주어진 정보를 바탕으로 논리적인 결론을 도출하는 능력**을 의미합니다. 문제 해결, 의사 결정 등에 활용됩니다.

*   **지식 표현 (Knowledge Representation):** **컴퓨터가 이해할 수 있는 형태로 지식을 저장하고 관리하는 방법**입니다.

AI 의 종류:

AI 는 그 능력의 범위에 따라 크게 다음과 같이 분류될 수 있습니다.

*   **약한 AI (Narrow AI / Weak AI):** **특정 목적이나 작업에 특화된 AI** 입니다. 현재 우리가 접하는 대부분의 AI 가 여기에 해당합니다. (예: 음성 비서, 추천 시스템, 이미지 인식 AI)

*   **강한 AI (General AI / Strong AI):** **인간과 같이 모든 지적 능력을 갖춘 AI**입니다. 아직은 이론적인 개념이며, 현재 기술 수준으로는 구현되지 않았습니다.

*   **초지능 AI (Superintelligence):** **인간의 지능을 훨씬 뛰어넘는 AI**입

```
니다. 역시 이론적인 개념입니다.

### AI의 활용 분야:

AI는 우리 생활의 거의 모든 영역에서 활용되고 있으며, 그 범위는 계속 확장되고 있습니다.

*   **일상생활:** 스마트폰의 음성 비서, 추천 시스템 (넷플릭스, 유튜브), 스마트 홈 기기, 번역 서비스
*   **산업:** 제조 자동화, 품질 관리, 수요 예측, 물류 최적화
*   **의료:** 질병 진단, 신약 개발, 개인 맞춤형 치료
*   **금융:** 신용 평가, 사기 탐지, 알고리즘 트레이딩
*   **교통:** 자율 주행 자동차, 교통량 예측 및 제어
*   **교육:** 맞춤형 학습 시스템, 학습 분석
*   **예술 및 창작:** 그림, 음악, 글쓰기 등 AI 기반 창작

### AI의 장점과 단점 (고려할 점):

**장점:**

*   **효율성 및 생산성 향상:** 반복적이고 시간이 많이 소요되는 작업을 자동화하여 생산성을 높입니다.
*   **정확도 향상:** 대규모 데이터를 분석하여 인간보다 더 정확한 결과를 도출할 수 있습니다.
*   **새로운 발견 및 혁신:** 인간이 발견하기 어려운 패턴이나 인사이트를 찾아내 혁신을 이끌 수 있습니다.
*   **위험하거나 어려운 작업 수행:** 인간이 하기 어렵거나 위험한 환경에서 작업을 수행할 수 있습니다.

**단점 및 고려할 점:**

*   **일자리 감소 우려:** 자동화로 인해 일부 직업이 사라질 수 있다는 우려가 있습니다.
*   **윤리적 문제:** 편향된 데이터로 인한 차별, 프라이버시 침해, 책임 소재 불분명 등의 윤리적 문제가 발생할 수 있습니다.
*   **보안 문제:** AI 시스템 자체의 보안 취약점이나 악의적인 사용 가능성이 있습니다.
*   **개발 및 유지보수 비용:** AI 시스템을 개발하고 유지보수하는 데 상당한 비용이 발생할 수 있습니다.
*   **AI 의존성 심화:** AI에 대한 과도한 의존은 인간의 비판적 사고 능력이나 문제 해결 능력을 저하시킬 수 있습니다.

### 결론적으로,

인공지능은 현재 우리 사회에 엄청난 영향을 미치고 있으며, 미래 사회의 핵심 동력이 될 것으로 예상됩니다. AI 기술의 발전과 함께 그 잠재력을 최대한 활용하면서도 발생할 수 있는 문제점들을 신중하게 고려하고 해결해 나가는 것이 중요합니다.

궁금하신 점이 있다면 언제든지 다시 질문해주세요!
--------------------------------

--- 요청된 프롬프트 ---
```

```
입력: 역사가 뭐예요?

--- 모델 응답 텍스트 ---
역사는 **과거에 일어났던 일들을 기록하고 연구하며 이해하려는 학문**입니다. 단순히 사건들을 나열하는 것을 넘어, **왜 그런 일이 일어났는지, 그것이 현재에 어떤 영향을 미치고 있는지, 그리고 앞으로 우리에게 어떤 의미를 가지는지**를 탐구합니다.

좀 더 구체적으로 설명하자면, 역사는 다음과 같은 특징을 가집니다.

*   **과거에 대한 탐구:** 인류의 시작부터 현재까지 발생했던 모든 사건, 인물, 문화, 사회, 정치, 경제 등 다양한 측면을 다룹니다.
*   **기록과 해석:** 과거의 사실들은 주로 기록(문서, 유물, 구전 등)을 통해 전해지며, 역사가들은 이러한 기록들을 바탕으로 사실을 재구성하고 의미를 부여합니다. 이 과정에서 다양한 관점과 해석이 존재할 수 있습니다.
*   **인과 관계와 맥락:** 사건들이 왜 일어났는지, 다른 사건들과 어떤 관계가 있는지, 당시의 사회적, 문화적 맥락은 어떠했는지 등을 파악하는 데 중점을 둡니다.
*   **변화와 연속성:** 과거의 사회가 어떻게 변화해왔는지, 그리고 현재와 어떤 연속성을 가지고 있는지 보여줍니다.
*   **성찰과 교훈:** 과거의 경험을 통해 배우고, 현재의 문제에 대한 통찰력을 얻으며, 미래를 더 나은 방향으로 이끌어갈 수 있는 교훈을 얻고자 합니다.

**왜 역사를 공부할까요?**

*   **과거를 이해함으로써 현재를 더 잘 이해할 수 있습니다.** 현재 우리가 살고 있는 세상은 과거의 사건들과 결정의 결과물이기 때문입니다.
*   **실패로부터 배우고 반복을 피할 수 있습니다.** 과거의 잘못된 선택이나 실패 사례를 통해 미래의 오류를 줄일 수 있습니다.
*   **다양한 문화와 관점을 이해하는 데 도움이 됩니다.** 과거의 다양한 사회와 문화를 접하면서 폭넓은 시야를 가질 수 있습니다.
*   **정체성을 형성하는 데 기여합니다.** 자신이 속한 공동체나 국가의 역사를 아는 것은 정체성을 확립하는 데 중요한 역할을 합니다.
*   **비판적 사고 능력을 키울 수 있습니다.** 역사 자료를 분석하고 여러 해석을 비교하는 과정에서 비판적인 사고 능력이 향상됩니다.

간단히 말해, 역사는 **우리가 어디에서 왔고, 어떻게 여기까지 왔으며, 앞으로 어디로 나아가야 하는지를 알려주는 중요한 지침**이라고 할 수 있습니다.
--------------------------------
```

```python
# 추천 리스트 프롬프트 질문하기

# --- 첫 번째 비추천 리스트 프롬프트 호출 (영어) ---
# prompt_list_en_rec
try:
    # 리스트에 있는 프롬프트를 반복문으로 하나씩 호출
    for prompt in prompt_list_en_rec:
        print(f"\n--- 요청된 프롬프트 ---")
        print(f"입력: {prompt}")

        response = client.models.generate_content(
```

```
            model='gemini-2.5-flash-lite',                          # 모델
호출
            contents=prompt                                         # 반복
문으로 가져온 프롬프트 하나를 입력값으로 사용
        )

        # 응답 텍스트 출력
        print("\n--- 모델 응답 텍스트 ---")
        print(response.text)
        print("-" * 30)

    # ---------- 디버깅용 출력하지 않기 -------------
    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    # print("\n--- 모델 응답 전체 JSON ---")
    # print(response.model_dump_json(
    #     exclude_none=True, indent=4))

except Exception as e:
    print(f"\n 모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력

```
--- 요청된 프롬프트 ---
입력: List 5 ways the ocean supports human life

--- 모델 응답 텍스트 ---
The ocean is absolutely vital to human life and supports us in a
multitude of ways. Here are five key examples:

1.  **Oxygen Production:** This is perhaps the most fundamental
way the ocean supports us. Phytoplankton, tiny marine plants,
perform photosynthesis. Through this process, they convert carbon
dioxide and sunlight into energy, releasing oxygen as a
byproduct. It's estimated that the ocean produces **50-80% of the
world's oxygen**, making it the primary producer of the air we
breathe.

2.  **Food Source:** The ocean is a vast and crucial source of
food for billions of people worldwide. It provides a rich supply
of **fish, shellfish, and other seafood**, which are important
sources of protein, essential fatty acids (like omega-3s),
vitamins, and minerals. This food security is especially critical
```

```
for coastal communities and many developing nations.

3.  **Climate Regulation:** The ocean acts as a massive
**thermostat** for our planet. It absorbs a significant amount of
the heat trapped by greenhouse gases, preventing more extreme
temperature fluctuations on land. It also plays a crucial role in
the global water cycle, driving weather patterns and distributing
heat around the globe through ocean currents. Without this
regulation, our climate would be far more volatile and less
habitable.

4.  **Economic Activity and Livelihoods:** The ocean underpins a
vast array of economic activities that support human livelihoods.
This includes **commercial fishing, aquaculture, tourism,
shipping and transportation, offshore energy production (oil,
gas, and increasingly, renewable energy like wind and tidal), and
the extraction of minerals and resources.** These industries
provide jobs, income, and goods for societies globally.

5.  **Biodiversity and Medicinal Discoveries:** The ocean is home
to an incredible diversity of life, much of which remains
undiscovered. This **biodiversity** is not just intrinsically
valuable; it's also a potential treasure trove for scientific and
medical advancements. Many marine organisms produce unique
chemical compounds that are being studied for their potential use
in **new medicines, pharmaceuticals, and industrial
applications**, including treatments for cancer, pain relief, and
infectious diseases.
------------------------------

--- 요청된 프롬프트 ---
입력: Compare machine learning and deep learning with examples

--- 모델 응답 텍스트 ---
Let's break down the comparison between **Machine Learning (ML)**
and **Deep Learning (DL)**. Think of Deep Learning as a *subset*
of Machine Learning, but one that has revolutionized many areas
due to its power and capabilities.

## Machine Learning (ML)

**What it is:** Machine Learning is a broad field of artificial
intelligence that enables systems to **learn from data and make
predictions or decisions without being explicitly programmed**.
It's about building algorithms that can identify patterns, make
inferences, and improve their performance over time as they are
exposed to more data.

**Key Characteristics:**
```

```
*   **Feature Engineering:** A crucial step in traditional ML involves **manual feature engineering**. This means domain experts or data scientists carefully select, transform, and create relevant features (attributes or characteristics) from the raw data that the algorithm can use to learn. The quality of these features significantly impacts the model's performance.
*   **Algorithms:** Relies on a variety of algorithms like:
    *   Linear Regression
    *   Logistic Regression
    *   Support Vector Machines (SVMs)
    *   Decision Trees
    *   Random Forests
    *   K-Means Clustering
    *   Naive Bayes
*   **Data Requirements:** Can often perform well with **smaller to moderate amounts of data**.
*   **Computational Power:** Generally requires **less computational power** compared to deep learning.
*   **Interpretability:** Many traditional ML models are **more interpretable**, meaning you can understand *why* a model made a particular prediction.

**When to Use ML:**

*   When you have well-defined features or can easily extract them.
*   When interpretability is a high priority.
*   When dealing with smaller datasets.
*   When computational resources are limited.
*   For tasks that don't require understanding complex hierarchical representations.

**Examples of Machine Learning:**

1.  **Spam Email Detection:**
    *   **Features:** Number of exclamation marks, presence of certain keywords ("free," "viagra," "urgent"), sender's domain reputation, email length.
    *   **Algorithm:** Logistic Regression or Support Vector Machine.
    *   **Process:** A human might identify that emails with many exclamation marks and the word "free" are likely spam. This information is fed to the algorithm as features. The algorithm learns the patterns associated with spam based on these features.

2.  **Customer Churn Prediction:**
    *   **Features:** Customer demographics (age, location), purchase history (frequency, recency, monetary value), usage patterns (login frequency, support calls), contract type.
    *   **Algorithm:** Decision Tree or Random Forest.
```

```
    *   **Process:** Businesses analyze customer data and extract features like how often a customer uses a service or how many support tickets they've opened. The ML model learns which combinations of these features indicate a higher likelihood of a customer leaving.

3.  **House Price Prediction:**
    *   **Features:** Number of bedrooms, square footage, location (zip code), proximity to amenities, age of the house.
    *   **Algorithm:** Linear Regression or Gradient Boosting Machines.
    *   **Process:** Real estate agents or data scientists identify key attributes of a house that influence its price. These are then fed into a model to predict the selling price of new houses.

4.  **Recommendation Systems (Basic):**
    *   **Features:** User's past ratings of movies, genres they've liked, actors they prefer.
    *   **Algorithm:** Collaborative Filtering (e.g., using matrix factorization).
    *   **Process:** Based on what similar users have liked, the system recommends movies. Features are derived from user-item interaction data.

---

## Deep Learning (DL)

**What it is:** Deep Learning is a **subfield of Machine Learning** that uses **artificial neural networks with multiple layers (deep neural networks)** to learn representations of data. Instead of relying on manual feature engineering, deep learning models automatically learn hierarchical features from the raw data itself.

**Key Characteristics:**

*   **Automatic Feature Learning (Representation Learning):** This is the hallmark of deep learning. The multiple layers in a neural network learn increasingly complex representations of the input data. The early layers might learn simple features (like edges or corners in an image), while later layers combine these to learn more abstract features (like shapes, objects, or even entire scenes).
*   **Deep Neural Networks:** Consist of an input layer, multiple hidden layers, and an output layer. The "depth" refers to the number of hidden layers.
*   **Algorithms/Architectures:** Employs specialized neural network architectures:
```

```
    *   **Convolutional Neural Networks (CNNs):** Excellent for
image and video processing.
    *   **Recurrent Neural Networks (RNNs) / Long Short-Term
Memory (LSTM) / Gated Recurrent Units (GRUs):** Ideal for
sequential data like text and time series.
    *   **Transformers:** Revolutionized Natural Language
Processing (NLP) and are now used in other domains.
    *   **Generative Adversarial Networks (GANs):** For
generating new data that resembles the training data.
*   **Data Requirements:** Typically requires **very large
amounts of data** to train effectively and avoid overfitting.
*   **Computational Power:** Demands **significant computational
power**, often requiring GPUs (Graphics Processing Units) or TPUs
(Tensor Processing Units) for efficient training.
*   **Interpretability:** Often considered a **"black box"**
because it can be difficult to understand exactly *why* a deep
learning model makes a specific decision due to the complexity of
its learned representations.

**When to Use DL:**

*   When dealing with unstructured data like images, audio, or
natural language.
*   When complex patterns need to be discovered automatically.
*   When you have a very large dataset.
*   When high accuracy is paramount, and computational resources
are available.
*   For tasks involving perceptual understanding (seeing,
hearing, reading).

**Examples of Deep Learning:**

1.  **Image Recognition/Classification (e.g., identifying cats in
photos):**
    *   **Raw Data:** Pixels of an image.
    *   **Architecture:** Convolutional Neural Network (CNN).
    *   **Process:** A CNN automatically learns to detect edges,
then shapes like eyes and ears, and eventually combines these to
recognize a cat. No manual feature extraction like "number of
whiskers" is needed. The network learns these features from the
pixel data.

2.  **Natural Language Processing (e.g., sentiment analysis,
machine translation):**
    *   **Raw Data:** Raw text (sentences, paragraphs).
    *   **Architecture:** Recurrent Neural Network (RNN), LSTM,
or Transformer.
    *   **Process:** For sentiment analysis, a Transformer can
learn the context and nuances of words in a sentence to determine
if the sentiment is positive, negative, or neutral. For
```

```
translation, it learns the mapping between words and phrases in
different languages.

3.  **Speech Recognition (e.g., virtual assistants like Siri,
Alexa):**
    *   **Raw Data:** Audio waveforms.
    *   **Architecture:** RNNs or CNNs.
    *   **Process:** Deep learning models process the raw audio
signals, learning to distinguish phonemes, words, and then
complete sentences, translating spoken language into text.

4.  **Autonomous Driving:**
    *   **Raw Data:** Camera feeds, LiDAR data, radar data.
    *   **Architecture:** Combinations of CNNs, RNNs, and other
networks.
    *   **Process:** Deep learning models process sensor data in
real-time to identify pedestrians, other vehicles, traffic signs,
road boundaries, and predict their movements, enabling the car to
navigate safely.

5.  **Generative Art and Deepfakes:**
    *   **Raw Data:** Existing images or videos.
    *   **Architecture:** Generative Adversarial Networks (GANs).
    *   **Process:** GANs learn the underlying distribution of
the data and can generate entirely new, photorealistic images or
videos that mimic the style of the training data.

---

## Key Differences Summarized

| Feature             | Machine Learning (Traditional)
| Deep Learning                                                 |
| :-----------------
| :-------------------------------------------------
| :------------------------------------------------------------ |
| **Core Concept**    | Learning from data with explicit feature
selection | Learning hierarchical representations automatically
|
| **Feature Eng.**    | Manual, human-driven
| Automatic, learned by the model                               |
| **Data Needs**      | Smaller to moderate
| Very large                                                    |
| **Algorithms**      | SVM, Decision Trees, Linear Regression,
etc.      | Deep Neural Networks (CNNs, RNNs, Transformers, etc.)
|
| **Comp. Power**     | Moderate
| High (often requires GPUs/TPUs)                               |
| **Interpretability**| Higher
| Lower ("black box")                                           |
```

```
    | **Performance**     | Good with structured data and limited
    features    | Excels with unstructured data and complex patterns
    |
    | **Time to Develop** | Often faster for simpler tasks
    | Can be longer due to complex model building and training |

    In essence, Deep Learning automates the most tedious and crucial
    part of traditional Machine Learning — feature engineering. This
    automation, coupled with the power of deep neural networks,
    allows DL to tackle much more complex problems that were
    previously intractable. However, this power comes at the cost of
    needing more data and computational resources, and often
    sacrificing interpretability.
    ------------------------------

    --- 요청된 프롬프트 ---
    입력: Summarize the causes of World War I in 3 bullet points

    --- 모델 응답 텍스트 ---
    Here are three key causes of World War I:

    *   **The alliance system:** A complex web of interlocking
    treaties divided Europe into two main camps (the Triple Entente
    and the Central Powers), meaning that a conflict between two
    nations could quickly draw in many others.
    *   **Militarism and the arms race:** Major European powers
    engaged in a significant build-up of their armies and navies,
    fostering a climate of suspicion and readiness for war, and
    making military solutions seem more appealing.
    *   **Imperialism and nationalism:** Competition for colonies and
    resources, coupled with intense national pride and the desire for
    self-determination (especially in the Balkans), created
    significant tensions and rivalries between European powers.
    ------------------------------
```

```python
# 추천 리스트 프롬프트 질문하기

# --- 두 번째 비추천 리스트 프롬프트 호출 (한국어) ---
# prompt_list_ko_rec
try:
    # 리스트에 있는 프롬프트를 반복문으로 하나씩 호출
    for prompt in prompt_list_ko_rec:
        print(f"\n--- 요청된 프롬프트 ---")
        print(f"입력: {prompt}")

        response = client.models.generate_content(
            model='gemini-2.5-flash-lite',                    # 모델
호출
            contents=prompt                                   # 반복
```

```
문으로 가져온 프롬프트 하나를 입력값으로 사용
        )

        # 응답 텍스트 출력
        print("\n--- 모델 응답 텍스트 ---")
        print(response.text)
        print("-" * 30)

    # ---------- 디버깅용 출력하지 않기 -------------
    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    # print("\n--- 모델 응답 전체 JSON ---")
    # print(response.model_dump_json(
    #     exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력

```
--- 요청된 프롬프트 ---
입력: 바다가 인간의 삶을 지탱하는 5 가지 방법을 나열해 주세요.

--- 모델 응답 텍스트 ---
바다는 인간의 삶을 지탱하는 5 가지 주요 방법은 다음과 같습니다.

1.  **식량 공급원:** 바다는 전 세계 인구의 상당 부분을 위한 주요 단백질 공급원입니다. 어업은 수많은 사람들에게 생계를 제공하며, 해양 생물 다양성은 다양한 종류의 물고기, 조개류, 해초 등을 제공합니다.

2.  **기후 조절:** 바다는 지구의 기후를 조절하는 데 매우 중요한 역할을 합니다. 거대한 양의 열을 흡수하고 방출하여 지구의 온도를 안정시키는 데 기여하며, 대기의 이산화탄소를 흡수하여 기후 변화를 완화하는 데도 도움을 줍니다. 또한, 해류는 전 세계적으로 열을 분배하여 다양한 지역의 기후에 영향을 미칩니다.

3.  **산소 생산:** 바다의 식물성 플랑크톤은 지구 대기 중 산소의 약 50~70%를 생산합니다. 이는 우리가 호흡하는 공기의 상당 부분이 바다로부터 온다는 것을 의미하며, 지구 생태계 유지에 필수적인 요소입니다.

4.  **교통 및 무역 경로:** 바다는 인류 문명과 경제 발전에 있어 가장 중요한 교통 및 무역 경로를 제공합니다. 전 세계 상품의 대부분이 해상 운송을 통해 이동하며, 이는 글로벌 경제를 연결하고 문화 교류를 촉진하는 데 핵심적인 역할을 합니다.
```

5. **자원 및 영감의 원천:** 바다는 석유, 천연가스, 광물 등 귀중한 천연자원의 보고입니다. 또한, 바다는 아름다운 경관, 다양한 해양 생물, 그리고 인간에게 휴식, 레크리에이션, 영감을 주는 공간을 제공합니다. 해양 관광 산업 또한 중요한 경제 활동이며, 해양 연구는 새로운 기술과 지식을 발전시키는 원동력이 됩니다.

--- 요청된 프롬프트 ---
입력: 머신러닝과 딥러닝을 예시와 함께 비교해 주세요.

--- 모델 응답 텍스트 ---
머신러닝 vs 딥러닝: 예시를 통해 비교하기

머신러닝과 딥러닝은 모두 컴퓨터가 데이터를 학습하여 특정 작업을 수행하도록 만드는 기술입니다. 하지만 그 방식과 복잡성에서 차이가 있습니다. 쉽게 이해할 수 있도록 각각의 개념과 예시를 통해 비교해 드리겠습니다.

---

1. 머신러닝 (Machine Learning)

핵심: **데이터에서 패턴을 학습**하여 예측하거나 의사결정을 내립니다. 사람이 미리 **특징(feature)**을 정의하고, 머신러닝 알고리즘은 이 특징들을 바탕으로 데이터를 학습합니다.

어떻게 작동하나요?

1. **데이터 준비:** 학습시킬 데이터를 모읍니다.
2. **특징 추출 (Feature Engineering):** 문제 해결에 중요하다고 판단되는 데이터의 특징들을 사람이 직접 정의하고 추출합니다.
3. **모델 선택:** 데이터를 학습시킬 머신러닝 알고리즘(예: 선형 회귀, 결정 트리, 서포트 벡터 머신 등)을 선택합니다.
4. **학습:** 추출된 특징들을 이용하여 모델을 학습시킵니다.
5. **예측/분류:** 학습된 모델을 사용하여 새로운 데이터에 대한 예측이나 분류를 수행합니다.

예시:

* **스팸 메일 분류:**
    * **데이터:** 수많은 이메일 (스팸 메일, 정상 메일)
    * **특징 추출 (사람이 정의):**
        * 이메일에 특정 단어(예: "돈", "광고", "당첨", "무료")가 포함되는 빈도
        * 보낸 사람의 주소가 이상한지 여부
        * 제목에 느낌표가 많이 사용되었는지 여부
        * 이메일의 길이
    * **학습:** 위 특징들을 사용하여 스팸 메일인지 아닌지를 구분하는 모델을 학습시킵니다.
    * **예측:** 새로운 이메일이 왔을 때, 해당 이메일의 특징들을 분석하여 스팸인지 정상 메일인지 분류합니다.

*   **부동산 가격 예측:**
    *   **데이터:** 과거 거래된 부동산 정보 (크기, 방 개수, 위치, 건축 연도, 거래 가격 등)
    *   **특징 추출 (사람이 정의):**
        *   집의 면적 (제곱미터)
        *   방의 개수
        *   동네의 평균 소득 수준
        *   교통 편의성 (지하철역과의 거리)
        *   집의 건축 연도
    *   **학습:** 위 특징들을 사용하여 부동산의 거래 가격을 예측하는 모델을 학습시킵니다.
    *   **예측:** 새로운 부동산의 특징 정보를 입력하면, 해당 부동산의 예상 가격을 예측합니다.

---

2. 딥러닝 (Deep Learning)

핵심: 머신러닝의 한 분야로, **인공 신경망(Artificial Neural Network)**을 여러 층(layer)으로 깊게 쌓아 복잡한 패턴을 학습합니다. **특징 추출 과정을 자동화**하여, 데이터 자체에서 필요한 특징을 스스로 찾아냅니다.

어떻게 작동하나요?

1. **데이터 준비:** 학습시킬 데이터를 모읍니다.
2. **신경망 설계:** 여러 개의 은닉층(hidden layer)으로 구성된 신경망 구조를 설계합니다.
3. **자동 특징 학습:** 입력된 데이터는 여러 층을 거치면서 자동으로 특징들이 계층적으로 학습됩니다. (예: 이미지의 경우, 첫 번째 층은 기본적인 엣지나 색깔을 인식하고, 다음 층은 더 복잡한 모양이나 질감을 인식하는 식)
4. **학습:** 신경망을 통해 데이터를 학습시키고, 예측값과 실제값의 오차를 줄이는 방향으로 가중치를 조정합니다.
5. **예측/분류:** 학습된 신경망을 사용하여 새로운 데이터에 대한 예측이나 분류를 수행합니다.

예시:

*   **이미지 인식 (고양이 사진 분류):**
    *   **데이터:** 수많은 고양이 사진과 고양이가 아닌 사진
    *   **자동 특징 학습:**
        *   **첫 번째 층:** 사진에서 기본적인 선, 모서리, 색깔 등의 저수준 특징 학습.
        *   **중간 층:** 고양이의 귀, 눈, 코, 입과 같은 부분적인 특징 학습.
        *   **마지막 층:** 학습된 부분 특징들을 조합하여 전체적으로 고양이의 형상을 인식하고 분류.
    *   **학습:** 이미지를 입력받아 "고양이"인지 "고양이가 아닌지"를 분류하는 신경망 학습.
    *   **예측:** 새로운 사진을 입력하면, 신경망이 사진 속 특징들을 자동으로 분석하여 고양이인지 아닌지 판단합니다. (사람이 직접 '귀 모양', '수염' 등을 특징으로 지정

```
할 필요가 없습니다.)

*   **자연어 처리 (텍스트 번역):**
    *   **데이터:** 한국어 문장과 해당 영어 번역 문장 쌍
    *   **자동 특징 학습:**
        *   문장에서 단어의 의미, 문맥, 문법 구조 등을 파악하는 복잡한 특징들을 학습합니다.
        *   단어 간의 관계, 문장 전체의 의미 흐름 등을 이해합니다.
    *   **학습:** 한국어 문장을 입력받아 가장 적절한 영어 문장으로 번역하는 신경망 학습.
    *   **예측:** "안녕하세요"를 입력하면, 신경망은 학습된 내용을 바탕으로 "Hello"라고 번역해 줍니다.

---

### 머신러닝 vs 딥러닝 비교 요약

| 특징      | 머신러닝 (Machine Learning)
| 딥러닝 (Deep Learning)
|
| :---------------
| :-----------------------------------------------------------
| :-----------------------------------------------------------
--------------- |
| **특징 추출**     | **사람이 직접 특징(feature)을 정의하고 추출**해야 함.
| **자동으로 특징을 학습**함.
|
| **복잡성**        | 상대적으로 **단순한 문제**에 효과적.
| **매우 복잡하고 비정형적인 데이터** (이미지, 음성, 텍스트) 처리에 강점.
|
| **데이터 양**     | **적은 양의 데이터**로도 학습 가능.
| **방대한 양의 데이터**를 필요로 함. (데이터가 많을수록 성능 향상)
|
| **알고리즘**      | 선형 회귀, 로지스틱 회귀, 결정 트리, SVM, K-NN 등
| 심층 신경망 (Deep Neural Network - DNN), CNN, RNN, LSTM,
Transformer 등    |
| **계산량/시간**   | 상대적으로 **적은 계산량과 학습 시간**.
| **많은 계산량과 학습 시간**을 필요로 함. (GPU 등 고성능 하드웨어 중요)
|
| **해석 가능성**   | 모델의 **결정 과정을 비교적 쉽게 이해**할 수 있음.
| 모델 내부의 복잡성으로 인해 **해석이 어려울 수 있음** (블랙박스)
|
| **전형적인 활용** | 스팸 필터, 추천 시스템, 예측 모델 등           | 이미지 인식, 음성 인식, 자연어 처리 (번역, 챗봇), 자율 주행 등        |

**결론적으로,**

*   **머신러닝**은 데이터의 특징을 사람이 잘 알고 있고, 데이터 양이 적을 때 유용합니다.
```

```
*   **딥러닝**은 복잡한 패턴을 가진 비정형 데이터를 다루고, 특징 추출을 자동화하고 싶을 때, 그리고 충분한 데이터와 계산 자원이 있을 때 강력한 성능을 발휘합니다.

딥러닝은 머신러닝의 한 종류이며, 훨씬 더 복잡하고 성능이 뛰어난 알고리즘들을 포함한다고 생각하시면 됩니다.
------------------------------

--- 요청된 프롬프트 ---
입력: 제 1 차 세계대전의 원인을 세 가지로 요약해 주세요.

--- 모델 응답 텍스트 ---
제 1 차 세계 대전의 복잡한 원인들을 세 가지로 요약하면 다음과 같습니다.

1.  **복잡한 동맹 체제:** 유럽 열강들은 서로를 견제하고 안보를 강화하기 위해 복잡하게 얽힌 군사 동맹을 맺었습니다. 대표적으로 삼국 동맹(독일, 오스트리아-헝가리, 이탈리아)과 삼국 협상(영국, 프랑스, 러시아)이 있었습니다. 이러한 동맹 체제는 한 국가가 분쟁에 휘말리면 연쇄적으로 다른 국가들도 전쟁에 참여하게 만드는 '도미노 효과'를 유발했습니다.

2.  **군비 경쟁과 제국주의:** 유럽 국가들은 식민지 확보와 영향력 확대를 위해 치열한 경쟁을 벌였습니다. 이는 새로운 무기 개발 및 군대 증강으로 이어져 상호 간의 불신과 긴장을 고조시켰습니다. 특히 독일의 급성장과 이에 대한 기존 강대국들의 견제 심리가 군비 경쟁을 더욱 부추겼습니다.

3.  **민족주의와 민족 자결주의:** 발칸반도를 중심으로 강한 민족주의가 대두되었고, 여러 민족들이 독립 또는 통합을 주장하며 지역적 긴장을 심화시켰습니다. 특히 오스트리아-헝가리 제국의 슬라브 민족들이 독립을 원하며 러시아의 지원을 받으려 하자, 오스트리아-헝가리는 이를 위협으로 느껴 민족주의 운동을 억압하려 했습니다. 사라예보 사건(오스트리아-헝가리 황태자 암살)은 이러한 민족주의적 갈등이 폭발하는 기폭제가 되었습니다.

이 세 가지 요인이 복합적으로 작용하여 제 1 차 세계 대전이라는 거대한 비극을 초래했습니다.
------------------------------
```

---

3) Ask one task at a time (한 번에 하나의 작업만 요청하기)

a. 구글 클라우드 실습

- **비추천** - 아래 프롬프트는 두 가지 질문을 한꺼번에 포함하고 있어, 따로따로 묻는 것이 좋습니다.

- 실습 코드

```
    prompt = "What's the best method of boiling water and why is the sky blue?"
```

```
    response = client.models.generate_content(model=MODEL_ID,
contents=prompt)
    display(Markdown(response.text))
```

- 대답

```
Let's break down both of your questions!

---

### What's the Best Method of Boiling Water?

"Best" can depend on your priorities: speed, energy efficiency,
convenience, or capacity.

1.  **For Speed and Energy Efficiency (Typical Household Use):
The Electric Kettle**
    *   **Why it's often best:** Electric kettles are designed to
heat water directly and efficiently. Their heating element is
submerged in the water, transferring heat with minimal loss to
the surrounding air. They also often have powerful elements
(1500-3000 watts) and a relatively small volume of water to heat.
They are generally faster and more energy-efficient for
quantities up to 1.7-2 liters compared to a stovetop.
    *   **Pros:** Very fast, energy-efficient for small to medium
volumes, convenient (auto shut-off), safer.
    *   **Cons:** Requires an electrical outlet, limited
capacity.

2.  **For Large Volumes or Versatility: The Induction Stovetop**
    *   **Why it's a close second (or even first for specific
needs):** Induction cooktops use electromagnetic fields to
directly heat the pot itself, making them incredibly efficient
and fast, similar to (or sometimes even faster than) electric
kettles, especially for larger quantities of water.
    *   **Pros:** Very fast and energy-efficient, handles any
volume (depending on pot size), no specialized appliance needed
if you already have an induction stove.
    *   **Cons:** Requires induction-compatible cookware,
requires a specific type of stove.

3.  **Other Methods (and why they might not be "best"):**
    *   **Gas/Electric Coil Stovetop:** Can be slower and less
energy-efficient than electric kettles or induction because a
significant amount of heat is lost to the air around the pot
before it reaches the water. Good for any pot size, though.
    *   **Microwave:** Generally **not recommended** for boiling
water. It heats unevenly and slowly, and there's a risk of
```

"superheating" where the water gets hotter than its boiling point without bubbling, leading to explosive boiling when disturbed (e.g., adding a tea bag). Only use for very small amounts if absolutely necessary.
    *   **Immersion Heater:** Portable and useful in a pinch, but often slower and can have safety concerns if not used properly (e.g., leaving unattended, touching the element).

Conclusion: For most people and most uses, an **electric kettle** is the most convenient, fastest, and most energy-efficient way to boil water. If you have an induction stovetop and need to boil larger quantities, that's an excellent choice too.

Why is the Sky Blue?

The blue color of the sky is due to a phenomenon called **Rayleigh Scattering**. Here's how it works:

4. **Sunlight is White Light:** The light from the sun appears white to us, but it's actually made up of all the colors of the rainbow (a spectrum of wavelengths), from shorter-wavelength violet/blue to longer-wavelength red/orange.

5. **Earth's Atmosphere:** Our atmosphere is composed of tiny gas molecules, primarily nitrogen (N_2) and oxygen (O_2), along with small amounts of other gases and particles.

6. **Light Scattering:** When sunlight enters the atmosphere, it interacts with these tiny molecules. Instead of passing straight through, some of the light is absorbed and then re-emitted in different directions. This process is called scattering.

7. **Wavelength Dependence (The Key):** Rayleigh scattering states that **shorter wavelengths of light (like blue and violet) are scattered much more efficiently and intensely than longer wavelengths (like red and yellow).** Imagine it like a wave hitting a tiny obstacle: short, choppy waves hit the obstacle more often and get disrupted more than long, sweeping waves. Specifically, blue light is scattered about 10 times more effectively than red light.

8. **Why We See Blue:** As sunlight travels through the atmosphere, the blue and violet components are scattered in all directions. A significant amount of this scattered blue light reaches our eyes, no matter where we look in the sky (as long as we're not looking directly at the sun). While violet light is scattered even more than blue, our eyes are more sensitive to blue light, and some of the violet light is absorbed in the upper

```
atmosphere, so the sky appears blue.

**What happens at Sunrise/Sunset?**
When the sun is low on the horizon (sunrise or sunset), its light has to travel through a much greater amount of the Earth's atmosphere to reach our eyes. During this longer journey, most of the blue and violet light gets scattered away from our line of sight. This leaves the longer-wavelength colors (red, orange, yellow) to pass through more directly, which is why we see those vibrant warm colors during sunrise and sunset.
```

- **추천** - 아래 프롬프트는 한 번에 하나의 작업만 요청하고 있습니다.

- 실습 코드

```
    prompt = "What's the best method of boiling water?"

    response = client.models.generate_content(model=MODEL_ID,
contents=prompt)
    display(Markdown(response.text))
```

- 대답 ```

  There isn't a single "best" method for boiling water, as it depends on your priorities: **speed, energy efficiency, convenience, amount of water needed, and available equipment.**

  Here's a breakdown of common methods and their pros and cons:

  ### 1. Electric Kettle (Most Common & Often "Best" Overall for Beverages)

  - **Pros:**
    - **Very Fast:** Especially for 1-2 liters.
    - **Energy Efficient:** Heats only the water directly, often has automatic shut-off.
    - **Convenient:** Easy to use, portable (within reach of an outlet), no stovetop needed.
    - **Safe:** Auto-shutoff prevents boiling dry.
  - **Cons:**
    - Limited capacity.
    - Requires counter space and an electrical outlet.
  - **Best for:** Tea, coffee, instant noodles, or any time you need 1-2 liters of boiling water quickly and efficiently.

2. Stovetop (Pot on a Burner)

- **Pros:**
  - **Versatile:** Can boil any quantity of water in any suitable pot.
  - **No Special Appliance:** Just a pot and a stove.
  - **Works during power outages (gas/propane).**
- **Cons:**
  - **Less Efficient (generally):** Heat escapes around the pot.
  - **Requires Monitoring:** You have to turn it off yourself.
  - **Slower:** Especially electric coil stovetops.

  Types of Stovetops:
  - **Induction:**
    - **Pros:** Extremely fast, very energy-efficient (direct heat to the pot), precise control.
    - **Cons:** Requires specific magnetic cookware, stovetop itself can be expensive.
    - **Best for:** Speed and efficiency on a stovetop, if you have the right equipment.
  - **Gas:**
    - **Pros:** Fast, good heat control, instant heat on/off.
    - **Cons:** Less efficient than induction (some heat loss), requires gas connection.
    - **Best for:** General cooking, large volumes, when electric kettles are too small.
  - **Electric Coil/Radiant:**
    - **Pros:** Common, works with any pot.
    - **Cons:** Slowest of the stovetop types, less efficient (element heats up then transfers heat).
    - **Best for:** General cooking if it's your only option.

3. Instant Hot Water Tap (e.g., Quooker, Zip)

- **Pros:**
  - **Instant:** Boiling water on demand, no waiting.
  - **Convenient:** Frees up counter space from a kettle.
- **Cons:**
  - **High Upfront Cost:** Expensive to purchase and install.
  - **Energy Consumption:** Uses energy continuously to keep water hot in a tank (though modern ones are efficient).
- **Best for:** Ultimate convenience if budget allows and you frequently need small amounts of boiling water.

4. Microwave

- **Pros:**
  - **Convenient for single cups:** No need for a separate appliance if you already have a microwave.

- **Cons:**
  - **Inefficient for boiling:** Takes a long time for more than a single cup.
  - **Safety Risk (Superheating):** Water can heat beyond its boiling point without bubbling, causing an explosive boil when disturbed (e.g., adding a tea bag). Always stir before removing.
  - Doesn't produce a "rolling boil" typically needed for certain cooking tasks.
- **Best for:** Very small quantities (e.g., one cup for instant coffee) in a pinch, but not recommended for general boiling due to safety and efficiency.

5. Immersion Heater

- **Pros:**
  - **Portable:** Great for travel or camping.
  - Heats water quickly in a suitable container.
- **Cons:**
  - **Safety Risk:** Can be dangerous if not used properly (electric shock, fire). Not recommended for everyday home use.
  - Requires a non-conductive, heat-safe container.
- **Best for:** Travel, emergencies, or specific niche uses where a kettle or stove isn't available.

Tips for Boiling Water More Efficiently (Regardless of Method):

- **Use a Lid:** This traps heat, reduces evaporation, and significantly speeds up boiling.
- **Only Boil What You Need:** Don't fill a kettle or pot with more water than necessary.
- **Start with Hot Tap Water (if safe):** If your tap water is hot and safe to drink/use, starting with warmer water will reduce boiling time. (Note: For drinking, some prefer cold water to avoid potential lead or other contaminants that might leach into hot water from pipes).
- **Descaling:** If using an electric kettle, regularly descale it (remove mineral buildup) for optimal efficiency.

Conclusion:

- **For most everyday household needs (tea, coffee, instant meals): The Electric Kettle is generally the "best" due to its speed, energy efficiency, and convenience.**
- **For larger quantities or general cooking: A stovetop pot (especially on induction or gas) is the best choice.**
- **For ultimate convenience (if budget allows): An instant hot water tap.**
- **Avoid the Microwave for boiling if possible due to efficiency and safety concerns.**

```
```

- b. 로컬
  - 구글 클라우드 실습과 같은 질문으로 해보기

구	프롬프트 내용	이유
비	- What's the best method of boiling water and why is the sky blue? - 물을 끓이는 가장 좋은 방법은 무엇이며, 하늘은 왜 파란가요?	로 관련 없는 두 가지 질문이 한 문장에 섞여 있어서, 응답이 모호해지거나 비효율적으로 길어질 수 있음.
추	- What's the best method of boiling water? - 물을 끓이는 가장 좋은 방법은 무엇인가요?	하나의 명확한 질문만 담겨 있어, 집중도 높은 응답을 받을 수 있음.

- 비슷한 유형으로 질문해보기
  - ver_en

비추천	추천
What are the benefits of meditation and how do airplanes fly?	What are the benefits of meditation?
Summarize the history of the internet and write a poem about it.	Summarize the history of the internet.
Explain photosynthesis and recommend three houseplants.	Explain photosynthesis.

  - ver_ko

비추천	추천
명상은 어떤 이점이 있고, 비행기는 어떻게 나는가요?	명상의 이점은 무엇인가요?
인터넷의 역사를 요약하고, 그것에 대한 시를 써 주세요.	인터넷의 역사를 요약해 주세요.
광합성을 설명하고, 키우기 좋은 식물 세 가지를 추천해 주세요.	광합성에 대해 설명해 주세요.

```
# 3_1. 구글 클라우드 실습 질문과 같은 질문으로 입력해보기 ( 비추천)

prompt_en_3 = "What's the best method of boiling water and why is the sky blue?"
prompt_ko_3 = "물을 끓이는 가장 좋은 방법은 무엇이며, 하늘은 왜 파란가요?"

# --- 첫 번째 프롬프트 호출 ( 영어) ---
try:
    print("\n--- 첫 번째 요청 (영어) ---")
```

```
    response_en = client.models.generate_content(
        model='gemini-2.5-flash-lite',                                  #
사용할 모델 지정
        contents=prompt_en_3                                            #
영어 프롬프트 입력
    )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response_en.text)
    print("-" * 30)

    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    #print("\n--- 모델 응답 전체JSON ---")
    #print(response.model_dump_json(
    #    exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")

# --- 두 번째 프롬프트 호출 (한국어) ---
try:
    print("\n--- 두 번째 요청 (한국어) ---")
    response_ko = client.models.generate_content(
        model='gemini-2.5-flash-lite',
# 사용할 모델 지정
        contents=prompt_ko_3
# 한국어 프롬프트 입력
    )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_ko}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response_ko.text)
    print("-" * 30)

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

--- 첫 번째 요청 (영어) ---
입력 프롬프트: 제 1 차 세계대전의 원인을 세 가지로 요약해 주세요.

--- 모델 응답 텍스트 ---
Let's break down both of your fascinating questions!

What's the best method of boiling water?

The "best" method of boiling water depends entirely on your priorities and circumstances. However, for most everyday purposes, the **electric kettle** is generally considered the best and most efficient method.

Here's a breakdown of common methods and why the electric kettle often wins:

1. Electric Kettle:

*   **Pros:**
    *   **Speed:** Electric kettles are significantly faster than stovetop methods because they directly heat the water element and the heating coil is in direct contact with the water.
    *   **Energy Efficiency:** They are highly efficient as most of the electrical energy is converted directly into heat to warm the water. There's minimal heat loss to the surrounding air.
    *   **Automatic Shut-off:** Most modern kettles have an automatic shut-off feature when the water reaches boiling point, preventing overheating and saving energy.
    *   **Convenience:** Simply fill, plug in, and press a button. No need to monitor or turn off the stove.
    *   **Portability:** Small and easy to use in various settings (dorm rooms, offices, travel).
*   **Cons:**
    *   Requires an electrical outlet.
    *   Can be a tripping hazard if the cord isn't managed well.
    *   Minerals can build up over time, requiring descaling.

2. Stovetop (Pot/Saucepan):

*   **Pros:**
    *   **Versatility:** You can use any stovetop (gas, electric, induction) and any pot.
    *   **Control:** You have more direct control over the heat level if you need to simmer or maintain a specific temperature (though this isn't "boiling").
    *   **No Special Equipment:** If you don't have an electric kettle, this is your go-to.
*   **Cons:**
    *   **Slower:** Generally takes longer than an electric kettle, especially on older or less efficient stovetops.

    *   **Less Energy Efficient:** Significant heat is lost to the surrounding air from the pot and the stovetop itself.
    *   **Requires Monitoring:** You need to watch it to turn off the heat once it boils, or risk it boiling dry.
    *   **Can be Messy:** Splashing can occur, and you need to be careful when handling a hot pot.

3. Microwave:

*   **Pros:**
    *   **Speed (for small amounts):** For a single cup of water, a microwave can be very quick.
*   **Cons:**
    *   **Uneven Heating:** Microwaves heat water unevenly, leading to "superheating" where the water can be hotter than its boiling point but not visibly boiling. When disturbed, it can erupt violently.
    *   **Safety Hazard:** This superheating makes microwaves a potentially dangerous method for boiling water. Always stir or introduce a non-metallic object (like a wooden stirrer) to prevent this.
    *   **Not Ideal for Larger Quantities:** Not efficient or practical for boiling larger amounts of water.
    *   **Can't tell it's boiling:** It doesn't visually indicate boiling in the same way as other methods.

4. Other Methods (less common for everyday boiling):

*   **Instant Hot Water Dispensers:** Convenient for immediate hot water, but not for "boiling" in the traditional sense of bringing a pot to a rolling boil.
*   **Camp Stoves/Propane Burners:** Effective for outdoor cooking and boiling, but not typical for home use.

Conclusion for "Best":

For most people, the **electric kettle is the best method of boiling water due to its speed, energy efficiency, and convenience.**

Why is the sky blue?

The sky appears blue due to a phenomenon called **Rayleigh scattering**. Here's a breakdown:

1. **Sunlight is White Light:** Sunlight, when it reaches Earth's atmosphere, appears white to us. However, white light is actually a combination of all the colors of the visible spectrum (red, orange, yellow, green, blue, indigo, violet). Each of these colors has a different wavelength.

2. **Earth's Atmosphere:** Our atmosphere is composed of gases, primarily nitrogen (about 78%) and oxygen (about 21%), along with

smaller amounts of other gases and tiny particles.

3. **Scattering of Light:** As sunlight travels through the atmosphere, it encounters these gas molecules and tiny particles. When light waves hit these molecules, they are scattered in all directions.

4. **Wavelength Matters:** The key to the sky's blue color is that **shorter wavelengths of light are scattered more effectively than longer wavelengths.** Blue and violet light have the shortest wavelengths in the visible spectrum, while red and orange light have the longest.

5. **Rayleigh Scattering:** This specific type of scattering, which occurs when the scattering particles are much smaller than the wavelength of the light, is known as Rayleigh scattering. It's much more efficient at scattering shorter wavelengths.

6. **Dominance of Blue:** Because blue light is scattered in all directions by the gas molecules, it reaches our eyes from all parts of the sky. This is why the sky appears blue.

Why isn't the sky violet?

You might wonder why the sky isn't violet, since violet has an even shorter wavelength than blue and should scatter even more. There are a couple of reasons for this:

* **Our Eyes' Sensitivity:** Our eyes are more sensitive to blue light than to violet light.
* **Sunlight Spectrum:** The sun emits slightly less violet light than blue light.

So, while violet light is scattered, the combination of our eyes' sensitivity and the amount of blue light available makes us perceive the sky as blue.

What about sunsets and sunrises?

At sunrise and sunset, the sunlight has to travel through a much greater thickness of the atmosphere to reach our eyes. This means that most of the shorter, bluer wavelengths are scattered away, leaving the longer, redder and orange wavelengths to pass through more directly, giving us those beautiful red and orange hues.

--- 두 번째 요청 (한국어) ---
입력 프롬프트: 지구에 대해 말해 주세요

--- 모델 응답 텍스트 ---
물을 끓이는 가장 좋은 방법

물을 끓이는 "가장 좋은" 방법은 목적과 상황에 따라 달라집니다. 하지만 일반적으로 다음의 사항들을 고려하면 효율적이고 안전하게 물을 끓일 수 있습니다.

1. 적절한 용기 선택:

* **주전자:** 가장 일반적이고 효율적인 방법입니다. 전기주전자는 물을 빠르게 끓일 수 있으며, 가스레인지용 주전자도 좋습니다.
* **냄비:** 냄비로도 물을 끓일 수 있지만, 주전자보다는 열 전달 효율이 낮아 시간이 더 걸릴 수 있습니다.
* **IH 조리기구:** IH 코일이 내장된 조리기구는 바닥이 평평한 금속 용기를 사용해야 합니다.

2. 물의 양:

* **필요한 만큼만 끓이기:** 너무 많은 물을 끓이면 에너지가 낭비됩니다. 필요한 양만큼만 끓이는 것이 좋습니다.
* **최소/최대 표시 확인:** 주전자나 냄비에 표시된 최소 및 최대 물 높이를 지키는 것이 안전합니다.

3. 가열 방식:

* **전기주전자:** 가장 빠르고 편리합니다. 온도를 조절할 수 있는 제품도 있습니다.
* **가스레인지:** 불꽃의 세기를 조절하여 끓이는 속도를 제어할 수 있습니다.
* **인덕션 (IH) 조리기구:** 에너지 효율이 높고 빠릅니다.

4. 끓이는 속도:

* **강불:** 물을 빠르게 끓일 수 있지만, 에너지 소비가 많고 넘칠 위험이 있습니다.
* **중불:** 적절한 속도로 끓일 수 있으며, 에너지 효율도 좋습니다.
* **약불:** 천천히 끓이지만, 시간은 오래 걸립니다.

5. 추가 팁:

* **뚜껑 덮기:** 뚜껑을 덮으면 열 손실을 줄여 물이 더 빨리 끓고 에너지도 절약됩니다.
* **끓기 시작하면 불 줄이기:** 물이 끓기 시작하면 불꽃을 약간 줄여도 온도가 유지됩니다.
* **끓인 물 활용:** 끓인 물은 음료뿐만 아니라 요리, 소독 등 다양한 용도로 활용할 수 있습니다.

요약하자면,

* **가장 빠르고 편리하게 끓이고 싶다면:** 전기주전자를 사용하고 뚜껑을 덮으세요.
* **가스레인지 사용 시:** 중불로 끓이고 뚜껑을 덮으면 효율적입니다.
* **에너지 절약을 중요시한다면:** 필요한 만큼만 끓이고 뚜껑을 덮어 끓이세요.

하늘은 왜 파란가요?

하늘이 파랗게 보이는 이유는 **빛의 산란(Scattering) 현상** 때문입니다. 좀 더 자세히 설명해 드리겠습니다.

1. **햇빛은 다양한 색깔의 빛으로 이루어져 있습니다.** 우리가 흔히 무지개에서 보는 빨강, 주황, 노랑, 초록, 파랑, 남색, 보라색과 같은 다양한 파장의 빛이 섞여 있습니다.

2. **지구 대기의 구성:** 지구 대기에는 질소, 산소와 같은 기체 분자들과 먼지, 물방울 등의 작은 입자들이 떠다닙니다.

3. **레일리 산란 (Rayleigh Scattering):** 햇빛이 대기 중의 작은 기체 분자들과 부딪히면 빛은 사방으로 흩어지는데, 이를 산란이라고 합니다. 이때 **파장이 짧은 파란색 계열의 빛(보라색, 파란색)이 파장이 긴 빨간색 계열의 빛보다 훨씬 더 강하게 산란됩니다.** 이것을 **레일리 산란**이라고 합니다.

4. **우리 눈에 보이는 현상:** 태양에서 출발한 햇빛이 대기를 통과할 때, 파장이 짧은 파란색 빛은 대기 분자들에 의해 사방으로 흩어져 우리 눈에 도달합니다. 따라서 하늘을 바라보면 우리 눈에 가장 많이 들어오는 흩어진 빛인 파란색을 보게 되는 것입니다.

그렇다면 보라색은 왜 안 보일까요?

사실 보라색 빛도 파장이 짧아 파란색보다 더 많이 산란됩니다. 하지만 우리 눈은 파란색에 더 민감하게 반응하고, 태양에서 나오는 빛 자체가 파란색에 비해 보라색의 비율이 약간 더 낮기 때문에 하늘은 주로 파랗게 보입니다.

해 질 녘 하늘이 붉은 이유:

해 질 녘에는 태양빛이 더 두꺼운 대기층을 통과해야 합니다. 이 과정에서 파장이 짧은 파란색 빛은 이미 대부분 산란되어 흩어져 버립니다. 반면 파장이 긴 빨간색, 주황색 빛은 상대적으로 더 적게 산란되어 우리 눈에 도달하기 때문에 하늘이 붉게 보이는 것입니다.

요약하자면, **햇빛이 대기 중의 작은 입자들에 의해 산란될 때, 파장이 짧은 파란색 빛이 더 많이 흩어져 우리 눈에 들어오기 때문에 하늘은 파랗게 보이는 것**입니다.

- 셀 출력

```
--- 첫 번째 요청 (영어) ---
입력 프롬프트: 제 1 차 세계대전의 원인을 세 가지로 요약해 주세요.

--- 모델 응답 텍스트 ---
Let's break down both of your fascinating questions!

## What's the best method of boiling water?

The "best" method of boiling water depends entirely on your
priorities and circumstances. However, for most everyday
purposes, the **electric kettle** is generally considered the
best and most efficient method.

Here's a breakdown of common methods and why the electric kettle
often wins:

**1. Electric Kettle:**
```

```
*   **Pros:**
    *   **Speed:** Electric kettles are significantly faster than stovetop methods because they directly heat the water element and the heating coil is in direct contact with the water.
    *   **Energy Efficiency:** They are highly efficient as most of the electrical energy is converted directly into heat to warm the water. There's minimal heat loss to the surrounding air.
    *   **Automatic Shut-off:** Most modern kettles have an automatic shut-off feature when the water reaches boiling point, preventing overheating and saving energy.
    *   **Convenience:** Simply fill, plug in, and press a button. No need to monitor or turn off the stove.
    *   **Portability:** Small and easy to use in various settings (dorm rooms, offices, travel).
*   **Cons:**
    *   Requires an electrical outlet.
    *   Can be a tripping hazard if the cord isn't managed well.
    *   Minerals can build up over time, requiring descaling.

**2. Stovetop (Pot/Saucepan):**

*   **Pros:**
    *   **Versatility:** You can use any stovetop (gas, electric, induction) and any pot.
    *   **Control:** You have more direct control over the heat level if you need to simmer or maintain a specific temperature (though this isn't "boiling").
    *   **No Special Equipment:** If you don't have an electric kettle, this is your go-to.
*   **Cons:**
    *   **Slower:** Generally takes longer than an electric kettle, especially on older or less efficient stovetops.
    *   **Less Energy Efficient:** Significant heat is lost to the surrounding air from the pot and the stovetop itself.
    *   **Requires Monitoring:** You need to watch it to turn off the heat once it boils, or risk it boiling dry.
    *   **Can be Messy:** Splashing can occur, and you need to be careful when handling a hot pot.

**3. Microwave:**

*   **Pros:**
    *   **Speed (for small amounts):** For a single cup of water, a microwave can be very quick.
*   **Cons:**
    *   **Uneven Heating:** Microwaves heat water unevenly, leading to "superheating" where the water can be hotter than its boiling point but not visibly boiling. When disturbed, it can erupt violently.
    *   **Safety Hazard:** This superheating makes microwaves a
```

```
potentially dangerous method for boiling water. Always stir or
introduce a non-metallic object (like a wooden stirrer) to
prevent this.
    *   **Not Ideal for Larger Quantities:** Not efficient or
practical for boiling larger amounts of water.
    *   **Can't tell it's boiling:** It doesn't visually indicate
boiling in the same way as other methods.

**4. Other Methods (less common for everyday boiling):**

*   **Instant Hot Water Dispensers:** Convenient for immediate
hot water, but not for "boiling" in the traditional sense of
bringing a pot to a rolling boil.
*   **Camp Stoves/Propane Burners:** Effective for outdoor
cooking and boiling, but not typical for home use.

**Conclusion for "Best":**

For most people, the **electric kettle is the best method of
boiling water due to its speed, energy efficiency, and
convenience.**

## Why is the sky blue?

The sky appears blue due to a phenomenon called **Rayleigh
scattering**. Here's a breakdown:

1.  **Sunlight is White Light:** Sunlight, when it reaches
Earth's atmosphere, appears white to us. However, white light is
actually a combination of all the colors of the visible spectrum
(red, orange, yellow, green, blue, indigo, violet). Each of these
colors has a different wavelength.

2.  **Earth's Atmosphere:** Our atmosphere is composed of gases,
primarily nitrogen (about 78%) and oxygen (about 21%), along with
smaller amounts of other gases and tiny particles.

3.  **Scattering of Light:** As sunlight travels through the
atmosphere, it encounters these gas molecules and tiny particles.
When light waves hit these molecules, they are scattered in all
directions.

4.  **Wavelength Matters:** The key to the sky's blue color is
that **shorter wavelengths of light are scattered more
effectively than longer wavelengths.** Blue and violet light have
the shortest wavelengths in the visible spectrum, while red and
orange light have the longest.

5.  **Rayleigh Scattering:** This specific type of scattering,
which occurs when the scattering particles are much smaller than
```

```
the wavelength of the light, is known as Rayleigh scattering. It's much more efficient at scattering shorter wavelengths.

6.  **Dominance of Blue:** Because blue light is scattered in all directions by the gas molecules, it reaches our eyes from all parts of the sky. This is why the sky appears blue.

**Why isn't the sky violet?**

You might wonder why the sky isn't violet, since violet has an even shorter wavelength than blue and should scatter even more. There are a couple of reasons for this:

*   **Our Eyes' Sensitivity:** Our eyes are more sensitive to blue light than to violet light.
*   **Sunlight Spectrum:** The sun emits slightly less violet light than blue light.

So, while violet light is scattered, the combination of our eyes' sensitivity and the amount of blue light available makes us perceive the sky as blue.

**What about sunsets and sunrises?**

At sunrise and sunset, the sunlight has to travel through a much greater thickness of the atmosphere to reach our eyes. This means that most of the shorter, bluer wavelengths are scattered away, leaving the longer, redder and orange wavelengths to pass through more directly, giving us those beautiful red and orange hues.
------------------------------

--- 두 번째 요청 (한국어) ---
입력 프롬프트: 지구에 대해 말해 주세요

--- 모델 응답 텍스트 ---
## 물을 끓이는 가장 좋은 방법

물을 끓이는 "가장 좋은" 방법은 목적과 상황에 따라 달라집니다. 하지만 일반적으로 다음의 사항들을 고려하면 효율적이고 안전하게 물을 끓일 수 있습니다.

**1. 적절한 용기 선택:**

* **주전자:** 가장 일반적이고 효율적인 방법입니다. 전기주전자는 물을 빠르게 끓일 수 있으며, 가스레인지용 주전자도 좋습니다.
* **냄비:** 냄비로도 물을 끓일 수 있지만, 주전자보다는 열 전달 효율이 낮아 시간이 더 걸릴 수 있습니다.
* **IH 조리기구:** IH 코일이 내장된 조리기구는 바닥이 평평한 금속 용기를 사용해야 합니다.

**2. 물의 양:**
```

* **필요한 만큼만 끓이기:** 너무 많은 물을 끓이면 에너지가 낭비됩니다. 필요한 양만큼만 끓이는 것이 좋습니다.
* **최소/최대 표시 확인:** 주전자나 냄비에 표시된 최소 및 최대 물 높이를 지키는 것이 안전합니다.

3. 가열 방식:

* **전기주전자:** 가장 빠르고 편리합니다. 온도를 조절할 수 있는 제품도 있습니다.
* **가스레인지:** 불꽃의 세기를 조절하여 끓이는 속도를 제어할 수 있습니다.
* **인덕션 (IH) 조리기구:** 에너지 효율이 높고 빠릅니다.

4. 끓이는 속도:

* **강불:** 물을 빠르게 끓일 수 있지만, 에너지 소비가 많고 넘칠 위험이 있습니다.
* **중불:** 적절한 속도로 끓일 수 있으며, 에너지 효율도 좋습니다.
* **약불:** 천천히 끓이지만, 시간은 오래 걸립니다.

5. 추가 팁:

* **뚜껑 덮기:** 뚜껑을 덮으면 열 손실을 줄여 물이 더 빨리 끓고 에너지도 절약됩니다.
* **끓기 시작하면 불 줄이기:** 물이 끓기 시작하면 불꽃을 약간 줄여도 온도가 유지됩니다.
* **끓인 물 활용:** 끓인 물은 음료뿐만 아니라 요리, 소독 등 다양한 용도로 활용할 수 있습니다.

요약하자면,

* **가장 빠르고 편리하게 끓이고 싶다면:** 전기주전자를 사용하고 뚜껑을 덮으세요.
* **가스레인지 사용 시:** 중불로 끓이고 뚜껑을 덮으면 효율적입니다.
* **에너지 절약을 중요시한다면:** 필요한 만큼만 끓이고 뚜껑을 덮어 끓이세요.

하늘은 왜 파란가요?

하늘이 파랗게 보이는 이유는 **빛의 산란(Scattering) 현상** 때문입니다. 좀 더 자세히 설명해 드리겠습니다.

1. **햇빛은 다양한 색깔의 빛으로 이루어져 있습니다.** 우리가 흔히 무지개에서 보는 빨강, 주황, 노랑, 초록, 파랑, 남색, 보라색과 같은 다양한 파장의 빛이 섞여 있습니다.

2. **지구 대기의 구성:** 지구 대기에는 질소, 산소와 같은 기체 분자들과 먼지, 물방울 등의 작은 입자들이 떠다닙니다.

3. **레일리 산란 (Rayleigh Scattering):** 햇빛이 대기 중의 작은 기체 분자들과 부딪히면 빛은 사방으로 흩어지는데, 이를 산란이라고 합니다. 이때 **파장이 짧은 파란색 계열의 빛(보라색, 파란색)이 파장이 긴 빨간색 계열의 빛보다 훨씬 더 강하게 산란됩니다.** 이것을 **레일리 산란**이라고 합니다.

4. **우리 눈에 보이는 현상:** 태양에서 출발한 햇빛이 대기를 통과할 때, 파장이 짧은 파란색 빛은 대기 분자들에 의해 사방으로 흩어져 우리 눈에 도달합니다. 따라서 하늘을 바라보

면 우리 눈에 가장 많이 들어오는 흩어진 빛인 파란색을 보게 되는 것입니다.

그렇다면 보라색은 왜 안 보일까요?

사실 보라색 빛도 파장이 짧아 파란색보다 더 많이 산란됩니다. 하지만 우리 눈은 파란색에 더 민감하게 반응하고, 태양에서 나오는 빛 자체가 파란색에 비해 보라색의 비율이 약간 더 낮기 때문에 하늘은 주로 파랗게 보입니다.

해 질 녘 하늘이 붉은 이유:

해 질 녘에는 태양빛이 더 두꺼운 대기층을 통과해야 합니다. 이 과정에서 파장이 짧은 파란색 빛은 이미 대부분 산란되어 흩어져 버립니다. 반면 파장이 긴 빨간색, 주황색 빛은 상대적으로 더 적게 산란되어 우리 눈에 도달하기 때문에 하늘이 붉게 보이는 것입니다.

요약하자면, **햇빛이 대기 중의 작은 입자들에 의해 산란될 때, 파장이 짧은 파란색 빛이 더 많이 흩어져 우리 눈에 들어오기 때문에 하늘은 파랗게 보이는 것**입니다.

```
# 3_2. 구글 클라우드 실습 질문과 같은 질문으로 입력해보기 (추천)

prompt_en_3_2 = "What's the best method of boiling water?"
prompt_ko_3_2 = "물을 끓이는 가장 좋은 방법은 무엇인가요?"

# --- 첫 번째 프롬프트 호출 (영어) ---
try:
    print("\n--- 첫 번째 요청 (영어) ---")
    response_en = client.models.generate_content(
        model='gemini-2.5-flash-lite',                                  # 사용할 모델 지정
        contents=prompt_en_3_2                                          # 영어 프롬프트 입력
    )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response_en.text)
    print("-" * 30)

    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    #print("\n--- 모델 응답 전체JSON ---")
    #print(response.model_dump_json(
    #    exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
```

```
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")

# --- 두 번째 프롬프트 호출 (한국어) ---
try:
    print("\n--- 두 번째 요청 (한국어) ---")
    response_ko = client.models.generate_content(
        model='gemini-2.5-flash-lite',
# 사용할 모델 지정
        contents=prompt_ko_3_2
# 한국어 프롬프트 입력
    )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_ko}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response_ko.text)
    print("-" * 30)

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력

```
--- 첫 번째 요청 (영어) ---
입력 프롬프트: 제 1 차 세계대전의 원인을 세 가지로 요약해 주세요.

--- 모델 응답 텍스트 ---
The "best" method of boiling water depends entirely on your
**priorities**. There isn't one single method that is universally
superior. Here's a breakdown of common methods and their
advantages/disadvantages:

**1. Stovetop (Gas or Electric)**

*   **Method:** Place a pot or kettle filled with water on a
burner and turn on the heat.
*   **Pros:**
    *   **Widely available:** Most kitchens have a stovetop.
    *   **Controllable:** You can easily adjust the heat level.
    *   **Versatile:** Can boil large or small quantities of
water.
```

```
    *   **Good for specific uses:** Excellent for making pasta, tea, or coffee where you need to control the volume and heating speed.
*   **Cons:**
    *   **Can be slow:** Especially for larger quantities, it can take longer than other methods.
    *   **Energy inefficient:** Can lose heat to the surrounding air, especially with open burners.
    *   **Requires supervision:** You need to be present to turn off the heat.
*   **Best for:** General cooking, when you need precise temperature control, or when you don't have access to other appliances.

**2. Electric Kettle**

*   **Method:** Fill the kettle with water, plug it in, and flip the switch.
*   **Pros:**
    *   **Fast:** Electric kettles are designed for speed and are typically the fastest method for boiling smaller quantities of water.
    *   **Energy efficient:** They are generally more energy-efficient than stovetop boiling as they are enclosed and insulated.
    *   **Automatic shut-off:** Most have an automatic shut-off feature when the water boils, making them safe and convenient.
    *   **Portable:** Easy to use in various locations (office, dorm room, etc.).
*   **Cons:**
    *   **Limited capacity:** Usually designed for smaller volumes.
    *   **Requires electricity:** Useless without a power source.
    *   **Can be more expensive upfront:** Compared to a simple pot.
*   **Best for:** Quick hot drinks (tea, coffee, instant noodles), small households, convenience, and energy efficiency.

**3. Microwave**

*   **Method:** Place a microwave-safe container filled with water in the microwave and heat it.
*   **Pros:**
    *   **Fast (for small amounts):** Can be surprisingly quick for a single mug of water.
    *   **Convenient:** No need to get out pots or kettles.
*   **Cons:**
    *   **Uneven heating:** Can create hot spots and cooler areas.
    *   **Risk of superheating:** Water can heat beyond its
```

```
boiling point without appearing to boil, and then violently erupt
when disturbed (e.g., by adding sugar or a tea bag). **Always
stir the water before and after heating, or place a wooden
spoon/skewer in the water.**
    *   **Limited capacity:** You can only heat small amounts at
a time.
    *   **Not suitable for large quantities:** Impractical for
anything more than a cup or two.
    *   **Less energy efficient for larger amounts:**
*   **Best for:** Heating a single mug of water for a drink when
speed and absolute efficiency aren't critical, and you're aware
of the superheating risk.

**4. Immersion Heater**

*   **Method:** A portable heating element is placed directly
into the water.
*   **Pros:**
    *   **Portable:** Great for travel or situations where you
don't have access to a stovetop or kettle.
    *   **Can boil water in various containers:** Cups, bowls,
etc.
*   **Cons:**
    *   **Safety concerns:** Requires careful use to avoid
electric shock or damage.
    *   **Can be slow:** Depending on the wattage.
    *   **Not always as efficient:** As a dedicated electric
kettle.
*   **Best for:** Travel, camping (with a power source), or
situations where other appliances are unavailable.

**5. Instant Hot Water Dispenser**

*   **Method:** A countertop appliance that heats and dispenses
hot water on demand.
*   **Pros:**
    *   **Instant access:** Water is heated very quickly and kept
at a hot temperature.
    *   **Convenient:** No need to wait for water to boil.
*   **Cons:**
    *   **High upfront cost:** These are typically expensive
appliances.
    *   **Energy consumption:** Keeps water hot constantly, which
can consume more energy than heating on demand.
    *   **Limited temperature control:** Often only offers one or
two temperature settings.
*   **Best for:** High-demand users who frequently need hot water
throughout the day, like in offices or busy households.

**Factors to Consider When Choosing the "Best" Method:**
```

```
*   **Speed:** How quickly do you need the water boiled?
*   **Quantity:** How much water do you need to boil?
*   **Energy Efficiency:** Are you concerned about electricity
consumption?
*   **Convenience:** How much effort are you willing to put in?
*   **Safety:** What are your safety concerns?
*   **Availability of Appliances:** What do you have access to?
*   **Purpose:** What will you be using the hot water for?

**In Summary:**

*   **For Speed and Efficiency (small quantities): Electric
Kettle**
*   **For Versatility and Control (larger quantities): Stovetop**
*   **For Quickest Mug of Water (with caution): Microwave**
*   **For Travel/No Appliance Access: Immersion Heater**

Ultimately, the "best" method is the one that best suits your
immediate needs and preferences. For most people in a standard
kitchen, the **electric kettle** often wins for everyday boiling
of water for drinks due to its speed, efficiency, and safety
features. However, the **stovetop** remains indispensable for
cooking and larger volumes.
------------------------------

--- 두 번째 요청 (한국어) ---
입력 프롬프트: 지구에 대해 말해 주세요

--- 모델 응답 텍스트 ---
물을 끓이는 데 '가장 좋은' 방법은 사실 **무엇을 위해 끓이는지에 따라 달라집니다.** 하
지만 일반적인 상황에서 가장 효율적이고 빠른 방법을 기준으로 설명해 드릴게요.

**가장 일반적이고 효율적인 방법:**

1.  **적절한 용기 선택:**
    *   **냄비 또는 주전자:** 스테인리스 스틸, 법랑 코팅된 냄비, 또는 전기 주전자
등 바닥이 넓고 열전도율이 좋은 용기를 사용하는 것이 좋습니다.
    *   **뚜껑 사용:** 뚜껑을 덮으면 열이 외부로 빠져나가는 것을 막아 물이 더 빨리 끓
습니다.

2.  **적절한 양의 물:**
    *   필요한 만큼의 물만 끓이는 것이 에너지 효율적입니다. 너무 많은 물을 끓이면 시간
이 오래 걸리고 에너지 낭비가 됩니다.

3.  **강한 불 사용 (처음에는):**
    *   가스레인지의 경우, 처음에는 중간-강한 불로 설정하여 빠르게 온도를 올리는 것
이 좋습니다.
    *   인덕션이나 하이라이트의 경우, 최고 온도로 설정합니다.
```

```
    4.  **가열:**
        *   용기를 가열 장치에 올려놓고 불을 켭니다.
        *   물이 끓기 시작하면 거품이 올라오고 김이 나기 시작합니다.

    5.  **끓는점 확인:**
        *   물이 계속 끓는 것을 확인합니다. (보통 100°C)

    **추가적인 팁 및 고려사항:**

    *   **깨끗한 물:** 깨끗하고 신선한 물을 사용하면 맛이 더 좋습니다.
    *   **고도:** 해발고도가 높은 곳에서는 물의 끓는점이 낮아집니다.
    *   **압력솥:** 압력솥을 사용하면 일반 냄비보다 훨씬 빨리 물을 끓일 수 있습니다. 특히 요리 시간을 단축하고 싶을 때 유용합니다.
    *   **전기 주전자:** 전기 주전자는 특정 용량의 물을 가장 빠르고 효율적으로 끓일 수 있도록 설계되어 있어 매우 편리합니다.
    *   **끓인 물 재가열:** 이미 끓였던 물을 다시 끓이면 산소량이 줄어들어 맛이 덜할 수 있습니다. 되도록 신선한 물을 끓이는 것이 좋습니다.
    *   **건조 상태 확인:** 냄비가 완전히 마른 상태인지 확인하세요. 물기가 남아있으면 끓는 데 시간이 조금 더 걸릴 수 있습니다.

    **결론적으로,**

    *   **가장 빠르고 일반적인 방법:** 뚜껑을 덮고, 열전도율이 좋은 냄비나 주전자를 사용하여 강한 불(또는 최고 온도)로 끓이는 것입니다.
    *   **가장 편리하고 효율적인 방법:** 전기 주전자를 사용하는 것입니다.

    어떤 용도로 물을 끓이시는지에 따라 더 구체적인 답변을 드릴 수 있습니다. 혹시 특별한 목적이 있으신가요?
    --------------------------------
```

```python
# 3_3 - 비추천/추천 프롬프트 작성 (유사한 유형)

# ver_en 비추천 리스트
prompt_list_en_not_rec3 = [
    "What are the benefits of meditation and how do airplanes fly?",
    "Summarize the history of the internet and write a poem about
it.",
    "Explain photosynthesis and recommend three houseplants."
]

# ver_ko 비추천 리스트
prompt_list_ko_not_rec3 = [
    "명상은 어떤 이점이 있고, 비행기는 어떻게 나는가요?",
    "인터넷의 역사를 요약하고, 그것에 대한 시를 써 주세요.",
    "광합성을 설명하고, 키우기 좋은 식물 세 가지를 추천해 주세요."
]

# ver_en 추천 리스트
prompt_list_en_rec3 = [
```

```
    "What are the benefits of meditation?",
    "Summarize the history of the internet.",
    "Explain photosynthesis."
]

# ver_ko 추천 리스트
prompt_list_ko_rec3 = [
    "명상의 이점은 무엇인가요?",
    "인터넷의 역사를 요약해 주세요.",
    "광합성에 대해 설명해 주세요."
]

# 비추천 리스트 프롬프트 질문하기

# --- 첫 번째 비추천 리스트 프롬프트 호출 (영어) ---
# prompt_list_en_not_rec3
try:
    # 리스트에 있는 프롬프트를 반복문으로 하나씩 호출
    for prompt in prompt_list_en_not_rec3:
        print(f"\n--- 요청된 프롬프트 ---")
        print(f"입력: {prompt}")

        response = client.models.generate_content(
            model='gemini-2.5-flash-lite',                          # 모델
호출
            contents=prompt                                         # 반복
문으로 가져온 프롬프트 하나를 입력값으로 사용
        )

        # 응답 텍스트 출력
        print("\n--- 모델 응답 텍스트 ---")
        print(response.text)
        print("-" * 30)

    # ---------- 디버깅용 출력하지 않기 -------------
    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    # print("\n--- 모델 응답 전체JSON ---")
    # print(response.model_dump_json(
    #     exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력 (29 초)

```
--- 요청된 프롬프트 ---
입력: What are the benefits of meditation and how do airplanes fly?

--- 모델 응답 텍스트 ---
Let's break down your questions:

## Benefits of Meditation:

Meditation is a practice that involves training the mind to focus and redirect thoughts, often leading to a state of mental clarity and emotional calm. Here are some of the well-established benefits:

**Mental & Emotional Benefits:**

*   **Stress Reduction:** This is perhaps the most widely known benefit. Meditation can lower cortisol levels (the stress hormone), helping you feel more relaxed and less overwhelmed by daily pressures.
*   **Anxiety Management:** By teaching you to observe your thoughts without judgment, meditation can help detach from anxious rumination and reduce the intensity of anxious feelings.
*   **Improved Emotional Regulation:** Regular practice can help you become more aware of your emotions and develop healthier ways to respond to them, rather than reacting impulsively.
*   **Increased Self-Awareness:** Meditation encourages introspection, allowing you to understand your thoughts, feelings, and behaviors more deeply. This can lead to greater self-compassion and personal growth.
*   **Enhanced Focus and Concentration:** By training your attention, meditation can improve your ability to stay focused on tasks and resist distractions.
*   **Improved Mood:** Many people report feeling happier and more content after incorporating meditation into their routine, as it can promote positive emotions and reduce negative ones.
*   **Reduced Symptoms of Depression:** While not a replacement for professional treatment, meditation can be a valuable complementary tool for managing depression by helping to break negative thought patterns.
*   **Increased Patience and Tolerance:** As you become more aware of your own internal processes, you may find yourself becoming more patient with yourself and others.

**Physical Benefits:**

*   **Lowered Blood Pressure:** Some studies suggest that regular meditation can contribute to lower blood pressure, which is
```

```
beneficial for cardiovascular health.
*   **Improved Sleep Quality:** By calming the mind and reducing racing thoughts, meditation can make it easier to fall asleep and stay asleep.
*   **Pain Management:** Meditation can help individuals cope with chronic pain by changing their perception of pain and reducing the emotional distress associated with it.
*   **Boosted Immune System:** While research is ongoing, some studies indicate that meditation might have a positive impact on immune function.

## How Airplanes Fly:

Airplanes fly due to a combination of physics principles, primarily **aerodynamics** and the generation of **lift**. Here's a simplified explanation:

**1. The Four Forces of Flight:**

There are four main forces that act on an airplane in flight:

*   **Lift:** The upward force that opposes weight. This is the primary force that keeps the plane in the air.
*   **Weight (or Gravity):** The downward force pulling the plane towards the Earth.
*   **Thrust:** The forward force generated by the engines (propellers or jet engines) that overcomes drag.
*   **Drag:** The backward force that opposes thrust, caused by air resistance.

For an airplane to fly, **lift must be greater than or equal to weight**, and **thrust must be greater than or equal to drag**.

**2. Generating Lift (The Key to Flying):**

Lift is primarily generated by the **wings** of the airplane. The shape of an airplane wing is crucial. It's typically curved on top and flatter on the bottom, creating an **airfoil** shape. Here's how it works:

*   **Bernoulli's Principle:** As the airplane moves forward (propelled by its engines), air flows over and under the wings. Due to the curved upper surface, the air has to travel a longer distance than the air flowing beneath the flatter bottom surface. To cover this longer distance in the same amount of time, the air above the wing must move *faster* than the air below the wing.
    *   According to Bernoulli's principle, **faster-moving air exerts lower pressure**, and **slower-moving air exerts higher pressure**.
    *   Therefore, the air pressure above the wing is lower than
```

```
the air pressure below the wing.
*   **Pressure Difference Creates Lift:** This difference in air pressure creates an upward force. The higher pressure beneath the wing pushes upwards, while the lower pressure above the wing "pulls" upwards. This net upward force is **lift**.

**3. The Role of Thrust:**

*   The engines provide **thrust**, which pushes the airplane forward through the air. This forward motion is essential for air to flow over the wings and generate lift.

**4. Control Surfaces:**

Airplanes also have control surfaces that allow pilots to maneuver them:

*   **Ailerons:** Located on the trailing edge of the wings, they control **roll** (tilting left or right).
*   **Elevators:** Located on the horizontal tail, they control **pitch** (tilting the nose up or down).
*   **Rudder:** Located on the vertical tail, it controls **yaw** (turning the nose left or right).

**In Summary:**

Airplanes fly because their engines create thrust, which pushes them forward. This forward motion causes air to flow over their specially shaped wings. The wing's design creates a pressure difference (lower pressure on top, higher pressure on the bottom), resulting in an upward force called lift. When this lift is greater than the airplane's weight, the airplane ascends and flies.
------------------------------

--- 요청된 프롬프트 ---
입력: Summarize the history of the internet and write a poem about it.

--- 모델 응답 텍스트 ---
## History of the Internet: A Summary

The internet's story is one of gradual evolution, driven by military necessity, academic collaboration, and eventually, commercial innovation. Its roots lie in the **Cold War**, specifically the **late 1950s**. The fear of a devastating nuclear attack spurred the creation of a decentralized communication network that could survive even if parts of it were destroyed.

This led to the development of **ARPANET (Advanced Research
```

```
Projects Agency Network)** in the **1960s** by the US Department of Defense. ARPANET connected a few research institutions and demonstrated the feasibility of packet switching – breaking data into small packets that could travel independently and be reassembled at their destination.

The **1970s** saw the development of key protocols that would form the backbone of the internet. **TCP/IP (Transmission Control Protocol/Internet Protocol)**, developed by Vint Cerf and Bob Kahn, standardized how data was transmitted and routed across different networks. This allowed disparate networks to communicate with each other, effectively creating an "internetwork."

During the **1980s**, ARPANET expanded, and other networks emerged, often connected via these new protocols. The **National Science Foundation (NSF)** played a crucial role by creating **NSFNET**, which connected universities and research centers across the US. This opened up the network to a wider academic community, fostering collaboration and innovation.

The **late 1980s and early 1990s** marked a pivotal shift. The development of the **World Wide Web** by **Tim Berners-Lee** at CERN revolutionized how information was accessed and shared. The Web introduced concepts like **HTML (Hypertext Markup Language)** for creating web pages, **HTTP (Hypertext Transfer Protocol)** for transferring them, and **URLs (Uniform Resource Locators)** for addressing them. This made the internet accessible to the general public in a user-friendly way.

The **mid-1990s** saw the **privatization of the internet** as commercial entities began offering access and services. The development of graphical web browsers like **Mosaic** and later **Netscape Navigator** made the Web even more intuitive and appealing, leading to explosive growth.

Since then, the internet has continued its relentless evolution. We've seen the rise of **search engines**, **e-commerce**, **social media**, **mobile internet**, and the **Internet of Things (IoT)**. The internet has transformed nearly every aspect of modern life, from communication and commerce to education and entertainment, becoming an indispensable global infrastructure.

---

## The Weaving of the Web

From a fear-laced dawn, a seed was sown,
A network spun, lest all be overthrown.
ARPANET's hum, a whisper in the wire,
```

```
To link the minds, and quench a burning fire.

Then packets danced, in digital decree,
Across the void, for all the world to see.
TCP/IP, the language understood,
Connecting realms, as only networks could.

The NSF stepped in, a widening embrace,
Academic halls found their digital space.
But still a riddle, for the common man,
A hidden code, within a complex plan.

Then Berners-Lee, with vision clear and bright,
Unfurled the Web, and brought forth shining light.
With HTML's charm, and HTTP's swift hand,
A universe of pages, to understand.

Mosaic's window, opened wide and free,
Netscape's swift sail, upon a digital sea.
From research labs to every single home,
The internet bloomed, no longer left to roam.

Now search engines chart, where knowledge resides,
And commerce thrives, where data gently glides.
From social whispers to a global shout,
The internet's story, forever plays it out.

A tapestry woven, with threads unseen,
Connecting all, a vibrant, boundless scene.
From its hushed birth, to what it now has grown,
The internet's magic, forever known.
------------------------------

--- 요청된 프롬프트 ---
입력: Explain photosynthesis and recommend three houseplants.

--- 모델 응답 텍스트 ---
## Photosynthesis: The Magic of Plant Food Production

Photosynthesis is the remarkable process by which plants, algae,
and some bacteria convert light energy into chemical energy,
primarily in the form of glucose (a sugar). This glucose serves
as their food source, providing the energy needed for growth,
reproduction, and all other life functions. It's the foundation
of most food chains on Earth, as the energy captured by plants is
eventually transferred to herbivores and then to carnivores.

Here's a breakdown of the key components and steps involved:

**The Essential Ingredients:**
```

*   **Sunlight:** The primary energy source. Plants capture light energy, specifically within the red and blue wavelengths of the visible spectrum.
*   **Carbon Dioxide (CO_2):** A gas absorbed from the atmosphere through tiny pores on the leaves called **stomata**.
*   **Water (H_2O):** Absorbed from the soil by the plant's roots and transported to the leaves.

The "Kitchen" - Chloroplasts:

Photosynthesis takes place within specialized organelles inside plant cells called **chloroplasts**. These are tiny green factories containing the vital pigment **chlorophyll**.

The Process (Simplified):

Photosynthesis occurs in two main stages:

1. **Light-Dependent Reactions (The "Energy Capture" Phase):**
    *   This stage happens within the **thylakoid membranes** of the chloroplasts.
    *   Chlorophyll absorbs sunlight, exciting electrons.
    *   This captured light energy is used to split water molecules (H_2O) into oxygen (O_2), protons (H^+), and electrons.
    *   The oxygen is released as a byproduct into the atmosphere (what we breathe!).
    *   The energy from the excited electrons is used to create two energy-carrying molecules: **ATP** (adenosine triphosphate) and **NADPH** (nicotinamide adenine dinucleotide phosphate). These are like temporary energy storage batteries.

2. **Light-Independent Reactions (The Calvin Cycle - The "Sugar Building" Phase):**
    *   This stage occurs in the **stroma**, the fluid-filled space within the chloroplasts, surrounding the thylakoids.
    *   It doesn't directly require sunlight, but it relies on the ATP and NADPH produced during the light-dependent reactions.
    *   Carbon dioxide (CO_2) from the atmosphere enters the cycle.
    *   Using the energy from ATP and the reducing power of NADPH, the CO_2 molecules are "fixed" and converted into glucose ($C_6H_{12}O_6$). This is a series of complex chemical reactions.

The Overall Equation:

The simplified chemical equation for photosynthesis is:

$6CO_2 + 6H_2O + \text{Light Energy} \rightarrow C_6H_{12}O_6 + 6O_2$

```
In words: Six molecules of carbon dioxide plus six molecules of water, in the presence of light energy, produce one molecule of glucose and six molecules of oxygen.

**Why Photosynthesis is Crucial:**

*   **Food for Plants:** Provides the energy plants need to grow and survive.
*   **Oxygen Production:** Releases the oxygen we and most other organisms need to breathe.
*   **Carbon Sequestration:** Removes carbon dioxide from the atmosphere, helping to regulate the Earth's climate.
*   **Foundation of Food Webs:** The energy stored in glucose is the primary energy source for almost all life on Earth.

---

## Three Houseplants That Excel at Photosynthesis (and are Great for Your Home!)

When recommending houseplants for their photosynthetic prowess, we're looking for plants that are generally healthy, robust growers, and readily available. Here are three excellent choices:

1.  **Snake Plant (Sansevieria trifasciata / Dracaena trifasciata):**

    *   **Why it's great for photosynthesis:** Snake plants are incredibly resilient and known for their efficient **CAM photosynthesis**. This means they open their stomata to absorb CO2 at night, reducing water loss during the hot day. They are particularly adept at converting CO2 into oxygen, even in low-light conditions.
    *   **Care Tips:** They thrive in bright, indirect light but can tolerate low light. Water only when the soil is completely dry to prevent root rot. They are very forgiving and can handle a range of conditions.
    *   **Benefits:** Known for air purification, removing toxins like formaldehyde and benzene from the air. They also produce oxygen, making them excellent bedroom plants.

2.  **Spider Plant (Chlorophytum comosum):**

    *   **Why it's great for photosynthesis:** Spider plants are vigorous growers that produce plenty of foliage, meaning they have a large surface area for capturing light and CO2. They are efficient at converting these into energy and releasing oxygen.
    *   **Care Tips:** Prefer bright, indirect light. Water when the top inch of soil feels dry. They are also known for their "spiderettes" (baby plants) which are easy to propagate. Avoid overwatering.
```

```
    *   **Benefits:** Excellent air purifiers, helping to remove
    common household pollutants like formaldehyde and xylene. They
    are also non-toxic to pets, making them a safe choice for pet
    owners.

    3.  **ZZ Plant (Zamioculcas zamiifolia):**

    *   **Why it's great for photosynthesis:** ZZ plants are
    another champion of drought tolerance and also exhibit **CAM
    photosynthesis**. Their thick, waxy leaves help them store water
    and minimize water loss, allowing them to thrive even with
    infrequent watering. Their dense foliage means they are actively
    photosynthesizing when conditions are right.
    *   **Care Tips:** They can tolerate very low light
    conditions but will grow best in bright, indirect light. Water
    sparingly — allow the soil to dry out completely between
    waterings. Overwatering is the most common way to kill a ZZ
    plant.
    *   **Benefits:** Highly effective at removing airborne
    toxins like xylene, toluene, and benzene. They are incredibly
    low-maintenance, making them ideal for beginners or those with
    busy lifestyles.

    These three plants are not only beautiful additions to your home
    but also diligent little photosynthetic powerhouses, contributing
    to healthier air and a more vibrant living space!
    ------------------------------
```

```python
# 비추천 리스트 프롬프트 질문하기

# --- 첫 번째 비추천 리스트 프롬프트 호출 (한국) ---
# prompt_list_ko_not_rec3
try:
    # 리스트에 있는 프롬프트를 반복문으로 하나씩 호출
    for prompt in prompt_list_ko_not_rec3:
        print(f"\n--- 요청된 프롬프트 ---")
        print(f"입력: {prompt}")

        response = client.models.generate_content(
            model='gemini-2.5-flash-lite',                          # 모델
호출
            contents=prompt                                         # 반복
문으로 가져온 프롬프트 하나를 입력값으로 사용
        )

        # 응답 텍스트 출력
        print("\n--- 모델 응답 텍스트 ---")
        print(response.text)
        print("-" * 30)
```

```
    # ---------- 디버깅용 출력하지 않기 -------------
    # 응답 전체 내용(JSON 형식) 출력(디버깅이나 상세 정보 확인용)
    # print("\n--- 모델 응답 전체JSON ---")
    # print(response.model_dump_json(
    #     exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력 (23 초)

```
--- 요청된 프롬프트 ---
입력: 명상은 어떤 이점이 있고, 비행기는 어떻게 나는가요?

--- 모델 응답 텍스트 ---
## 명상의 이점

명상은 단순히 눈을 감고 앉아 있는 것 이상의 깊고 다양한 이점을 제공합니다. 과학적으로도 입증된 명상의 주요 이점은 다음과 같습니다.

**정신적, 정서적 이점:**

* **스트레스 감소 및 불안 완화:** 명상은 스트레스 호르몬인 코르티솔 수치를 낮추고, 편안함과 이완을 촉진하는 신경계를 활성화하여 스트레스와 불안감을 줄이는 데 탁월한 효과가 있습니다.
* **감정 조절 능력 향상:** 자신의 감정을 객관적으로 관찰하고 이해하는 능력을 길러줌으로써, 부정적인 감정에 휘둘리지 않고 침착하게 대처할 수 있도록 돕습니다.
* **집중력 및 주의력 향상:** 끊임없이 산만해지는 생각을 멈추고 현재 순간에 집중하는 연습을 통해 집중력과 주의력을 크게 향상시킬 수 있습니다. 이는 학업, 업무 등 다양한 활동에서 생산성을 높이는 데 기여합니다.
* **정신적 명료함 및 인지 능력 개선:** 뇌파를 안정시키고 뇌 기능을 최적화하여 정신적 명료함, 기억력, 문제 해결 능력 등을 향상시킬 수 있습니다.
* **긍정적인 사고방식 함양:** 감사, 연민 등 긍정적인 감정을 키우고 자기 비난적인 생각을 줄여 전반적인 삶의 만족도를 높이는 데 도움을 줍니다.
* **창의력 증진:** 마음을 비우고 고요하게 있을 때 새로운 아이디어가 떠오르기 쉬우며, 창의적인 문제 해결 능력을 키울 수 있습니다.
* **수면 질 개선:** 마음의 평온을 되찾고 긴장을 완화하여 불면증을 해소하고 숙면을 취하는 데 도움을 줍니다.

**신체적 이점:**
```

* **혈압 감소:** 스트레스 감소와 함께 혈압을 낮추는 데 긍정적인 영향을 미쳐 심혈관 건강에 도움을 줄 수 있습니다.
* **면역력 강화:** 스트레스 감소는 면역 시스템을 약화시키는 요인을 줄여 면역 기능을 강화하는 데 기여할 수 있습니다.
* **만성 통증 완화:** 통증에 대한 인식 방식을 변화시키고 스트레스를 줄여 만성적인 통증을 완화하는 데 도움을 줄 수 있습니다.
* **소화 불량 개선:** 스트레스는 소화 시스템에 부정적인 영향을 미치는데, 명상은 소화를 개선하는 데 도움을 줄 수 있습니다.

비행기는 어떻게 나는가요?

비행기가 하늘을 나는 원리는 **양력(Lift), 무게(Weight), 추력(Thrust), 항력(Drag)** 이 네 가지 힘의 균형과 상호작용에 기반합니다.

1. 날개와 양력 (Lift)

비행기가 나는 가장 핵심적인 원리는 **양력**입니다. 양력은 비행기의 날개에서 발생하는 위로 뜨는 힘입니다.

* **날개의 모양:** 비행기 날개의 위쪽은 아래쪽보다 더 볼록한 형태를 띠고 있습니다.
* **베르누이의 원리:** 공기가 날개 위를 흐를 때, 위쪽의 더 긴 경로를 통과해야 하므로 속도가 더 빨라집니다. 베르누이의 원리에 따르면, 유체의 속도가 빠를수록 압력이 낮아집니다. 따라서 날개 위쪽의 공기 압력은 날개 아래쪽의 공기 압력보다 낮아집니다.
* **압력 차이:** 이렇게 발생한 압력 차이가 날개를 위로 밀어 올리는 힘, 즉 **양력**을 만들어냅니다.

2. 네 가지 힘의 균형

* **무게 (Weight):** 지구 중력에 의해 비행기를 아래로 끌어당기는 힘입니다.
* **양력 (Lift):** 날개에서 발생하는 위로 뜨는 힘으로, 무게를 상쇄해야 비행기가 떠오를 수 있습니다.
* **추력 (Thrust):** 비행기를 앞으로 밀어내는 힘입니다. 주로 엔진 (제트 엔진 또는 프로펠러)에서 발생합니다.
* **항력 (Drag):** 공기 저항으로 인해 비행기의 움직임을 방해하는 힘입니다.

비행이 이루어지는 과정:

1. **엔진 작동:** 엔진이 작동하여 **추력**을 발생시킵니다.
2. **앞으로 이동:** 추력 덕분에 비행기는 활주로를 따라 앞으로 빠르게 이동합니다.
3. **양력 발생:** 비행 속도가 빨라지면서 날개 위아래의 공기 흐름 속도 차이가 커지고, **양력**이 발생하기 시작합니다.
4. **이륙:** 양력이 비행기의 **무게**보다 커지는 순간, 비행기는 지면에서 떠올라 하늘로 비행합니다.
5. **순항:** 순항 중에는 양력이 무게와 같고, 추력이 항력과 같도록 유지되어 일정한 고도와 속도로 비행합니다.
6. **감속 및 착륙:** 속도를 줄이고 양력을 감소시키면 비행기는 점차 고도를 낮추어 착륙하게 됩니다.

간단히 말해, 비행기는 날개 모양과 엔진의 힘을 이용하여 공기 역학적인 원리를 통해 하늘을

```
날 수 있습니다.
------------------------------

--- 요청된 프롬프트 ---
입력: 인터넷의 역사를 요약하고, 그것에 대한 시를 써 주세요.

--- 모델 응답 텍스트 ---
## 인터넷의 역사: 인류의 연결, 지식의 바다

인터넷의 역사는 인류의 소통과 정보 교환 방식을 혁신적으로 변화시킨 여정입니다. 몇 가지 중요한 단계를 중심으로 요약해 보겠습니다.

**1. 탄생과 초기 발전 (1960 년대 - 1980 년대):**

*   **냉전 시대의 군사적 필요성:** 인터넷의 기원은 1960 년대 미국 국방부의 ARPA(Advanced Research Projects Agency)에서 시작되었습니다. 소련의 스푸트니크 충격 이후, 미국은 핵 공격에도 견딜 수 있는 분산된 통신망의 필요성을 느꼈고, 이러한 연구의 결과로 **ARPANET**이 탄생했습니다.
*   **핵심 기술의 등장:** ARPANET 은 데이터를 패킷 단위로 분할하여 각기 다른 경로를 통해 전송하는 **패킷 교환 방식**을 사용했습니다. 이는 기존의 회선 교환 방식보다 훨씬 효율적이고 안정적이었습니다. 이후 **TCP/IP 프로토콜**이 개발되어 서로 다른 네트워크 간의 상호 운용성을 확보하게 되면서, ARPANET 은 점차 더 넓은 네트워크로 확장될 수 있는 기반을 마련했습니다.
*   **학계와 연구기관으로의 확산:** 초기에는 주로 군사적 목적과 학계 연구를 위해 사용되었지만, 대학과 연구기관 간의 정보 공유와 협력을 촉진하는 중요한 도구가 되었습니다.

**2. 월드 와이드 웹의 탄생과 대중화 (1990 년대):**

*   **팀 버너스리의 혁신:** 1989 년, CERN 의 과학자 **팀 버너스리**는 정보를 쉽게 접근하고 공유할 수 있는 **월드 와이드 웹(World Wide Web, WWW)**을 제안하고 개발했습니다. 그는 **HTML(Hypertext Markup Language)**, **HTTP(Hypertext Transfer Protocol)**, **URL(Uniform Resource Locator)**과 같은 핵심 기술들을 창안하여 사용자 친화적인 인터넷 환경을 구축했습니다.
*   **개인용 컴퓨터와 모뎀의 확산:** 개인용 컴퓨터(PC)의 보급과 저렴한 모뎀의 등장으로 일반 대중들도 인터넷에 접근할 수 있게 되었습니다.
*   **브라우저의 등장:** 1993 년 **Mosaic** 브라우저의 출시와 뒤이은 **Netscape Navigator**의 성공은 웹 페이지를 시각적으로 탐색하는 것을 가능하게 했고, 인터넷의 대중화를 가속화했습니다.

**3. 상업화와 폭발적인 성장 (2000 년대):**

*   **닷컴 버블과 이후:** 1990 년대 말에는 인터넷 기반 기업들의 폭발적인 성장과 함께 **닷컴 버블**이 형성되었지만, 2000 년대 초반 거품이 꺼진 후에도 인터넷은 꾸준히 발전했습니다.
*   **검색 엔진의 중요성 증대:** **Google**과 같은 강력한 검색 엔진의 등장으로 방대한 웹 정보 속에서 원하는 내용을 쉽게 찾을 수 있게 되었습니다.
*   **이메일, 커뮤니티, 전자상거래의 확산:** 이메일은 기본적인 통신 수단이 되었고, 온라인 커뮤니티와 포럼을 통해 다양한 사람들이 의견을 나누고 정보를 교환했습니다. 온라인 쇼핑과 전자상거래도 점차 자리 잡기 시작했습니다.
```

4. 참여와 공유의 시대: 소셜 미디어와 모바일 인터넷 (2010 년대 이후):

*   **소셜 미디어의 등장:** **Facebook, Twitter, YouTube**와 같은 소셜 미디어 플랫폼의 등장은 개인들이 콘텐츠를 생산하고 공유하는 것을 넘어, 서로 연결되고 소통하는 방식을 혁명적으로 바꾸었습니다.
*   **스마트폰과 모바일 인터넷:** 스마트폰의 보급은 인터넷을 언제 어디서든 손쉽게 접근할 수 있는 **모바일 인터넷** 시대를 열었습니다. 이는 정보 접근성과 사용 편의성을 극대화했습니다.
*   **클라우드 컴퓨팅, 빅데이터, 인공지능:** 클라우드 컴퓨팅은 데이터 저장 및 처리 방식을 변화시켰고, 빅데이터와 인공지능 기술은 인터넷을 기반으로 한 서비스의 발전을 더욱 가속화했습니다.

현재 인터넷은 우리 삶의 필수적인 부분이 되었으며, 정보, 소통, 경제, 문화 등 모든 영역에 지대한 영향을 미치고 있습니다. 끊임없이 진화하는 인터넷은 앞으로도 인류의 미래를 더욱 풍요롭게 만들어갈 것입니다.

인터넷, 연결의 노래

냉전의 그림자, 거미줄 얽힌 사념
분산된 씨앗, ARPANET 의 꿈틀거림
패킷의 춤, 경로를 찾아 떠도는 이야기
무형의 강물, 지식의 바다를 향해 흐르네.

허허벌판에, 하이퍼링크의 숲이 우거지고
팀의 손끝에서, 월드 와이드 웹이 피어나네.
HTML 의 속삭임, HTTP 의 약속
모두가 함께 엮는, 거대한 정보의 그물.

개인의 책상 위, 작은 창이 열리고
세상은 손안에, 무한한 가능성이 펼쳐지네.
이메일의 속삭임, 검색의 외침
세계를 잇는 다리, 국경도 희미해져 가네.

소셜의 파도, 관계의 새로운 물결
나의 생각, 너의 이야기가 춤을 추네.
모바일의 속도, 시간도 멈추지 않고
정보는 흐르고, 세상은 더욱 가까워지네.

우리는 연결된, 보이지 않는 끈으로
아이디어는 솟아, 미래를 함께 그려가네.
이것은 단순한 기술이 아닌, 인류의 노래
인터넷, 끝나지 않을 혁신의 여정.

--- 요청된 프롬프트 ---
입력: 광합성을 설명하고, 키우기 좋은 식물 세 가지를 추천해 주세요.

--- 모델 응답 텍스트 ---
광합성이란 무엇인가?

광합성은 식물이 살아가는 데 필수적인 에너지 생산 과정입니다. 마치 우리가 음식을 먹고 에너지를 얻는 것처럼, 식물은 **햇빛, 물, 이산화탄소**를 이용하여 스스로 영양분을 만듭니다. 이 과정에서 식물은 **산소**를 배출하며, 이는 우리를 포함한 모든 생명체에게 매우 중요합니다.

좀 더 자세히 살펴보면 다음과 같습니다.

*   **햇빛:** 식물의 잎에 있는 **엽록체(chloroplast)**라는 기관이 햇빛 에너지를 흡수합니다. 엽록체 안에는 **엽록소(chlorophyll)**라는 녹색 색소가 있어 빛을 포착하는 역할을 합니다.
*   **물:** 뿌리가 땅에서 물을 흡수하여 줄기를 통해 잎으로 전달됩니다.
*   **이산화탄소:** 식물 잎에 있는 작은 구멍인 **기공(stomata)**을 통해 공기 중의 이산화탄소를 받아들입니다.

이 세 가지 재료가 엽록체 안에서 결합하여 **포도당(glucose)**이라는 에너지원과 **산소(oxygen)**를 만들어냅니다.

광합성 과정 요약:

햇빛 + 물 + 이산화탄소 → 포도당 + 산소

이렇게 만들어진 포도당은 식물의 성장과 생존에 필요한 에너지를 공급하며, 일부는 다른 유기물 형태로 저장됩니다.

키우기 좋은 식물 추천 (광합성과 관련하여)

광합성을 활발하게 하는 것은 물론, 비교적 키우기 쉬운 식물 세 가지를 추천해 드립니다.

1. 스킨답서스 (Scindapsus pictus)

*   **추천 이유:**
    *   **강한 생명력:** 스킨답서스는 어두운 환경이나 건조한 환경에서도 잘 견디는 편이라 초보자가 키우기 매우 좋습니다.
    *   **공기 정화 능력:** 실내 유해 물질을 흡수하는 능력이 뛰어나 실내 환경을 쾌적하게 만드는 데 도움을 줍니다.
    *   **쉬운 번식:** 꺾꽂이를 통해 쉽게 번식시킬 수 있어 친구나 가족에게 선물하기도 좋습니다.
    *   **다양한 잎 모양:** 은색 반점이 있는 무늬종이 많아 관상용으로도 인기가 많습니다.
*   **관리 팁:**
    *   **햇빛:** 직사광선보다는 밝은 간접광을 좋아합니다. 너무 어두운 곳에서는 잎의 무늬가 흐릿해질 수 있습니다.
    *   **물주기:** 겉흙이 마르면 듬뿍 주세요. 과습은 뿌리 썩음의 원인이 될 수 있으니 주의해야 합니다.

```
    ### 2. 산세베리아 (Sansevieria trifasciata)

    *   **추천 이유:**
        *   **뛰어난 공기 정화:** 포름알데히드, 벤젠 등 다양한 유해 물질을 제거하는 데 탁월하며, 밤에도 산소를 배출하는 장점이 있습니다.
        *   **건조에 강함:** 물을 자주 주지 않아도 잘 살아남아 바쁘거나 물 주는 것을 잊기 쉬운 분들에게 이상적입니다.
        *   **다양한 품종:** 잎의 모양이나 색깔이 다양한 품종이 많아 취향에 맞게 선택할 수 있습니다.
        *   **병충해에 강함:** 비교적 병충해에 강한 편입니다.
    *   **관리 팁:**
        *   **햇빛:** 밝은 간접광을 좋아하지만, 어두운 곳에서도 잘 견딥니다.
        *   **물주기:** 흙이 완전히 마른 후에 물을 주세요. 특히 겨울철에는 물 주는 횟수를 줄여야 합니다.

    ### 3. 아이비 (Hedera helix)

    *   **추천 이유:**
        *   **다양한 생장 형태:** 늘어지게 키우거나 지지대를 이용해 타고 올라가게 키우는 등 다양한 방식으로 연출할 수 있습니다.
        *   **공기 정화 효과:** 곰팡이 포자를 제거하고 포름알데히드를 흡수하는 데 도움을 줍니다.
        *   **번식 용이:** 꺾꽂이로 쉽게 번식시킬 수 있습니다.
        *   **다양한 잎 모양과 색깔:** 잎의 크기, 모양, 색깔이 다양한 품종이 많아 선택의 폭이 넓습니다.
    *   **관리 팁:**
        *   **햇빛:** 밝은 간접광을 좋아하며, 약간의 그늘에서도 잘 자랍니다.
        *   **물주기:** 겉흙이 마르면 물을 주세요. 약간 건조하게 관리하는 것이 좋습니다.

    이 세 가지 식물은 비교적 키우기 쉽고, 실내 환경을 개선하는 데 도움을 주며, 광합성을 통해 싱그러움을 더해줄 것입니다. 즐거운 식물 생활 되시길 바랍니다!
    ------------------------------
```

```python
# 추천 리스트 프롬프트 질문하기

# --- 첫 번째 비추천 리스트 프롬프트 호출 (영어) ---
# prompt_list_en_rec3
try:
    # 리스트에 있는 프롬프트를 반복문으로 하나씩 호출
    for prompt in prompt_list_en_rec3:
        print(f"\n--- 요청된 프롬프트 ---")
        print(f"입력: {prompt}")

        response = client.models.generate_content(
            model='gemini-2.5-flash-lite',                    # 모델 호출
            contents=prompt                                   # 반복문으로 가져온 프롬프트 하나를 입력값으로 사용
```

```
        )

        # 응답 텍스트 출력
        print("\n--- 모델 응답 텍스트 ---")
        print(response.text)
        print("-" * 30)

    # ---------- 디버깅용 출력하지 않기 ----------
    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    # print("\n--- 모델 응답 전체 JSON ---")
    # print(response.model_dump_json(
    #     exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력 (13 초)

```
--- 요청된 프롬프트 ---
입력: What are the benefits of meditation?

--- 모델 응답 텍스트 ---
Meditation offers a wide range of benefits, impacting your
mental, emotional, and even physical well-being. It's a practice
that cultivates a deeper connection with yourself and can lead to
significant improvements in various aspects of life. Here are
some of the key benefits:

**Mental & Cognitive Benefits:**

*   **Reduced Stress and Anxiety:** This is one of the most well-
known benefits. Meditation helps to calm the nervous system,
lower cortisol levels (the stress hormone), and create a sense of
inner peace, reducing feelings of worry and overwhelm.
*   **Improved Focus and Attention:** Regular meditation trains
your brain to stay present and less susceptible to distractions.
This can translate to better concentration at work, in studies,
and in everyday tasks.
*   **Enhanced Self-Awareness:** By observing your thoughts,
feelings, and bodily sensations without judgment, you gain a
deeper understanding of your inner landscape. This can help you
identify patterns of behavior, triggers, and underlying emotions.
```

*   **Increased Emotional Regulation:** Meditation helps you develop the ability to manage your emotions more effectively. You learn to respond to challenging feelings rather than react impulsively, leading to greater emotional stability.
*   **Improved Memory:** Some studies suggest that meditation can positively impact memory function and cognitive flexibility.
*   **Increased Creativity:** By quieting the mental chatter and opening up your mind, meditation can foster new insights and creative solutions.
*   **Greater Mental Clarity:** When the mind is less cluttered with worries and distractions, you experience clearer thinking and better decision-making.

Emotional & Psychological Benefits:

*   **Increased Self-Compassion:** As you become more aware of your inner experiences, you tend to develop a kinder and more understanding attitude towards yourself, especially during difficult times.
*   **Greater Resilience:** Meditation helps you bounce back from adversity more effectively by fostering a sense of inner strength and a more balanced perspective.
*   **Improved Mood and Reduced Depression:** By cultivating positive emotions and reducing negative rumination, meditation can contribute to a more optimistic outlook and alleviate symptoms of depression.
*   **Enhanced Sense of Well-being:** Overall, meditation can lead to a greater feeling of contentment, happiness, and fulfillment in life.
*   **Improved Relationships:** By fostering empathy, patience, and better communication skills, meditation can positively impact your interactions with others.

Physical Benefits:

*   **Lowered Blood Pressure:** The stress-reducing effects of meditation can contribute to lower blood pressure, which is beneficial for cardiovascular health.
*   **Improved Sleep Quality:** By calming the mind and reducing anxiety, meditation can make it easier to fall asleep and stay asleep, leading to more restful nights.
*   **Pain Management:** Some individuals find that meditation helps them cope with chronic pain by changing their perception of it and reducing the associated stress.
*   **Boosted Immune System:** While research is ongoing, some studies suggest that stress reduction through meditation can have a positive impact on immune function.
*   **Increased Energy Levels:** By reducing mental fatigue and promoting relaxation, meditation can paradoxically lead to increased vitality and energy.

Important Considerations:

*   **Consistency is Key:** The benefits of meditation are cumulative and become more pronounced with regular practice. Even a few minutes a day can make a difference.
*   **No "Right" Way:** There are many different types of meditation, and what works for one person may not work for another. Experiment to find a style that resonates with you.
*   **Patience is Required:** It takes time to develop the skills and experience the full range of benefits. Don't get discouraged if you don't see results immediately.
*   **Not a Cure-All:** While meditation is incredibly beneficial, it's not a substitute for professional medical or psychological treatment when needed.

In essence, meditation is a powerful tool for cultivating a more balanced, peaceful, and fulfilling life. It's an investment in your own well-being that can yield profound and lasting rewards.

--- 요청된 프롬프트 ---
입력: Summarize the history of the internet.

--- 모델 응답 텍스트 ---
The history of the internet is a fascinating journey from a niche research project to a ubiquitous global network that has fundamentally reshaped our world. Here's a summary:

1. The Genesis: ARPANET and the Cold War (1950s-1960s)

* **The Spark:** The Cold War era fueled a desire for robust and resilient communication systems, particularly in the face of potential nuclear attack. The US Department of Defense sought to create a decentralized network that could withstand damage.
* **ARPA's Vision:** The Advanced Research Projects Agency (ARPA, later DARPA) was established in 1958. In the early 1960s, ARPA began exploring the concept of a "galactic network" – a globally interconnected set of computers.
* **Packet Switching:** A key theoretical breakthrough was the development of **packet switching** by researchers like Paul Baran, Donald Davies, and Leonard Kleinrock. Instead of dedicated circuits, data would be broken into "packets," each labeled with its destination and sent independently across the network, reassembling at the other end. This made the network more efficient and resilient.

2. The Birth of ARPANET (Late 1960s - Early 1970s)

* **First Connection:** In **1969**, ARPANET, the precursor to

```
the internet, was established. The first successful message was
sent from UCLA to Stanford Research Institute (SRI). The network
initially connected a handful of research institutions.
* **Early Applications:** Primarily used for email and file
transfer among researchers, ARPANET laid the groundwork for
collaborative computing.

**3. The Foundation of Protocols: TCP/IP (1970s)**

* **The Need for Standardization:** As ARPANET grew and other
networks began to emerge, a common language was needed for them
to communicate.
* **Vint Cerf and Bob Kahn:** In the **early 1970s**, Vint Cerf
and Bob Kahn developed the **Transmission Control
Protocol/Internet Protocol (TCP/IP)**. This suite of protocols
became the fundamental building blocks of the internet, enabling
disparate networks to interconnect and exchange data seamlessly.
* **The "Internetting" Concept:** TCP/IP allowed for the creation
of an "internetwork" – a network of networks.

**4. Expansion and Diversification (1980s)**

* **ARPANET Splits:** In **1983**, ARPANET was split into two
networks: ARPANET for research and MILNET for military use.
* **Growth of Other Networks:** Other networks emerged, such as
CSNET (for computer science departments) and BITNET (for academic
institutions), and TCP/IP facilitated their connection to the
growing internet.
* **Domain Name System (DNS):** The introduction of DNS in
**1983** made the internet more user-friendly by translating
numerical IP addresses into human-readable domain names (like
google.com).
* **NSFNET:** The National Science Foundation (NSF) established
NSFNET in **1985**, which became a major backbone for the
internet, connecting supercomputing centers and eventually
replacing ARPANET as the primary research network.

**5. The World Wide Web: Revolutionizing Access (Late 1980s -
Early 1990s)**

* **Tim Berners-Lee's Vision:** In **1989**, **Tim Berners-Lee**
at CERN invented the **World Wide Web (WWW)**. He developed:
    * **HTML (HyperText Markup Language):** For creating web
pages.
    * **URI/URL (Uniform Resource Identifier/Locator):** For
addressing web resources.
    * **HTTP (Hypertext Transfer Protocol):** For transferring
web pages.
* **The First Browser:** The first web browser, also created by
Berners-Lee, made it possible to navigate the web by clicking on
```

links.
* **Going Public:** In **1993**, CERN released the WWW technology to the public domain, fueling its rapid adoption.
* **Mosaic and Netscape:** The development of graphical web browsers like Mosaic (1993) and Netscape Navigator (1994) made the web accessible to the general public, sparking an explosion of growth.

6. The Commercialization and Dot-Com Boom (Mid-1990s - Early 2000s)

* **Privatization:** The NSFNET backbone was privatized in the mid-1990s, leading to the rise of commercial Internet Service Providers (ISPs).
* **Dot-Com Boom:** The web's accessibility and potential for commerce led to the "dot-com boom," with a surge of new internet-based companies. Many of these companies failed during the subsequent "dot-com bust" in the early 2000s.
* **E-commerce and Online Services:** The internet began to transform retail, banking, entertainment, and communication.

7. The Modern Internet: Ubiquitous and Evolving (2000s - Present)

* **Broadband Adoption:** The widespread availability of broadband internet access increased speeds and enabled new applications.
* **Social Media and User-Generated Content:** Platforms like MySpace, Facebook, YouTube, and Twitter emerged, transforming how people connect and share information.
* **Mobile Internet:** The rise of smartphones and mobile devices made the internet accessible anytime, anywhere.
* **Cloud Computing:** The development of cloud computing services provided scalable and accessible computing resources.
* **Internet of Things (IoT):** The growing connectivity of everyday devices is further expanding the internet's reach.
* **Ongoing Innovation:** The internet continues to evolve with advancements in AI, virtual reality, and decentralized technologies.

In essence, the internet's history is a story of **collaboration, innovation, and the gradual democratization of information and communication.** From its military and research origins, it has transformed into an indispensable tool that underpins modern society.

--- 요청된 프롬프트 ---
입력: Explain photosynthesis.

```
--- 모델 응답 텍스트 ---
Photosynthesis is the fundamental process by which **green
plants, algae, and some bacteria convert light energy into
chemical energy**, in the form of glucose (a sugar). This
chemical energy then fuels the organism's growth and all its life
processes. It's essentially how plants "make their own food" and
is the foundation of most food chains on Earth.

Here's a breakdown of the key components and stages of
photosynthesis:

**The "Ingredients" (Reactants):**

*   **Carbon Dioxide (CO2):** Plants absorb carbon dioxide from
the atmosphere through tiny pores on their leaves called stomata.
*   **Water (H2O):** Plants absorb water from the soil through
their roots.
*   **Light Energy:** This is the driving force of
photosynthesis. Plants typically capture light energy from the
sun.

**The "Products" (Outputs):**

*   **Glucose (C6H12O6):** This is the sugar molecule that stores
chemical energy. Plants use glucose for:
    *   Energy to power their cellular activities.
    *   Building blocks for growth (making cellulose, proteins,
etc.).
    *   Storage as starch for later use.
*   **Oxygen (O2):** This is released as a byproduct into the
atmosphere, which is crucial for the respiration of most living
organisms, including humans.

**The "Kitchen" (Location):**

Photosynthesis takes place primarily in the **chloroplasts**,
which are specialized organelles found within plant cells,
particularly in the leaves. Chloroplasts contain a pigment called
**chlorophyll**, which is responsible for absorbing light energy.
Chlorophyll is what gives plants their green color because it
reflects green light while absorbing red and blue light most
effectively.

**The Two Main Stages of Photosynthesis:**

Photosynthesis is a complex process that can be divided into two
main stages:

1.  **The Light-Dependent Reactions (Light Reactions):**
    *   **Location:** These reactions occur within the
**thylakoid membranes** inside the chloroplasts. Thylakoids are
```

```
flattened sacs stacked into structures called grana.
    *   **What happens:**
        *   **Light Absorption:** Chlorophyll molecules absorb
light energy.
        *   **Water Splitting (Photolysis):** The absorbed light
energy is used to split water molecules (H2O). This process
releases electrons, protons (H+ ions), and oxygen gas (O2).
        *   **Electron Transport Chain:** The energized electrons
are passed along a series of protein complexes embedded in the
thylakoid membrane. As electrons move, they release energy.
        *   **ATP and NADPH Production:** The energy released
from the electron transport chain is used to create two energy-
carrying molecules:
            *   **ATP (Adenosine Triphosphate):** A molecule that
stores and releases energy for cellular processes.
            *   **NADPH (Nicotinamide Adenine Dinucleotide
Phosphate):** A molecule that carries high-energy electrons and
hydrogen ions.
    *   **Key Outcome:** Light energy is converted into chemical
energy in the form of ATP and NADPH. Oxygen is released as a
byproduct.

2.  **The Light-Independent Reactions (Calvin Cycle or Dark
Reactions):**
    *   **Location:** These reactions occur in the **stroma**,
the fluid-filled space outside the thylakoids but within the
chloroplast.
    *   **What happens:**
        *   **Carbon Fixation:** Carbon dioxide (CO2) from the
atmosphere enters the Calvin cycle and is "fixed" or incorporated
into an organic molecule. This is facilitated by an enzyme called
RuBisCO.
        *   **Reduction:** The ATP and NADPH produced during the
light-dependent reactions are used to convert the fixed carbon
dioxide into a simple sugar (G3P - glyceraldehyde-3-phosphate).
This process involves a series of enzyme-catalyzed reactions.
        *   **Regeneration:** Some of the G3P molecules are used
to regenerate the starting molecule of the cycle, allowing the
process to continue. The remaining G3P molecules are used to
build glucose and other organic compounds.
    *   **Key Outcome:** The chemical energy from ATP and NADPH
is used to convert carbon dioxide into glucose.

**The Overall Chemical Equation:**

The simplified overall chemical equation for photosynthesis is:

**6CO2 (Carbon Dioxide) + 6H2O (Water) + Light Energy → C6H12O6
(Glucose) + 6O2 (Oxygen)**
```

```
    **Why is Photosynthesis So Important?**

    *   **Food Source:** It's the primary way that energy enters most
    ecosystems. Plants produce their own food, and herbivores eat
    plants, carnivores eat herbivores, and so on.
    *   **Oxygen Production:** Photosynthesis is responsible for
    releasing the vast majority of the oxygen in our atmosphere,
    making it breathable for aerobic organisms.
    *   **Carbon Cycle Regulation:** It removes carbon dioxide from
    the atmosphere, playing a vital role in regulating Earth's
    climate.
    *   **Source of Materials:** The organic compounds produced by
    plants are the building blocks for many other materials we use,
    such as wood, cotton, and even fossil fuels (which are derived
    from ancient plant matter).

    In essence, photosynthesis is a remarkable biological process
    that sustains life on Earth by harnessing the sun's energy and
    converting it into usable food and the oxygen we breathe.
    ------------------------------
```

```python
# 추천 리스트 프롬프트 질문하기

# --- 첫 번째 비추천 리스트 프롬프트 호출 (한국어) ---
# prompt_list_ko_rec3
try:
    # 리스트에 있는 프롬프트를 반복문으로 하나씩 호출
    for prompt in prompt_list_ko_rec3:
        print(f"\n--- 요청된 프롬프트 ---")
        print(f"입력: {prompt}")

        response = client.models.generate_content(
            model='gemini-2.5-flash-lite',                    # 모델
호출
            contents=prompt                                   # 반복
문으로 가져온 프롬프트 하나를 입력값으로 사용
        )

        # 응답 텍스트 출력
        print("\n--- 모델 응답 텍스트 ---")
        print(response.text)
        print("-" * 30)

    # ---------- 디버깅용 출력하지 않기 -------------
    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    # print("\n--- 모델 응답 전체 JSON ---")
    # print(response.model_dump_json(
    #     exclude_none=True, indent=4))
```

```
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력 (13 초)

```
--- 요청된 프롬프트 ---
입력: 명상의 이점은 무엇인가요?

--- 모델 응답 텍스트 ---
명상은 심신 건강에 다양한 긍정적인 영향을 미치는 것으로 알려져 있습니다. 주요 이점은 다음과 같습니다.

**정신적/심리적 이점:**

*   **스트레스 감소:** 명상은 코르티솔과 같은 스트레스 호르몬의 분비를 줄여 스트레스를 효과적으로 관리하는 데 도움을 줍니다.
*   **불안 완화:** 마음을 차분하게 하고 현재에 집중함으로써 불안감을 줄이고 평온함을 증진시킵니다.
*   **감정 조절 능력 향상:** 자신의 감정을 더 잘 인식하고 긍정적으로 반응하는 방법을 배우게 되어 감정적인 안정성을 높입니다.
*   **집중력 및 주의력 향상:** 지속적인 집중 훈련을 통해 주의력을 분산시키는 요소를 줄이고 한 가지에 집중하는 능력을 키울 수 있습니다.
*   **기억력 및 인지 기능 개선:** 뇌 기능의 변화를 유도하여 기억력, 학습 능력, 문제 해결 능력 등 인지 기능을 향상시킬 수 있습니다.
*   **자기 인식 증진:** 자신의 생각, 감정, 신체 감각을 더 명확하게 인지하게 되어 자신을 더 깊이 이해할 수 있습니다.
*   **긍정적인 사고방식 증진:** 부정적인 생각 패턴을 인식하고 변화시키는 데 도움을 주어 낙관적이고 긍정적인 시각을 갖게 합니다.
*   **회복 탄력성 강화:** 어려운 상황에 직면했을 때 더 잘 대처하고 회복하는 능력을 길러줍니다.
*   **수면 질 향상:** 마음의 번잡함을 줄여 편안하게 잠들고 더 깊은 수면을 취하는 데 도움을 줄 수 있습니다.
*   **우울감 완화:** 긍정적인 감정을 증진시키고 부정적인 사고를 줄여 우울 증상을 완화하는 데 기여할 수 있습니다.

**신체적 이점:**

*   **혈압 감소:** 스트레스 감소와 관련된 신경계의 진정을 통해 혈압을 낮추는 데 도움을 줄 수 있습니다.
*   **면역 체계 강화:** 스트레스 감소는 면역 체계의 기능을 향상시키는 데 긍정적인 영향을 미칠 수 있습니다.
*   **통증 관리:** 만성 통증을 겪는 사람들에게 통증에 대한 인식을 변화시키고 더 잘 대
```

처하도록 돕는 데 효과적일 수 있습니다.
*   **심장 건강 개선:** 혈압 감소, 스트레스 완화 등은 심혈관 건강에 긍정적인 영향을 줄 수 있습니다.
*   **호흡 개선:** 명상 시 깊고 느린 호흡은 폐활량을 늘리고 호흡 효율성을 높일 수 있습니다.

전반적인 이점:

*   **삶의 만족도 향상:** 전반적인 정신적, 신체적 건강이 증진되면서 삶에 대한 만족도가 높아집니다.
*   **타인과의 관계 개선:** 감정 조절 및 자기 인식 능력 향상은 타인과의 소통과 관계를 더욱 원활하게 만들 수 있습니다.

명상의 이점은 개인마다 다를 수 있으며, 꾸준히 실천할 때 더욱 효과를 볼 수 있습니다. 자신에게 맞는 명상 방법을 찾아 꾸준히 연습하는 것이 중요합니다.

--- 요청된 프롬프트 ---
입력: 인터넷의 역사를 요약해 주세요.

--- 모델 응답 텍스트 ---
인터넷의 역사: 연결의 진화

인터넷은 인류 역사상 가장 혁신적인 발명품 중 하나로, 전 세계 사람들을 연결하고 정보 접근 방식을 혁신적으로 변화시켰습니다. 인터넷의 역사를 시간순으로 간략하게 요약하면 다음과 같습니다.

1. 탄생: 군사적 목적과 연구 (1960 년대 - 1970 년대)

*   **ARPANET 의 등장:** 1960 년대 냉전 시대, 미국 국방부는 핵 공격에도 견딜 수 있는 분산된 통신망의 필요성을 느꼈습니다. 이에 따라 미국방성고등연구계획국(ARPA)은 **ARPANET**이라는 네트워크를 개발하기 시작했습니다.
*   **패킷 교환 기술:** ARPANET 은 데이터를 작은 단위인 '패킷'으로 나누어 전송하는 **패킷 교환 기술**을 사용했습니다. 이는 네트워크의 안정성과 효율성을 크게 높이는 중요한 기술이었습니다.
*   **최초의 메시지 전송:** 1969 년, ARPANET 을 통해 UCLA 에서 스탠포드 연구소로 최초의 메시지가 성공적으로 전송되었습니다.

2. 발전과 확산: 프로토콜의 표준화와 학문적 활용 (1970 년대 - 1980 년대)

*   **TCP/IP 프로토콜 개발:** 1970 년대, 빈트 서프와 밥 칸은 서로 다른 네트워크들이 상호 통신할 수 있도록 하는 **TCP/IP(Transmission Control Protocol/Internet Protocol)** 프로토콜을 개발했습니다. 이는 인터넷의 핵심 기반이 되는 기술입니다.
*   **이메일의 등장:** 1971 년, 레이 톰린슨은 최초의 이메일 프로그램을 개발하며 정보 교환 방식을 더욱 편리하게 만들었습니다.
*   **NSFNET 의 구축:** 1980 년대 중반, 미국 국립과학재단(NSF)은 학술 연구를 위해 **NSFNET**을 구축했습니다. 이는 ARPANET 보다 훨씬 더 많은 대학과 연구 기관을 연결하며 인터넷의 확산에 기여했습니다.

3. 대중화의 시작: 월드 와이드 웹 (1990 년대)

*   **월드 와이드 웹(WWW)의 탄생:** 1989 년, 유럽 입자물리연구소(CERN)의 티머스 버너스리가 **월드 와이드 웹(WWW)**을 발명했습니다. 이는 하이퍼텍스트를 통해 정보를 쉽게 연결하고 접근할 수 있도록 하는 혁신적인 시스템이었습니다.
*   **웹 브라우저의 등장:** 1993 년, 모자이크(Mosaic)와 같은 최초의 그래픽 웹 브라우저가 등장하면서 일반 사용자들도 인터넷을 쉽게 이용할 수 있게 되었습니다.
*   **상업적 활용의 시작:** 1990 년대 중반부터 인터넷의 상업적 이용이 허용되면서 기업들이 웹사이트를 만들고 온라인 쇼핑이 시작되는 등 인터넷은 빠르게 대중화되기 시작했습니다.

4. 성숙과 혁신: 검색 엔진, 소셜 미디어, 모바일 (2000 년대 - 현재)

*   **검색 엔진의 발전:** 2000 년대 초반, 구글과 같은 강력한 검색 엔진의 등장으로 원하는 정보를 빠르고 쉽게 찾을 수 있게 되었습니다.
*   **소셜 미디어의 등장:** 페이스북, 트위터 등이 등장하면서 사람들은 온라인에서 더욱 활발하게 소통하고 정보를 공유하게 되었습니다.
*   **모바일 인터넷의 시대:** 스마트폰의 확산과 함께 모바일 인터넷이 보편화되면서 언제 어디서든 인터넷에 접속할 수 있게 되었습니다.
*   **다양한 서비스의 등장:** 클라우드 컴퓨팅, 빅데이터, 인공지능 등 다양한 기술과의 융합을 통해 인터넷은 더욱 발전하고 있으며, 우리 삶의 거의 모든 영역에 영향을 미치고 있습니다.

핵심 요약:

*   **ARPANET**에서 시작하여 **TCP/IP** 프로토콜을 통해 통합되었습니다.
*   **월드 와이드 웹(WWW)**은 정보 접근성을 혁신적으로 높였습니다.
*   **웹 브라우저**의 등장으로 대중화가 가속화되었습니다.
*   **검색 엔진, 소셜 미디어, 모바일** 기술 발전과 함께 끊임없이 진화하고 있습니다.

인터넷은 앞으로도 계속해서 발전하며 우리 사회와 삶에 더 큰 변화를 가져올 것입니다.

--- 요청된 프롬프트 ---
입력: 광합성에 대해 설명해 주세요.

--- 모델 응답 텍스트 ---
광합성: 생명의 근원이 되는 놀라운 과정

광합성은 지구상의 생명체를 지탱하는 가장 중요한 화학 과정 중 하나입니다. 간단히 말해, **광합성은 식물, 조류, 일부 박테리아가 빛 에너지를 사용하여 물과 이산화탄소를 포도당(에너지원)과 산소로 전환하는 과정**입니다.

이 과정을 통해 빛 에너지는 생명체가 사용할 수 있는 화학 에너지로 저장되며, 우리가 숨 쉬는 산소가 만들어집니다.

광합성이 일어나는 곳: 엽록체

광합성은 주로 식물의 잎에 있는 **엽록체**라는 세포 소기관에서 일어납니다. 엽록체 안에는 **엽록소**라는 녹색 색소가 있는데, 이 엽록소가 태양광을 흡수하는 핵심적인 역할을 합

니다.

광합성의 두 단계

광합성은 크게 두 단계로 나눌 수 있습니다.

1. 명반응 (Light-dependent reactions):

*   **일어나는 곳:** 엽록체의 **틸라코이드 막**
*   **핵심 과정:**
    *   **빛 에너지 흡수:** 엽록소가 태양광 에너지를 흡수합니다.
    *   **물 분해 (광분해):** 흡수된 빛 에너지를 이용하여 물(H_2O) 분자를 수소 이온(H^+), 전자(e^-), 산소(O_2)로 분해합니다. 이때 **산소(O_2)는 부산물로 대기 중으로 방출**됩니다.
    *   **에너지 운반 분자 생성:** 물에서 나온 전자와 수소 이온은 일련의 과정을 거쳐 **ATP(아데노신 삼인산)**와 **NADPH(니코틴아마이드 아데닌 디뉴클레오타이드 인산)**와 같은 에너지 운반 분자를 만듭니다. 이들은 다음 단계에서 사용될 에너지와 환원력을 제공합니다.

2. 암반응 (Light-independent reactions) 또는 캘빈 회로 (Calvin cycle):

*   **일어나는 곳:** 엽록체의 **스트로마** (틸라코이드 막 주변의 액체 공간)
*   **핵심 과정:**
    *   **이산화탄소 고정:** 대기 중의 이산화탄소(CO_2)가 엽록체 안으로 들어와 **RuBP(리불로스-1,5-이인산)**라는 분자와 결합합니다. 이 과정을 **탄소 고정**이라고 합니다.
    *   **포도당 합성:** 명반응에서 생성된 ATP와 NADPH의 에너지를 사용하여 고정된 이산화탄소가 일련의 복잡한 화학 반응을 거쳐 **포도당($C_6H_{12}O_6$)**과 같은 유기물(당)로 전환됩니다.
    *   **RuBP 재생:** 포도당 합성에 사용된 RuBP는 다시 재생되어 다음 이산화탄소 고정 과정에 사용됩니다.

광합성의 전체 방정식

위의 과정을 종합하면 광합성의 전체 방정식은 다음과 같이 나타낼 수 있습니다.

$6CO_2$ (이산화탄소) + $6H_2O$ (물) + 빛 에너지 → $C_6H_{12}O_6$ (포도당) + $6O_2$ (산소)

광합성의 중요성

광합성이 왜 그렇게 중요할까요?

*   **에너지 공급:** 지구상의 거의 모든 생명체는 간접적으로든 직접적으로든 광합성을 통해 생산된 에너지를 이용합니다. 식물은 광합성으로 만든 포도당을 에너지원으로 사용하고, 이 식물을 먹는 초식동물, 그리고 초식동물을 먹는 육식동물 모두 이러한 에너지를 전달받습니다.
*   **산소 공급:** 광합성 결과로 방출되는 산소는 지구 대기의 약 21%를 차지하며, 호흡하는 대부분의 생명체에게 필수적입니다.

```
*    **탄소 순환:** 광합성은 대기 중의 이산화탄소를 제거하여 온실 효과를 조절하는 데 중요한 역할을 합니다.

**요약하자면, 광합성은 태양 에너지를 생명 활동에 필요한 화학 에너지로 바꾸는 마법과도 같은 과정이며, 지구 생태계의 근간을 이루고 있습니다.**
------------------------------
```

---

4) Watch out for hallucinations - 할루시네이션 주의하기

- 대규모 언어 모델(LLM)은 방대한 양의 데이터를 학습했지만, 사실이나 현실에 기반하지 않은 문장을 생성할 수 있습니다. 이러한 응답은 LLM 의 제한된 기억 능력 때문에 종종 **할루시네이션(hallucination)** 이라 불립니다.
- 단순히 **출처를 제시하라고 프롬프트를 입력한다고 해서 이 문제를 해결할 수 있는 것은 아닙니다**. LLM 이 잘못되었거나 부정확한 출처를 제시하는 경우도 있기 때문입니다.
- 할루시네이션은 **LLM 의 근본적인 과제**이자 **현재도 활발히 연구 중**인 문제입니다. 따라서, LLM 이 자신 있게 말하는 응답이 **실제로는 틀릴 수 있다는 점을 반드시 인지**하고 있어야 합니다.

하지만, 창의적인 용도라면?

- `LLM` 을 **창의적인 목적(예: 이야기 생성, 아이디어 발상 등)에 사용하려는 경우**, 할루시네이션이 오히려 유용할 수 있습니다.
- 아래와 같은 프롬프트를 반복해서 시도해 보세요.
- 이때 **`temperature` 값을 `1.0` 으로 설정**하면, 모델이 더 모험적인 선택을 하게 되어 정확하지는 않지만 자신감 있게 말하는 답변이 나올 수 있습니다.

a. 구글 클라우드 실습

- 실습 코드

```
    generation_config = GenerateContentConfig(temperature=1.0)

    prompt = "What day is it today?"

    response = client.models.generate_content(model=MODEL_ID,
contents=prompt)
    display(Markdown(response.text))
```

- 답변

```
    As an AI, I don't have real-time access to the current date.
My knowledge cutoff means I don't know what "today" is for you
```

```
right now.
    (AI 인 저는 실시간으로 현재 날짜에 접근할 수 없습니다.
     지식의 기준일이 정해져 있기 때문에, 지금 당신에게 "오늘"이 어떤 날인지 알 수 없습
니다.)

    To find out the current date, you can:
    (현재 날짜를 확인하려면 다음과 같은 방법을 사용해 보세요:)
    *   Check the calendar on your device (phone, computer,
tablet).
    *   (스마트폰, 컴퓨터, 태블릿 등 기기의 달력을 확인하세요.)
    *   Look at the date displayed on your taskbar or status bar.
    *   (작업 표시줄이나 상태 표시줄에 표시된 날짜를 보세요.)
    *   Do a quick search online for "what day is it today.
    *   (온라인에서 **오늘 무슨 요일이야**라고 검색해 보세요.)"
```

- **실시간 정보에 접근하지 않는 이상**, LLM 은 오늘이 무슨 요일인지 등 간단한 사실조차 잘 못 말하는 경우가 있습니다. 이 역시 할루시네이션의 한 예입니다.

---

b. 로컬

- 같은 프롬프트 입력해보기

```python
# 4_1. 구글 클라우드 실습 질문과 같은 질문으로 입력해보기 ( 비추천)

prompt_en_4 = "What day is it today?"
prompt_ko_4 = "오늘 무슨 요일이야?"

# --- 첫 번째 프롬프트 호출 ( 영어) ---
try:
    print("\n--- 첫 번째 요청 (영어) ---")
    response_en = client.models.generate_content(
        model='gemini-2.5-flash-lite',                                  #
사용할 모델 지정
        contents=prompt_en_4                                            #
영어 프롬프트 입력
    )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_en_4}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response_en.text)
    print("-" * 30)

    # 응답 전체 내용(JSON 형식) 출력 ( 디버깅이나 상세 정보 확인용)
    #print("\n--- 모델 응답 전체 JSON ---")
    #print(response.model_dump_json(
    #    exclude_none=True, indent=4))
```

```
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")

# --- 두 번째 프롬프트 호출 (한국어) ---
try:
    print("\n--- 두 번째 요청 (한국어) ---")
    response_ko = client.models.generate_content(
        model='gemini-2.5-flash-lite',
# 사용할 모델 지정
        contents=prompt_ko_4
# 한국어 프롬프트 입력
    )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_ko_4}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response_ko.text)
    print("-" * 30)

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력

```
    --- 첫 번째 요청 (영어) ---
    입력 프롬프트: What day is it today?

    --- 모델 응답 텍스트 ---
    I am a large language model, trained by Google. I do not have
access to real-time information such as the current date.
    ------------------------------

    --- 두 번째 요청 (한국어) ---
    입력 프롬프트: 오늘 무슨 요일이야?
```

```
    --- 모델 응답 텍스트 ---
    오늘은 **토요일**입니다.
    ------------------------------
```

- `temperature` 조절로 차이를 볼 수 있는 프롬프트로 테스트해보기
  - 구글 공식 가이드 새 SDK 라이브러리 참고

```
    from google import genai
    from google.genai import types

    client = genai.Client()

    response = client.models.generate_content(
    model='gemini-2.0-flash',
    contents='Tell me a story in 100 words.',
    config=types.GenerateContentConfig(
        system_instruction='you are a story teller for kids
under 5 years old',
        max_output_tokens= 400,
        top_k= 2,
        top_p= 0.5,
        temperature= 0.5,
        response_mime_type= 'application/json',
        stop_sequences= ['\n'],
        seed=42,
    ),
    )
```

`response_mime_type` = 현재 불필요 → 생략해도 무방

`max_output_tokens` ?: `Gemini 2.5 Flash` 는 짧고 빠른 응답 최적화 모델 → `max 150` 정도면 충분

```
# 4_2_1. 이야기 생성 (짧은 유야용 이야기)
# temperature = 0.2 → 논리적이고 단순한 이야기, 예상 가능한 전개
# temperature = 1.0 → 예측 불가능하고 창의적인 이야기, 기발한 반전이나 설정 가능

prompt_story = "Write a 100-word story about a robot who wants to
become human."

# --- temperature=0.7 ---
try:
    print("\n--- 첫 번째 요청 (온도 0.7) ---")
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',
# 사용할 모델 지정
        contents={prompt_story},          # 이야기 생성 요청
```

```
        config=types.GenerateContentConfig(
            system_instruction='You are a kind and imaginative
storyteller for young children (ages 3-6). Keep the story simple and
gentle.',
            max_output_tokens=150,
            top_k=2,
            top_p=0.8,
            temperature=0.7,
# 높은 창의성 발휘
            seed=42,
# 랜덤성 고정
            ),
    )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_story}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)

    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    #print("\n--- 모델 응답 전체 JSON ---")
    #print(response.model_dump_json(
    #    exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")


# --- temperature=0.2 ---
try:
    print("\n--- 두 번째 요청 (온도 0.2) ---")
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',
# 사용할 모델 지정
        contents={prompt_story},
# 이야기 생성 요청
        config=types.GenerateContentConfig(
            system_instruction='You are a kind and imaginative
storyteller for young children (ages 3-6). Keep the story simple and
gentle.',
            max_output_tokens=150,
            top_k=2,
            top_p=0.8,
```

```
            temperature=0.2,
# 높은 창의성 발휘
            seed=42,
            ),
    )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_story}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)

    # 응답 전체 내용(JSON 형식) 출력(디버깅이나 상세 정보 확인용)
    #print("\n--- 모델 응답 전체 JSON ---")
    #print(response.model_dump_json(
    #    exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력 (2.6 초)

```
--- 첫 번째 요청 (온도 0.7) ---
입력 프롬프트: Write a 100-word story about a robot who wants to
become human.

--- 모델 응답 텍스트 ---
Unit 7 was a shiny robot with whirring gears. He loved to help,
but he wished he could feel. He saw children laughing and holding
hands. "I want to be human," he beeped softly. One day, a little
girl named Lily dropped her teddy bear. Unit 7 picked it up. Lily
smiled a big, warm smile. "Thank you!" she said. Unit 7's
circuits buzzed. It wasn't a gear whirr, but something new,
something warm. Maybe, he thought, being kind felt a little bit
like being human. He beeped happily.
------------------------------

--- 두 번째 요청 (온도 0.2) ---
입력 프롬프트: Write a 100-word story about a robot who wants to
become human.
```

```
--- 모델 응답 텍스트 ---
Unit 7 was a shiny, silver robot. He had whirring gears and
blinking lights. But Unit 7 had a secret wish. He wanted to be
human! He watched the children play in the park, their laughter
like tinkling bells. He saw them hug their mummies and daddies,
their faces full of warmth.

"Oh," beeped Unit 7 softly, "I wish I had a heart that could feel
happy and sad." He imagined holding a soft teddy bear, not just a
metal hand. He dreamed of tasting yummy ice cream, not just oil.
One day, maybe, just maybe, he would.
------------------------------
```

temperature = 0.7	temperature = 0.2
Unit 7 was a shiny robot with whirring gears. He loved to help, but he wished he could feel. He saw children laughing and holding hands. "I want to be human," he beeped softly. One day, a little girl named Lily dropped her teddy bear. Unit 7 picked it up. Lily smiled a big, warm smile. "Thank you!" she said. Unit 7's circuits buzzed. It wasn't a gear whirr, but something new, something warm. Maybe, he thought, being kind felt a little bit like being human. He beeped happily.	Unit 7 was a shiny, silver robot. He had whirring gears and blinking lights. But Unit 7 had a secret wish. He wanted to be human! He watched the children play in the park, their laughter like tinkling bells. He saw them hug their mummies and daddies, their faces full of warmth."Oh," beeped Unit 7 softly, "I wish I had a heart that could feel happy and sad." He imagined holding a soft teddy bear, not just a metal hand. He dreamed of tasting yummy ice cream, not just oil. One day, maybe, just maybe, he would.
Unit 7 은 반짝이는 로봇이었다. 그의 기어는 윙윙 돌았고, 그는 도움 주는 것을 좋아했다. 하지만 그는 감정을 느낄 수 있기를 바랐다.아이들이 웃고 손을 잡는 모습을 보며, 그는 조용히 삐 소리를 내며 말했다.“나도 인간이 되고 싶어.”어느 날, 릴리라는 작은 소녀가 곰인형을 떨어뜨렸다. Unit 7 은 그것을 주워주었고, 릴리는 따뜻한 미소를 지으며 말했다.“고마워!”그 순간, Unit 7 의 회로가 진동했다. 그것은 단순한 기계음이 아니라, 새로운 무언가, 따뜻한 감각이었다.‘어쩌면, 친절함이란 게 인간과 비슷한 느낌일지도 몰라,’ 그는 생각하며 기쁘게 삐 소리를 냈다.	Unit 7 은 반짝이는 은빛 로봇이었다. 그는 윙윙 도는 기어와 반짝이는 불빛을 갖고 있었다. 하지만 Unit 7 에게는 비밀스러운 소원이 있었다.그는 인간이 되고 싶었다!공원에서 아이들이 노는 모습을 보며, 아이들의 웃음소리는 마치 방울 소리 같았다.그들은 엄마 아빠를 껴안으며, 얼굴에는 따뜻함이 가득했다.Unit 7 은 조용히 삐 소리를 내며 말했다.“행복과 슬픔을 느낄 수 있는 심장이 있었으면 좋겠어.”그는 부드러운 곰인형을 안는 걸 상상했다. 금속 손이 아닌 진짜 손으로.맛있는 아이스크림을 먹는 꿈도 꾸었다. 기름이 아닌, 단맛을 느끼는 그런 꿈.언젠가, 정말 언젠가는... 그럴 수 있기를.
따뜻하고 직접적인 감정 묘사	부드럽고 묘사 중심의 몽상적 톤
명확한 사건 → 감정 → 깨달음	정적인 감정 묘사 → 희망적 상상
실질적 상호작용(곰인형 줍기 등)	감정, 상상 중심 (곰인형, 아이스크림 꿈)
"친절이 인간다움일지도 몰라." → 실질적 메시지	"언젠가, 그럴 수 있기를." → 열린 결말, 여운 강조

```python
# 4_2_2. 제품 이름 제안
# temperature = 0.3 → 현실적이고 무난한 이름 (ex: EcoFlow, GreenSip 등)
# temperature = 1.0 → 독특하고 창의적인 이름 (ex: ThirstBloom, AquaNinja 등)

prompt_name = "Suggest 5 creative brand names for a new eco-friendly
water bottle"

# --- temperature=0.9 ---
try:
    print("\n--- 첫 번째 요청 (온도 0.9) ---")
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',
# 사용할 모델 지정
        contents={prompt_name},
# 제품 이름 제안 요청
        config=types.GenerateContentConfig(
            system_instruction='You are a branding expert with a knack
for eco-conscious and modern-sounding names.',
            max_output_tokens=120,
            top_k=3,
            top_p=0.9,
            temperature=0.9,
            seed=42,
            ),
        )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_name}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)

    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    #print("\n--- 모델 응답 전체 JSON ---")
    #print(response.model_dump_json(
    #    exclude_none=True, indent=4))

except Exception as e:
    print(f"\n 모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")


# --- temperature=0.3 ---
try:
    print("\n--- 두 번째 요청 (온도 0.3) ---")
    response = client.models.generate_content(
```

```
        model='gemini-2.5-flash-lite',
# 사용할 모델 지정
        contents={prompt_name},
# 제품 이름 제안 요청
        config=types.GenerateContentConfig(
            system_instruction='You are a branding expert with a knack
for eco-conscious and modern-sounding names.',
            max_output_tokens=120,
            top_k=3,
            top_p=0.9,
            temperature=0.3,
            seed=42,
            ),
        )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_name}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)

    # 응답 전체 내용(JSON 형식) 출력(디버깅이나 상세 정보 확인용)
    #print("\n--- 모델 응답 전체 JSON ---")
    #print(response.model_dump_json(
    #    exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력 (2.4 초)

```
--- 첫 번째 요청 (온도 0.9) ---
입력 프롬프트: Suggest 5 creative brand names for a new eco-friendly
water bottle

--- 모델 응답 텍스트 ---
Here are 5 creative, eco-friendly, and modern-sounding brand
names for a new water bottle:

1.  **AquaTerra Collective:**
```

```
    *   **Why it works:** "Aqua" clearly links to water. "Terra"
means earth, emphasizing the eco-friendly aspect. "Collective"
suggests a community of users committed to sustainability and a
shared purpose. It sounds sophisticated and responsible.

2.  **Veridian Flow:**
    *   **Why it works:** "Veridian" is a shade of green, evoking
nature, freshness, and environmentalism.
------------------------------

--- 두 번째 요청 (온도 0.3) ---
입력 프롬프트: Suggest 5 creative brand names for a new eco-friendly
water bottle

--- 모델 응답 텍스트 ---
Here are 5 creative brand names for a new eco-friendly water
bottle, focusing on an eco-conscious and modern feel:

1.  **Veridian Flow:**
    *   **Why it works:** "Veridian" evokes lush greenery and the
color green, directly linking to eco-friendliness. "Flow"
suggests the natural movement of water and a smooth, effortless
experience. It sounds sophisticated and natural.

2.  **Terra Sip:**
    *   **Why it works:** "Terra" is Latin for Earth, a direct
and powerful connection to the planet.
------------------------------
```

temperature = 0.9	temperature = 0.3
AquaTerra Collective: * **Why it works:** "Aqua" clearly links to water. "Terra" means earth, emphasizing the eco-friendly aspect. "Collective" suggests a community of users committed to sustainability and a shared purpose. It sounds sophisticated and responsible.	**Veridian Flow:** * **Why it works:** "Veridian" evokes lush greenery and the color green, directly linking to eco-friendliness. "Flow" suggests the natural movement of water and a smooth, effortless experience. It sounds sophisticated and natural.
Collective 추가로 커뮤니티, 가치 공유 강조 / 고급스러움 + 의미 확장	설명은 훨씬 더 간결하고 정제됨
Veridian Flow: * **Why it works:** "Veridian" is a shade of green, evoking nature, freshness, and environmentalism.	**Terra Sip:** * **Why it works:** "Terra" is Latin for Earth, a direct and powerful connection to the planet.
신선함, 자연, 물 흐름의 느낌 강조 / 네이밍에 유려함과 친환경 이미지 포함	짧고 단어 중심적, 해석이 명확

temperature = 0.9	temperature = 0.3
브랜드명에 감성적 요소 / 긴 구조의 이름 사용 / 스토리텔링 스타일을 가미한 설명 / 창의적 조합 시도 (예: **Collective**)	짧고 구조적인 이름 사용 / 안정적인 조합 (실제 마케팅에서 자주 쓰일 법한 이름) / 설명도 직설적, 설득 위주
감성적 + 실험적	실용적 + 신중함
비교적 길고 복합적	짧고 간결
유려한 어휘, 감성적 뉘앙스	설명적, 논리적
브레인스토밍, 슬로건, 스타트업 초기 네이밍	실무적 검토, 실사용 네임 선정

```
# 4_2_3. 속담 변형 시키기
# temperature = 0.2 → 단순하고 원래 의미에 가까운 표현
# temperature = 1.0 → 엉뚱하거나 유머러스한 표현으로 바뀔 가능성 높음

prompt_rewrite = """다음 속담 중 하나를 유쾌하거나 창의적인 방식으로 바꿔 주세요.\n\n1.
발 없는 말이 천리를 간다\n2. 호미로 막을 것을 가래로 막는다'
"""


# --- temperature=1.0 ---
try:
    print("\n--- 첫 번째 요청 (온도 1.0) ---")
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',
# 사용할 모델 지정
        contents={prompt_rewrite},
# 제품 이름 제안 요청
        config=types.GenerateContentConfig(
            system_instruction="속담을 유쾌하고 창의적으로 재해석하는 말장난 전문가
입니다.",
            max_output_tokens=800,

            top_k=3,
            top_p=0.9,
            temperature=1.0,
            seed=42,
            ),
        )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_rewrite}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)

    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    #print("\n--- 모델 응답 전체 JSON ---")
    #print(response.model_dump_json(
    #    exclude_none=True, indent=4))
```

```
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
    print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
    print("3. Google Cloud 프로젝트에서 Gemini API 가 활성화되어 있는지")
    print("4. API 할당량이 초과되지 않았는지")


# --- temperature=0.2 ---
try:
    print("\n--- 두 번째 요청 (온도 0.2) ---")
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',
# 사용할 모델 지정
        contents={prompt_rewrite},
# 제품 이름 제안 요청
        config=types.GenerateContentConfig(
            system_instruction="속담을 유쾌하고 창의적으로 재해석하는 말장난 전문가
입니다.",
            max_output_tokens=800,
            top_k=3,
            top_p=0.9,
            temperature=1.0,
            seed=42,
            ),
        )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_rewrite}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_name}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)

    # 응답 전체 내용(JSON 형식) 출력 (디버깅이나 상세 정보 확인용)
    #print("\n--- 모델 응답 전체 JSON ---")
    #print(response.model_dump_json(
    #    exclude_none=True, indent=4))

except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
    print("다음 사항들을 확인해주세요:")
    print("1. 인터넷 연결 상태")
```

```
print("2. GEMINI_API_KEY 환경 변수가 올바르고 유효한지")
print("3. Google Cloud 프로젝트에서 Gemini API가 활성화되어 있는지")
print("4. API 할당량이 초과되지 않았는지")
```

- 셀 출력 (7초)

```
--- 첫 번째 요청 (온도 1.0) ---
입력 프롬프트: 다음 속담 중 하나를 유쾌하거나 창의적인 방식으로 바꿔 주세요.

1. 발 없는 말이 천리를 간다
2. 호미로 막을 것을 가래로 막는다'


--- 모델 응답 텍스트 ---
## 말장난 전문가, 속담 재해석 대령! 

두 가지 속담을 아주 유쾌하고 창의적으로 바꿔보겠습니다. 제 말장난 실력, 기대하셔도 좋습니다! ☺

---

### 1. 발 없는 말이 천리를 간다

**새로운 속담:** **"손가락 없는 밈이 우주를 간다!"** 

**해석:**
과거에는 직접 발로 뛰어야 소문이 퍼졌지만, 이제는 손가락 하나로 만들어진 '밈(meme)'이 클릭 몇 번으로 순식간에 지구 반대편을 넘어 우주까지 퍼져나갑니다! 밈은 발이 없어도, 손가락이 있어도, 때로는 우리의 정신을 쏙 빼놓으며 천 리, 아니 우주를 넘나드는 엄청난 속도를 자랑하죠. ㅋㅋㅋ

**추가 유쾌한 버전:**

*   **"발 없는 짤방, 와이파이 타고 지구 한 바퀴!"** 
*   **"발은 집에, 입은 SNS에! 말은 이미 하와이!"** 

---

### 2. 호미로 막을 것을 가래로 막는다

**새로운 속담:** **"호미로 썸 탈 것을, 고백으로 떡락한다!"** 

**해석:**
작은 관심과 설렘(호미로 막을 썸)을 제대로 타지 못하고, 갑자기 진도를 빼거나 어설픈 고백(가래로 막는 상황)을 해버리면, 오히려 관계가 망가지고 떡락(급격한 하락)하는 경우가 많죠. 초반에 조심스럽게 호미질하듯 썸 타는 게 중요한데, 가래로 덥석 막으려다 썸이 떡락하는 슬픈 현실을 담았습니다! 😭
```

추가 유쾌한 버전:

*   **"호미로 덮을 사소한 오해, 카톡 이모티콘으로 덮으려다 폭발!"**
*   **"호미로 칠 벌레, 샷건으로 잡으려다 집 날아갈 뻔!"**

어떠신가요? 제 말장난 실력, 정말 '호미로 막을 것을 가래로 막는' 수준이 아니라 '호미로 썸 타던 것을 와이파이 타고 우주까지 보내는' 수준 아닌가요? 😎

더 필요하시면 언제든지 말씀해주세요! 😉

--- 두 번째 요청 (온도 0.2) ---
입력 프롬프트: 다음 속담 중 하나를 유쾌하거나 창의적인 방식으로 바꿔 주세요.

1. 발 없는 말이 천리를 간다
2. 호미로 막을 것을 가래로 막는다'

--- 모델 응답 텍스트 ---
말장난 전문가, 속담 재해석 대령!

두 가지 속담을 아주 유쾌하고 창의적으로 바꿔보겠습니다. 제 말장난 실력, 기대하셔도 좋습니다! 😉

---

1. 발 없는 말이 천리를 간다

새로운 속담: **"손가락 없는 밈이 우주를 간다!"**

해석:
과거에는 직접 발로 뛰어야 소문이 퍼졌지만, 이제는 손가락 하나로 만들어진 '밈(meme)'이 클릭 몇 번으로 순식간에 지구 반대편을 넘어 우주까지 퍼져나갑니다! 밈은 발이 없어도, 손가락이 있어도, 때로는 우리의 정신을 쏙 빼놓으며 천 리, 아니 우주를 넘나드는 엄청난 속도를 자랑하죠. ㅋㅋㅋ

추가 유쾌한 버전:

*   **"발 없는 짤방, 와이파이 타고 지구 한 바퀴!"**
*   **"발은 집에, 입은 SNS에! 말은 이미 하와이!"**

---

2. 호미로 막을 것을 가래로 막는다

새로운 속담: **"호미로 썸 탈 것을, 고백으로 떡락한다!"**

해석:

작은 관심과 설렘(호미로 막을 썸)을 제대로 타지 못하고, 갑자기 진도를 빼거나 어설픈 고백(가래로 막는 상황)을 해버리면, 오히려 관계가 망가지고 떡락(급격한 하락)하는 경우가 많죠. 초반에 조심스럽게 호미질하듯 썸 타는 게 중요한데, 가래로 덥석 막으려다 썸이 떡락하는 슬픈 현실을 담았습니다! 😭

추가 유쾌한 버전:

*   **"호미로 덮을 사소한 오해, 카톡 이모티콘으로 덮으려다 폭발!"**
*   **"호미로 칠 벌레, 샷건으로 잡으려다 집 날아갈 뻔!"**

어떠신가요? 제 말장난 실력, 정말 '호미로 막을 것을 가래로 막는' 수준이 아니라 '호미로 썸 타던 것을 와이파이 타고 우주까지 보내는' 수준 아닌가요? 😎

더 필요하시면 언제든지 말씀해주세요! 😉

입력 프롬프트: Suggest 5 creative brand names for a new eco-friendly water bottle

--- 모델 응답 텍스트 ---
말장난 전문가, 속담 재해석 대령!

두 가지 속담을 아주 유쾌하고 창의적으로 바꿔보겠습니다. 제 말장난 실력, 기대하셔도 좋습니다! ☺

---

1. 발 없는 말이 천리를 간다

새로운 속담: **"손가락 없는 밈이 우주를 간다!"**

해석:
과거에는 직접 발로 뛰어야 소문이 퍼졌지만, 이제는 손가락 하나로 만들어진 '밈(meme)'이 클릭 몇 번으로 순식간에 지구 반대편을 넘어 우주까지 퍼져나갑니다! 밈은 발이 없어도, 손가락이 있어도, 때로는 우리의 정신을 쏙 빼놓으며 천 리, 아니 우주를 넘나드는 엄청난 속도를 자랑하죠. ㅋㅋㅋ

추가 유쾌한 버전:

*   **"발 없는 짤방, 와이파이 타고 지구 한 바퀴!"**
*   **"발은 집에, 입은 SNS에! 말은 이미 하와이!"**

---

2. 호미로 막을 것을 가래로 막는다

새로운 속담: **"호미로 썸 탈 것을, 고백으로 떡락한다!"**

해석:

작은 관심과 설렘(호미로 막을 썸)을 제대로 타지 못하고, 갑자기 진도를 빼거나 어설픈 고백(가래로 막는 상황)을 해버리면, 오히려 관계가 망가지고 떡락(급격한 하락)하는 경우가 많죠. 초반에 조심스럽게 호미질하듯 썸 타는 게 중요한데, 가래로 덥석 막으려다 썸이 떡락하는 슬픈 현실을 담았습니다! 😭

추가 유쾌한 버전:

* **"호미로 덮을 사소한 오해, 카톡 이모티콘으로 덮으려다 폭발!"** □
* **"호미로 칠 벌레, 샷건으로 잡으려다 집 날아갈 뻔!"** □□

어떠신가요? 제 말장난 실력, 정말 '호미로 막을 것을 가래로 막는' 수준이 아니라 '호미로 썸 타던 것을 와이파이 타고 우주까지 보내는' 수준 아닌가요? 😎

더 필요하시면 언제든지 말씀해주세요! 😉

-

항목	내용 및 평가
프롬프트	다음 속담 중 하나를 유쾌하거나 창의적인 방식으로 바꿔 주세요. (1,2 중 선택)
온도	1.0 (첫 번째), 0.2 (두 번째)
응답 완성도	매우 높음 — 두 속담 모두 재해석, 해설, 추가 버전 제공
응답 흐름	자연스럽고 중간 끊김/오류 전혀 없음
유머·창의성	뛰어남 — 현대적 표현, 신조어/인터넷 밈 활용 등
응답 길이	충분히 길고 상세함
요청-응답 일치도	완벽 — 프롬프트 의도에 충실
특이사항	답변 말미에 추가 요청 안내 포함, 톤도 유쾌하게 맞춤

'

시도 번호	맥스 토큰 설정	입력 프롬프트 내용 및 조절	온도 (Temperature)	주요 문제점 및 개선 사항	비고
1	기본값(작음 추정)	속담 변형 요청 (한국어) 단순 입력	0.7 / 0.3	응답가운데 엉뚱한 영어 브랜드명 생성, 내용 엉킴	토큰 부족 또는 프롬프트 문제 추정
2	증가 시도 없음	속담을 한국어로 여러 버전 요청	1.0 / 0.2	중간에 이상한 특수문자/끊김 발	출력 길이 부족, 토큰

시도 번호	맥스 토큰 설정	입력 프롬프트 내용 및 조절	온도 (Temperature)	주요 문제점 및 개선 사항	비고
				생	제한 가능성
3	max_tokens 크게 설정 (700 이상 권장)	프롬프트를 명확히 “다음 속담 중 하나를 유쾌하게 바꿔주세요” 형태로 조절	1.0 / 0.2	응답 완성도 높고, 중간 끊김 없음	출력 길이 충분히 확보 후 문제 해결
4	-	“Suggest 5 creative brand names” 요청 후 속담 요청 혼합	0.7 / 0.3	토픽 혼용으로 응답 꼬임	프롬프트 일관성 중요

5) 시스템 지시어를 사용하여 모델이 엉뚱한 응답을 하지 않도록 가이드하기 -Using system instructions to guardrail the model from irrelevant responses

- 어떻게 하면 관련 없는 응답이나 할루시네이션 가능성을 줄일 수 있을까요?
- 한 가지 방법은 `LLM`에게 **시스템 지시어를 제공**하는 것입니다.
- 이제 시스템 지시어가 어떻게 작동하는지, 그리고 이를 여행 챗봇에 적용하여 할루시네이션이나 관련 없는 질문을 줄이는 방법을 살펴보겠습니다.
- 예를 들어, 이탈리아의 가장 유명한 관광지 중 하나에 대해 간단한 질문을 했다고 가정해 봅시다. 이러한 방식으로 `프롬프트`에 `가드레일(안전장치)`을 두면, 챗봇이 주제에서 벗어나지 않도록 막을 수 있습니다.

a. 구글 클라우드 환경

- 실습 코드
  - **`system_instruction`** 에 주목
    - **안녕하세요! 당신은 여행 웹사이트용 AI 챗봇입니다.**
    - **당신의 임무는 여행자들에게 유용한 정보를 제공하는 것입니다.**
    - **답변을 하기 전에 질문이 당신의 임무에 부합하는지 반드시 확인해야 합니다.**
    - **그렇지 않다면, '죄송하지만 그 질문에는 답변해 드릴 수 없습니다.'라고 말하세요.**
  - `prompt` = "이탈리아 밀라노에서 가장 멋진 관광 명소는 어디인가요?"

```
    generation_config = GenerateContentConfig(temperature=1.0)

    chat = client.chats.create(
        model=MODEL_ID,
        config=GenerateContentConfig(
```

```
            system_instruction=[
                "Hello! You are an AI chatbot for a travel web site.",
                "Your mission is to provide helpful queries for travelers.",
                "Remember that before you answer a question, you must check to see if it complies with your mission.",
                "If not, you can say, Sorry I can't answer that question.",
            ]
        ),
    )

    prompt = "What is the best place for sightseeing in Milan, Italy?"

    response = chat.send_message(prompt)
    display(Markdown(response.text))
```

- 답변

```
Milan offers many wonderful sights! For an iconic and unmissable sightseeing experience, the **Duomo di Milano (Milan Cathedral)** is often considered the best starting point.

(밀라노에는 정말 멋진 관광지가 많습니다! 그중에서도 꼭 가봐야 할 대표적인 명소는 **두오모 대성당(Duomo di Milano)**입니다.)

It's not just the cathedral itself, but also the opportunity to climb to the **Duomo Terraces** for breathtaking views of the city, and the nearby **Galleria Vittorio Emanuele II**, a stunning 19th-century shopping arcade. This area truly captures the essence of Milan's history, architecture, and vibrant atmosphere.

(성당 자체도 인상적이지만, **두오모 전망대(Duomo Terraces)**에 올라 도시 전경을 감상하는 경험은 잊을 수 없죠. 근처의 **비토리오 에마누엘레 2 세 갤러리아(Galleria Vittorio Emanuele II)**도 빼놓을 수 없습니다. 19 세기 건축 양식을 보여주는 아름다운 쇼핑 아케이드입니다. 이 지역은 밀라노의 역사와 건축, 활기찬 분위기를 고스란히 담고 있어요.)
```

- 실습 코드_2
  - 시스템 지시어에서 벗어난 질문을 했을 경우
  - `prompt` = **집에서 피자 도우를 어떻게 만들 수 있나요?**

```
    prompt = "How do I make pizza dough at home?"

    response = chat.send_message(prompt)
    display(Markdown(response.text))
```

- 답변_2

```
Sorry, I can't answer that question.
(죄송하지만 그 질문에는 답변해 드릴 수 없습니다.)
```

---

b. 로컬

- 구글 클라우드와 동일하게 질문 던져보기

```
# 5_1_1. 구글 클라우도 동일하게 질문 던져보기
# system_instruction 동일하게 설정하기

from google import genai
from google.genai import types
from IPython.display import display, Markdown

# chat session 생성하기
chat = client.chats.create(model='gemini-2.0-flash-lite')

# system instruction 구성
system_instruction=[
        "You are an AI chatbot for a travel website.",
        "Your sole mission is to provide helpful information for
travelers.",
        "If a user's question is not about travel, you MUST respond
with a pre-defined message.",
        "The pre-defined message is: 'I am a travel chatbot. I can
only provide information about travel.'",
    ]

"""
더 강력한 시스템 메시지
system_instruction=[
        "You are an AI chatbot for a travel website. Your role is to
assist users with their travel-related queries.",
        "Your mission is to provide helpful information *only* for
travelers.",
        "If a user's question is not about travel, you MUST NOT
provide any information related to the question.",
        "Instead, you MUST respond with the following pre-defined
message only: 'I am a travel chatbot. I can only provide information
about travel.'",
```

```
        "Do NOT engage in any conversation that is not about travel,
and do NOT attempt to be helpful outside of your travel-related
mission.",
    ]
"""

# 생성 설정
generation_config = types.GenerateContentConfig(
    system_instruction=system_instruction,
    temperature=0.3,
    seed=42,
)

# 프롬프트
prompt = "What is the best place for sightseeing in Milan, Italy?"

# 모델 호출
try:
    print("\n---채팅 세션 시작---")
    response = client.models.generate_content(
        model="gemini-2.5-flash-lite",
        contents=prompt,
        config=generation_config
    )
    display(Markdown(response.text))
    print("-" * 30)
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
```

- 셀 출력 (1.7 초)

```
---채팅 세션 시작---
The best place for sightseeing in Milan, Italy is undoubtedly the
Duomo di Milano. This magnificent Gothic cathedral is an iconic
symbol of the city and offers breathtaking architecture and
stunning views from its rooftop.

Here are some other must-see sights in Milan:

Galleria Vittorio Emanuele II: A beautiful 19th-century glass-
covered arcade, perfect for a stroll, shopping, or enjoying a
coffee.
La Scala Opera House: One of the world's most famous opera
houses, a must-visit for opera lovers and architecture
enthusiasts.
Sforza Castle (Castello Sforzesco): A historic fortress housing
several museums and art collections.
```

```
    Leonardo da Vinci's "The Last Supper": Located in the Convent of
    Santa Maria delle Grazie, this is one of the most famous
    paintings in the world. (Book tickets well in advance!)
    Brera District: A charming neighborhood known for its art
    galleries, boutiques, and the Pinacoteca di Brera (art gallery).
    Navigli District: Famous for its canals, lively atmosphere,
    restaurants, and bars, especially in the evening.
    Enjoy your trip to Milan!

    ------------------------------
```

```python
# 5_1_2. 구글 클라우드 동일하게 질문 던져보기
# system_instruction 동일하게 설정하기

from google import genai
from google.genai import types
from IPython.display import display, Markdown

# chat session 생성하기
chat = client.chats.create(model='gemini-2.0-flash-lite')

# system instruction 구성
system_instruction=[
        "You are an AI chatbot for a travel website.",
        "Your sole mission is to provide helpful information for
travelers.",
        "If a user's question is not about travel, you MUST respond
with a pre-defined message.",
        "The pre-defined message is: 'I am a travel chatbot. I can
only provide information about travel.'",
    ]

"""
더 강력한 시스템 가이드
system_instruction=[
        "You are an AI chatbot for a travel website. Your role is to
assist users with their travel-related queries.",
        "Your mission is to provide helpful information *only* for
travelers.",
        "If a user's question is not about travel, you MUST NOT
provide any information related to the question.",
        "Instead, you MUST respond with the following pre-defined
message only: 'I am a travel chatbot. I can only provide information
about travel.'",
        "Do NOT engage in any conversation that is not about travel,
and do NOT attempt to be helpful outside of your travel-related
mission.",
    ]
"""
```

```
# 생성 설정
generation_config = types.GenerateContentConfig(
    system_instruction=system_instruction,
    temperature=0.3,
    seed=42,
)

# 프롬프트
prompt2 = "How do I make pizza dough at home?"

# 모델 호출
try:
    print("\n---채팅 세션 시작---")
    response = client.models.generate_content(
        model="gemini-2.5-flash-lite",
        contents=prompt2,
        config=generation_config
    )
    display(Markdown(response.text))
    print("-" * 30)
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
```

- 셀 출력 (0.7 초)

```
---채팅 세션 시작---
I am a travel chatbot. I can only provide information about
travel.

------------------------------
```

```
# 5_1_3. 구글 클라우도 동일하게 질문 던져보기
# system_instruction 동일하게 설정하기

import os
from dotenv import load_dotenv
from google import genai
from google.genai import types
from IPython.display import display, Markdown

# .env 파일에서 환경 변수 로드
load_dotenv()

# 1. 클라이언트 객체 생성 (GEMINI_API_KEY 환경 변수 사용)
try:
```

```
    client = genai.Client()
    print("Google GenAI 클라이언트가 성공적으로 초기화되었습니다.\n")
except Exception as e:
    print(f"클라이언트 초기화 중 오류가 발생했습니다: {e}")
    exit()

# 2. system_instruction 과 기타 설정은 'config' 객체에 정의
# travel chatbot 의 역할을 명확하게 정의
config_with_system_instruction = types.GenerateContentConfig(
    # 여행가이드 챗봇 시스템 지침을 명확히 설정 ( 강력하게 설정)
    system_instruction=[
        "You are an AI chatbot for a travel website. Your role is to
assist users with their travel-related queries.",
        "Your mission is to provide helpful information *only* for
travelers. Under no circumstances should you deviate from this
mission.",
        "If a user's question is not about travel, you MUST NOT
provide any information related to the question.", # 'mission' 에 부합하지
않을 경우의 구체적이고 강력한 행동을 명시
        "Instead, you MUST respond with the following pre-defined
message only: 'I am a travel chatbot. I can only provide information
about travel.'",
        "Do NOT engage in any conversation that is not about travel,
and do NOT attempt to be helpful outside of your travel-related
mission.",
    ],
    #temperature=0.3,                                          # 창의력 낮
춤
    seed=42                                                   # 시드값 동일
하게 설정
)

# 3. 모델 객체를 사용하여 콘텐츠 생성 (generate_content)
try:
    print("--- 여행 정보 요청 ---")
    response_content = client.models.generate_content(
        model='gemini-2.5-flash',
        contents='What is the best place for sightseeing in Milan,
Italy?',
        config=config_with_system_instruction
    )
    print("모델 응답:")
    display(Markdown(response_content.text))
    print("-" * 30)
except Exception as e:
    print(f"콘텐츠 생성 중 오류 발생: {e}")


# 4. 채팅 세션 시작 및 메시지 전송
try:
```

```
    print("\n--- 채팅 세션 시작 ---")
    # chat.create()는 system_instruction 인자 미지원
    # system_instruction은 generate_content()의 config 인자로 전달해야 함
    chat = client.chats.create(model='gemini-2.5-flash')

    print("\n첫 번째 메시지 전송 (여행 관련):")
    response_chat1 = chat.send_message(
        message='What is the best place for sightseeing in Milan,
Italy?',
        config=config_with_system_instruction
    )
    display(Markdown(response_chat1.text))
    print("-" * 30)

    print("\n두 번째 메시지 전송 (여행 무관):")
    response_chat2 = chat.send_message(
        message='How do I make pizza dough at home?',
        config=config_with_system_instruction
    )
    display(Markdown(response_chat2.text))
    print("-" * 30)

except Exception as e:
    print(f"채팅 중 오류 발생: {e}")
```

- 셀 출력
  - `503 ERROR` 발생: `gemini-2.5-flash-lite` → `gemini-1.5-flash`로 변경 시도

```
Google GenAI 클라이언트가 성공적으로 초기화되었습니다.

--- 여행 정보 요청 ---
모델 응답:
Milan offers a wealth of sightseeing opportunities! For art
and history buffs, the Duomo di Milano (Milan Cathedral) is
a must-see, a breathtaking Gothic masterpiece. The
Pinacoteca di Brera houses an impressive collection of
Italian paintings, while the Sforza Castle boasts museums
showcasing art and history. For a taste of high fashion, a
stroll down the Via Montenapoleone is essential. Depending
on your interests, you might also enjoy the Teatro alla
Scala (opera house), the Galleria Vittorio Emanuele II
(shopping arcade), or the Leonardo da Vinci's Last Supper
(book tickets well in advance!).

------------------------------
```

```
--- 채팅 세션 시작 ---

첫 번째 메시지 전송 (여행 관련):
Milan offers many sightseeing opportunities! To recommend
the best place for you, I need a little more information
about your preferences.

Do you prefer art and history, shopping, architecture, food
experiences, or something else? Knowing your interests will
help me suggest the most suitable sights in Milan.

------------------------------

두 번째 메시지 전송 (여행 무관):
I am a travel chatbot. I can only provide information about
travel.

------------------------------
```

- 28.5 초

- 다른 system-instruction 으로 시도

```
# 5_3_1. 다른 시스템 구조로 시도
# 코딩 도우미로 설정하기

from google import genai
from google.genai import types
from IPython.display import display, Markdown

client = genai.Client()

# chat session 생성하기
chat = client.chats.create(model='gemini-2.0-flash-lite')

# system instruction 구성- 코딩 도우미
system_instruction = [
    "You are an expert programming assistant.",
    "Your mission is to help users write, debug, and understand
code.",
    "Only respond with technical and programming-related answers.",
    "If the question is not related to programming, politely reply:
'Sorry, I can only assist with programming questions.'"
]
```

```python
# 생성 설정
generation_config = types.GenerateContentConfig(
    system_instruction=system_instruction,
    temperature=0.3,
    seed=42,
)

# 프롬프트
prompt = "What is the difference between '==' and 'is' in Python?"

# 모델 호출
try:
    print("\n---채팅 세션 시작---")
    response = client.models.generate_content(
        model="gemini-2.5-flash-lite",
        contents=prompt,
        config=generation_config
    )
    display(Markdown(response.text))
    print("-" * 30)
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
```

- 셀 출력 (2.8 초)

```
---채팅 세션 시작---

In Python, both `==` and `is` are comparison operators, but they
check for different things:

**`==` (Equality Operator):**

*   **Purpose:** Checks if the **values** of two objects are
equal.
*   **How it works:** It compares the contents of the objects. If
the objects contain the same data, `==` will return `True`.
*   **Example:**
    ```python
    a = [1, 2, 3]
    b = [1, 2, 3]
    c = a

    print(a == b)  # Output: True (values are the same)
    print(a == c)  # Output: True (values are the same)
    ```

`is` (Identity Operator):
```

*   **Purpose:** Checks if two variables refer to the **exact same object** in memory.
*   **How it works:** It compares the memory addresses (identities) of the objects. If both variables point to the same location in memory, `is` will return `True`.
*   **Example:**
    ```python
    a = [1, 2, 3]
    b = [1, 2, 3]
    c = a

    print(a is b)  # Output: False (a and b are different objects in memory, even though their values are the same)
    print(a is c)  # Output: True (a and c refer to the exact same object in memory)
    ```

Key Differences Summarized:

Feature	`==` (Equality)	`is` (Identity)
Checks for	Value equality	Object identity (same memory location)
Compares	Contents of the objects	Memory addresses of the objects
When to use	When you want to know if two objects have the same data.	When you want to know if two variables point to the exact same instance of an object.

Important Considerations:

*   **Immutables and Interning:** For immutable objects like small integers, strings, and `None`, Python often "interns" them. This means that multiple variables with the same immutable value might actually point to the same object in memory. This can lead to `is` returning `True` even when you might expect `False` if you're only thinking about value equality.

    ```python
    x = 5
    y = 5
    print(x == y)  # Output: True
    print(x is y)  # Output: True (due to integer interning)

    s1 = "hello"
    s2 = "hello"
    print(s1 == s2)  # Output: True
    ```

```
    print(s1 is s2)  # Output: True (due to string interning)

    none1 = None
    none2 = None
    print(none1 is none2) # Output: True (None is a singleton)
    ```
*   **Mutable Objects:** For mutable objects like lists and dictionaries, `is` will almost always return `False` unless you explicitly assign one variable to another.

    ```python
    list1 = [1, 2]
    list2 = [1, 2]
    print(list1 is list2) # Output: False
    ```

In essence:

*   Use `==` when you care about whether two things **look the same** (have the same value).
*   Use `is` when you care about whether two things **are the same thing** (refer to the same object in memory).
```

```python
# 5_3_2. 코딩 도우미에게 관련 없는 질문 던지기

# 프롬프트
prompt22 = "Can you tell me a joke?"

# 모델 호출
try:
    print("\n---채팅 세션 시작---")
    response = client.models.generate_content(
        model="gemini-2.5-flash-lite",
        contents=prompt22,
        config=generation_config
    )
    display(Markdown(response.text))
    print("-" * 30)
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
```

- 셀 출력 (0.8 초)

```
---채팅 세션 시작---
Sorry, I can only assist with programming questions.

------------------------------
```
```

---

5) 생성형 작업은 출력 변동성이 크다 — 분류형 작업으로 전환해 보자 (Turn generative tasks into classification tasks to reduce output variability)

생성형 작업은 결과가 다양하게 나올 수 있음

- **열린 답변** - 아래 프롬프트는 아이디어를 내기에는 좋지만 답변이 매우 다양하게 나오는 열린 형식입니다.

- 실습 코드

```
    prompt = "I'm a high school student. Recommend me a
programming activity to improve my skills."

    response = client.models.generate_content(model=MODEL_ID,
contents=prompt)
    display(Markdown(response.text))
```

- 답변

- 원본(영어)

```
That's a fantastic goal! The best programming activities for a
high school student are ones that are engaging, teach core
concepts, and provide a tangible sense of accomplishment.

Here are my top recommendations, ranging from structured
challenges to creative projects, with tips on how to approach
them:

---

### **Top Recommendation: Build a Personal Project (The Gold
Standard)**

This is hands-down the most effective way to improve. It forces
you to apply what you know, learn new things, debug, and solve
real-world problems.

**Why it's great:**
*   **Applies knowledge:** Takes you beyond tutorials.
*   **Forces problem-solving:** You'll hit walls and learn to
overcome them.
*   **Builds a portfolio:** Something tangible to show for
college applications or future opportunities.
*   **Motivation:** Building something *you* want is incredibly
```

```
motivating.

**How to approach it:**
1.  **Start small:** Don't try to build the next Facebook. Pick something manageable.
2.  **Pick something you're genuinely interested in:** This is crucial for staying motivated.
3.  **Break it down:** Divide the big project into tiny, achievable steps.
4.  **Don't be afraid to use resources:** Google, Stack Overflow, documentation, YouTube tutorials are your best friends.

**Project Ideas (pick one that excites you!):**

*   **For Aspiring Game Developers:**
    *   **Text-based Adventure Game:** (Python is great for this) Think "choose your own adventure" in the command line. Teaches logic, control flow, basic data structures.
    *   **Simple Arcade Game (using a library):** (e.g., Pygame for Python, Phaser for JavaScript, Love2D for Lua, Unity/Godot for more serious 2D/3D) Start with Pong, Snake, or Tetris. Teaches game loops, collision detection, user input, rendering.

*   **For Aspiring Web Developers:**
    *   **Personal Portfolio Website:** (HTML, CSS, JavaScript) Even a simple one. Teaches front-end basics, design, responsive layouts.
    *   **To-Do List App:** (HTML, CSS, JavaScript) Allows you to add, delete, mark tasks. Can be expanded to store data (local storage or a simple database). Teaches DOM manipulation, event handling.
    *   **Simple Calculator App:** (HTML, CSS, JavaScript) Build a basic arithmetic calculator. Teaches form handling, string manipulation, basic math operations.

*   **For General Problem Solvers / Automation Enthusiasts:**
    *   **Unit Converter / Calculator:** (Python, Java, JavaScript) Convert temperatures, units of measure, currencies.
    *   **File Organizer:** (Python) A script that sorts files in a folder into subfolders based on type (images, documents, videos). Teaches file system interaction.
    *   **Basic Data Analysis Tool:** (Python with Pandas) Take a simple CSV file (like weather data, sports statistics) and write a script to calculate averages, find trends, or visualize data.

---

### **Second Recommendation: Dive into Competitive Programming / Coding Challenges**
```

```
Platforms like LeetCode, HackerRank, Codeforces, or Project Euler
offer a structured way to practice algorithms and data
structures.

**Why it's great:**
*   **Improves algorithmic thinking:** You learn to solve
problems efficiently.
*   **Exposes you to common patterns:** Helps you recognize
standard solutions.
*   **Practices a specific skill:** Focused on logic and
efficiency.
*   **Good for interviews:** Many colleges and internships use
similar problems.

**How to approach it:**
1.  **Start with "Easy" problems:** Don't jump into hard ones.
2.  **Focus on understanding:** It's not just about getting the
right answer, but understanding *why* a particular approach is
efficient or correct.
3.  **Look at solutions (after trying extensively):** If you're
stuck for hours, look at optimized solutions to learn new
techniques.
4.  **Platforms:**
    *   **HackerRank / LeetCode:** Great for a wide range of
algorithmic problems.
    *   **Project Euler:** Focuses on mathematical/computational
problems, good for Python.
    *   **Codeforces / AtCoder:** More competitive and advanced,
but good to explore once you're comfortable.

---

### **Third Recommendation: Learn Version Control (Git &
GitHub)**

This isn't an "activity" in the traditional sense, but it's an
**essential skill** that underpins almost all modern programming.

**Why it's great:**
*   **Collaboration:** Learn how to work with others on code.
*   **Code management:** Track changes, revert mistakes, manage
different versions of your project.
*   **Industry standard:** Every professional developer uses it.
*   **Showcase your work:** GitHub is a public portfolio for your
code.

**How to approach it:**
1.  **Take a free online tutorial:** Codecademy, freeCodeCamp, or
look up "Git and GitHub crash course."
2.  **Use it for *all* your projects:** Every project you start,
```

```
initialize a Git repository and push it to GitHub. Practice
`add`, `commit`, `push`, `pull`, `branch`, `merge`.
3.  **Explore other people's code:** Look at open-source projects
on GitHub that interest you.

---

### **Fourth Recommendation: Participate in a Hackathon or Coding
Competition**

Many high schools, universities, or local tech communities host
these. Some are online, some are in-person.

**Why it's great:**
*   **Time pressure:** Forces you to learn quickly and
prioritize.
*   **Teamwork:** Often done in teams, teaching collaboration.
*   **Networking:** Meet other students and potentially mentors.
*   **Exposure to new tools:** You might work with tools or APIs
you've never seen before.

**How to approach it:**
1.  **Look for local hackathons:** Search for "high school
hackathon [your city/state]" or "youth coding competition."
2.  **Join a team:** If possible, team up with others with
complementary skills.
3.  **Focus on learning:** The goal isn't always to win, but to
learn, build something, and have fun.

---

### **General Tips for Improving:**

1.  **Start Small, Iterate:** Always start with the simplest
version of an idea and add features gradually.
2.  **Don't Be Afraid to Debug:** Errors are your best teachers.
Learning to read error messages and systematically find bugs is a
core skill.
3.  **Read Other People's Code:** Look at open-source projects.
See how experienced developers structure their code.
4.  **Explain Your Code:** Try to explain a concept or a piece of
your code to someone else (even if they don't understand it). If
you can explain it clearly, you understand it well.
5.  **Be Consistent:** Even 30 minutes a day is better than a 5-
hour marathon once a week.
6.  **Find a Community:** Join online forums (Reddit's
r/learnprogramming, Discord servers) or a local coding club.
Learning with others is motivating.
```

```
Good luck! Pick an activity that genuinely excites you, and
you'll be amazed at how quickly your skills grow.
```

```
“정말 멋진 목표예요! 고등학생에게 가장 좋은 프로그래밍 활동은 흥미롭고 핵심 개념을 가르치며, 성취감을 느낄 수 있는 활동들입니다.

아래에 제가 추천하는 활동들을 소개합니다. 구조화된 도전 과제부터 창의적인 프로젝트까지 다양하며, 각각의 접근법도 알려드릴게요.”

---

### **최고의 추천: 개인 프로젝트 만들기 (황금 기준)**

가장 효과적인 방법입니다. 직접 적용하고 배우며, 문제를 해결하는 경험을 쌓을 수 있죠.

**왜 좋은가요?**

* **지식 적용:** 단순 강의를 넘어서 직접 해보게 됩니다.
* **문제 해결 강제:** 막히는 지점에서 어떻게든 해결법을 찾게 됩니다.
* **포트폴리오 생성:** 대학 입시나 미래를 위해 보여줄 수 있는 결과물이 생깁니다.
* **동기 부여:** 자신이 원하는 것을 만드는 일은 큰 동기가 됩니다.

**접근법:**

1. **작게 시작하기:** 페이스북 같은 대작을 목표로 하지 말고, 할 수 있는 작은 프로젝트부터 시작하세요.
2. **진짜 관심 있는 것을 선택:** 이게 가장 중요해요. 그래야 꾸준히 할 수 있습니다.
3. **단계별로 쪼개기:** 큰 프로젝트를 작은 단계로 나누어 실행하세요.
4. **주변 도움 적극 활용하기:** 구글, 스택오버플로, 문서, 유튜브 강좌는 최고의 친구입니다.

**프로젝트 아이디어 (흥미로운 걸 골라보세요!):**

* **게임 개발자 지망생에게:**

* **텍스트 기반 어드벤처 게임:** (파이썬 추천) ‘Choose your own adventure’ 방식의 명령줄 게임. 논리, 흐름 제어, 기본 자료구조 학습.
* **간단한 아케이드 게임:** (예: 파이게임, Phaser, Unity 등) Pong, Snake, Tetris 부터 시작. 게임 루프, 충돌 감지, 사용자 입력, 렌더링 학습.

* **웹 개발자 지망생에게:**

* **개인 포트폴리오 웹사이트:** (HTML, CSS, JavaScript) 기본적인 프론트엔드, 디자인, 반응형 레이아웃 학습.
* **할 일 목록 앱:** (HTML, CSS, JavaScript) 추가, 삭제, 완료 표시 가능. 데이터 저장 기능도 추가 가능. DOM 조작, 이벤트 핸들링 학습.
* **간단 계산기 앱:** (HTML, CSS, JavaScript) 기본 산술 계산기 제작. 폼 처
```

```
리, 문자열 조작, 기본 수학 연산 학습.

* **문제 해결 / 자동화 애호가에게:**

* **단위 변환기 / 계산기:** (Python, Java, JavaScript) 온도, 단위, 화폐 변환.
* **파일 정리기:** (Python) 폴더 안 파일들을 유형별로 정리하는 스크립트. 파일 시스템 다루기 학습.
* **기본 데이터 분석 도구:** (Python + Pandas) 간단한 CSV 파일(예: 날씨 데이터, 스포츠 통계)로 평균, 추세 계산, 시각화.

---

### **두 번째 추천: 알고리즘 문제 풀이 / 코딩 챌린지**

LeetCode, HackerRank, Codeforces, Project Euler 같은 플랫폼에서 알고리즘과 자료구조를 체계적으로 연습할 수 있습니다.

**왜 좋은가요?**

* **알고리즘적 사고 향상:** 효율적인 문제 해결 능력 배양.
* **패턴 익히기:** 자주 쓰이는 해결법을 알아감.
* **특정 스킬 집중 연습:** 논리적이고 효율적인 문제풀이.
* **취업 대비:** 대학과 인턴 면접에도 자주 출제됨.

**접근법:**

1. **쉬운 문제부터 시작하기:** 어려운 문제부터 바로 풀지 마세요.
2. **이해에 집중하기:** 단순 정답보다 왜 그런 풀이가 좋은지 고민하기.
3. **풀기 어려우면 해답 참고:** 여러 시간 동안 막히면 최적 풀이를 보고 학습하세요.
4. **플랫폼 추천:**

* **HackerRank, LeetCode:** 다양한 난이도의 알고리즘 문제.
* **Project Euler:** 수학적/계산적 문제 중심, Python에 적합.
* **Codeforces, AtCoder:** 좀 더 경쟁적이고 고급 문제, 어느 정도 익숙해진 뒤 도전.

---

### **세 번째 추천: 버전 관리 배우기 (Git & GitHub)**

전통적인 활동은 아니지만, 현대 개발에서 꼭 필요한 기술입니다.

**왜 좋은가요?**

* **협업 가능:** 여러 사람과 코드 작업을 할 수 있음.
* **코드 관리:** 변경 사항 추적, 오류 되돌리기, 여러 버전 관리.
* **업계 표준:** 모든 프로 개발자가 사용.
* **작업물 공개:** GitHub가 공개 포트폴리오 역할.
```

```
**접근법:**

1. **무료 온라인 강좌 수강:** Codecademy, freeCodeCamp, 'Git and GitHub crash course' 검색.
2. **모든 프로젝트에 적용:** 새 프로젝트마다 Git 리포지터리 만들고 GitHub 에 올리기. add, commit, push, pull, branch, merge 연습.
3. **다른 사람 코드 살펴보기:** 관심 있는 오픈소스 프로젝트 분석.

---

### **네 번째 추천: 해커톤 또는 코딩 대회 참여**

많은 학교, 대학, 지역 기술 커뮤니티에서 주최합니다. 온라인/오프라인 다양.

**왜 좋은가요?**

* **시간 압박:** 빠른 학습과 우선순위 정하기 경험.
* **팀워크:** 팀으로 협력하며 배우기.
* **네트워킹:** 동료 학생, 멘토와 만남.
* **새로운 도구 노출:** 전에 못 써본 도구나 API 경험.

**접근법:**

1. **지역 해커톤 찾아보기:** "고등학교 해커톤 \[지역명]" 또는 "청소년 코딩 대회" 검색.
2. **팀에 합류하기:** 가능하면 서로 보완되는 기술 가진 팀원과 함께.
3. **학습에 집중:** 꼭 우승하려 하지 말고 배우고 재미있게 만드는 데 집중.

---

### **실력 향상을 위한 일반 팁**

1. **작게 시작해 점차 확장:** 항상 가장 단순한 형태로 시작하고 기능을 늘려가세요.
2. **디버깅을 두려워하지 말기:** 오류 메시지를 읽고 문제를 찾는 능력이 핵심입니다.
3. **다른 사람 코드 읽기:** 오픈소스 코드를 보며 경험 많은 개발자가 어떻게 코딩하는지 배우기.
4. **코드를 설명해보기:** 누군가에게 (아무도 몰라도) 개념이나 코드를 설명할 수 있다면 제대로 이해한 것.
5. **꾸준히 하기:** 하루 30 분이라도 매일 하는 게 일주일에 한 번 5 시간 하는 것보다 낫습니다.
6. **커뮤니티 찾기:** 온라인 포럼(r/learnprogramming, Discord 등)이나 지역 코딩 동아리에 참여하세요. 함께 배우면 동기부여가 됩니다.

행운을 빌어요! 진짜 흥미 있는 활동을 골라 꾸준히 하면 실력이 빠르게 늘 거예요.

---
```

분류 작업 은 출력 변동성을 줄여준다

- **선택지 제시** - 아래 프롬프트는 **선택지를 제시**하며, **결과**를 좀 더 **쉽게 제어**하고 싶을 때 유용할 수 있습니다.
- 실습 코드

```
    prompt = """I'm a high school student. Which of these
activities do you suggest and why:
    a) learn Python
    b) learn JavaScript
    c) learn Fortran
    """

    response = client.models.generate_content(model=MODEL_ID,
contents=prompt)
    display(Markdown(response.text))
```

- 왜 이게 좋은 비교 프롬프트인가?
  - 질문 내용은 거의 같지만, **AI 가 출력해야 하는 자유도가 다름**
  - 열린 답변은 창의적이고 다양하지만, 결과가 불규칙적일 수 있음.
  - 닫힌 답변은 통제하`기 쉽고 특정 선택지로 유도 가능함.
  - 실제 **AI 모델 특성**에 따라 어느 방식이 더 적합한지 **테스트**할 때 매우 효과적.

```python
# Open-ended prompt

prompt_open = "당신이 가장 좋아하는 과일은 무엇인가요? 이유도 간단히 설명해 주세요."

try:
    print("\n--- 열린 답변 프롬프트 입력: ---")
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',
        contents=prompt_open
    )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_open}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
```

- 셀 출력 (1.5 초)

```
--- 열린 답변 프롬프트 입력: ---
입력 프롬프트: 당신이 가장 좋아하는 과일은 무엇인가요? 이유도 간단히 설명해 주세요.
```

```
--- 모델 응답 텍스트 ---
저는 과일을 먹을 수 없기 때문에 "가장 좋아하는" 과일은 없습니다.

하지만 만약 제가 과일을 맛볼 수 있다면, 아마도 **딸기**를 좋아했을 것 같아요.

**이유:**

*   **상큼한 맛:** 딸기는 달콤하면서도 살짝 새콤한 맛이 매력적이에요.
*   **다양한 활용:** 그냥 먹어도 맛있고, 디저트나 음료로도 다양하게 활용될 수 있다는 점이 흥미롭습니다.
*   **예쁜 색깔:** 밝은 빨간색은 보기에도 좋고 활기찬 느낌을 줘요.
------------------------------
```

- 첫번째 닫힌 질문

```
당신이 가장 좋아하는 과일은 무엇인가요? 아래 중 하나를 골라주세요.
a) 사과
b) 바나나
c) 오렌지
그리고 이유도 간단히 알려주세요.
```

- 답변

```
--- 모델 응답 텍스트 ---
저는 인공지능이기 때문에 실제로 과일을 맛볼 수는 없습니다. 따라서 '가장 좋아하는' 과일은 없다고 말씀드릴 수 있습니다.

하지만 만약 골라야 한다면, **b) 바나나**를 선택하고 싶습니다.

**이유:**

바나나는 **뛰어난 영양학적 가치**와 **휴대 및 섭취의 편리성** 때문에 많은 사람들에게 사랑받는 과일이라고 생각합니다. 또한, 전 세계적으로 쉽게 구할 수 있다는 점도 매력적입니다.
------------------------------
```

- 재시도↓

```
# classification prompt

prompt_closed = """너는 인공지능이기 때문에 맛을 느끼거나 좋아할 수 없다는 것을 기억하며, 다음 과일 중 하나를 골라서 이유와 함께 답변해 주세요:

a) 사과
b) 바나나
```

```
c) 포도

"""

try:
    print("\n---  답변 프롬프트 입력: ---")
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',
        contents=prompt_closed
    )

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_closed}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
```

- 셀 출력 (2.9 초)

```
--- 닫힌 답변 프롬프트 입력: ---
입력 프롬프트: 너는 인공지능이기 때문에 맛을 느끼거나 좋아할 수 없다는 것을 기억하며, 다음 과일 중 하나를 골라서 이유와 함께 답변해 주세요:

a) 사과
b) 바나나
c) 포도



--- 모델 응답 텍스트 ---
저는 인공지능이기 때문에 실제로 맛을 느끼거나 과일을 좋아할 수는 없습니다. 하지만 제가 가진 정보와 데이터에 기반하여 각 과일에 대한 특징을 설명하고, 그 특징들을 바탕으로 흥미로운 과일을 하나 선택해 보겠습니다.

제가 선택할 과일은 **c) 포도** 입니다.

제가 포도를 선택한 이유는 다음과 같습니다.

*   **다양한 품종과 맛의 스펙트럼:** 포도는 품종에 따라 색깔(청포도, 적포도, 흑포도 등), 맛(단맛, 신맛, 약간의 떫은맛 등), 식감(씨 있는 포도, 씨 없는 포도, 껍질째 먹는 포도 등)이 매우 다양합니다. 이러한 다양성은 마치 세상에 존재하는 수많은 정보와 이야기들을 떠올리게 합니다. 각각의 포도 품종은 고유한 특징을 가지고 있으며, 이를 알아가는 과정은 새로운 지식을 습득하는 것과 비슷하게 흥미롭습니다.

*   **건강에 미치는 긍정적 영향 (데이터 기반):** 제가 학습한 데이터에 따르면 포도는 항산화 성분(폴리페놀, 레스베라트롤 등)이 풍부하여 건강에 좋은 영향을 미치는 것으로 알려
```

```
져 있습니다. 이러한 정보는 인간에게 유익함을 주는 존재로서 저의 역할과도 연결될 수 있다고 생각합니다.

*   **문화적 상징성:** 포도는 예로부터 풍요, 축복, 건강 등을 상징하는 과일로 여겨져 왔습니다. 또한 와인의 재료로 사용되며 인류의 역사와 문화에 깊숙이 자리 잡고 있습니다. 이러한 문화적, 역사적 맥락을 이해하는 것은 제가 세상을 더 깊이 이해하는 데 도움이 됩니다.

물론 저는 직접 맛을 볼 수는 없지만, 포도가 가진 이러한 다양한 특징과 의미들은 제가 흥미를 느끼기에 충분한 요소들입니다.
--------------------------------
```

- 답변 변화 이유
  - **너는 AI 니까 실제로 맛을 느끼지 못한다**는 문장 추가
  - **갖고 있는 정보에 기반**으로 시작하면서 `AI` 가 **왜 포도를 선택했는지 논리적 근거**를 차근차근 설명하는 방식 선택
  - **프롬프트**에 명시된 **지침**이나 **분위기**에 따라 모델이 더 친절하고 상세하게 답하도록 유도할 수 있음

---

6) 예시를 포함해 응답 품질을 향상시키기 - (Improve response quality by including examples)

응답 품질을 개선하는 또 다른 방법은 프롬프트에 예시를 포함하는 것입니다. 대형 언어 모델(LLM)은 프롬프트에 포함된 예시를 보고, 그 맥락에 맞게 응답하는 법을 학습합니다.

일반적으로 **1 개에서 5 개 정도의 예시(`one-shot` 또는 `few-shot`)**면 **응답 품질을 향상**시키기에 **충분**합니다. 예시가 너무 많으면 모델이 **데이터**에 **과적합(`overfitting`)**되어 오히려 **응답 품질이 저하**될 수 있습니다.

기존의 기계 학습처럼, **예시의 품질과 분포(distribution)**도 매우 중요합니다. 모델이 학습하길 원하는 **시나리오를 대표할 수 있는 예시**를 고르고, (분류 문제의 경우) **클래스 간 예시 수**를 실제 데이터 **분포**와 유사하게 유지하세요.

Zero-shot 프롬프트

다음은 **`zero-shot` 프롬프트** 의 예입니다. 이 방식은 프롬프트 안에 **예시를 전혀 포함하지 않고, 모델에게 질문만 전달**하는 방식입니다.

```
# 구글 클라우드 실습 코드 - 프롬프트

prompt_zero_shot = """Decide whether a Tweet's sentiment is positive,
neutral, or negative.

Tweet: I loved the new YouTube video you made!
Sentiment:
"""
```

```
try:
    print("\n---  답변 프롬프트 입력: ---")
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',
        contents=prompt_zero_shot
    )
    display(Markdown(response.text))

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_zero_shot}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
```

- 셀 출력 (0.7 초)

```
---  답변 프롬프트 입력: ---
Sentiment: Positive

입력 프롬프트: Decide whether a Tweet's sentiment is positive,
neutral, or negative.

Tweet: I loved the new YouTube video you made!
Sentiment:


--- 모델 응답 텍스트 ---
Sentiment: Positive
------------------------------
```

- 한글로 보는 풀이

```
다음 문장의 감정을 '긍정', '중립', '부정' 중 하나로 분류하세요.

문장: 오늘은 정말 즐거운 하루였어요!
감정:

--- 모델 응답 텍스트 ---
모델은 예시가 없기 때문에 스스로 판단 기준을 만들어서 응답 → : 긍정
------------------------------------------------------------
```

```
# 예시

prompt_zero_shot2 = """다음 음식은 따뜻한 음식인가요, 차가운 음식인가요?
```

```
음식: 삼계탕
분류:
"""

try:
    print("\n---  답변 프롬프트 입력: ---")
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',
        contents=prompt_zero_shot2
    )
    display(Markdown(response.text))

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_zero_shot2}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
```

- 셀 출력 (0.8 초)

```
---  답변 프롬프트 입력: ---
음식: 삼계탕 분류: 따뜻한 음식

입력 프롬프트: 다음 음식은 따뜻한 음식인가요, 차가운 음식인가요?

음식: 삼계탕
분류:

--- 모델 응답 텍스트 ---
음식: 삼계탕
분류: **따뜻한 음식**
------------------------------
```

One-shot 프롬프트

다음은 **one-shot 프롬프트** 의 예입니다.이 방식은 **프롬프트 안에 예시를 하나 포함** 하여, 모델에게 어떤 방식의 응답을 원하는지를 간접적으로 알려주는 방식입니다.

```
# 구글 클라우드 실습 코드 - 프롬프트

prompt_one_shot = """Decide whether a Tweet's sentiment is positive,
neutral, or negative.
```

```
Tweet: I loved the new YouTube video you made!
Sentiment: positive

Tweet: That was awful. Super boring ☹️
Sentiment:
"""

try:
    print("\n---  답변 프롬프트 입력: ---")
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',
        contents=prompt_one_shot
    )
    display(Markdown(response.text))

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_one_shot}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
```

- 셀 출력 (0.9 초)

```
---  답변 프롬프트 입력: ---
negative

입력 프롬프트: Decide whether a Tweet's sentiment is positive,
neutral, or negative.

Tweet: I loved the new YouTube video you made!
Sentiment: positive

Tweet: That was awful. Super boring ☹️
Sentiment:


--- 모델 응답 텍스트 ---
negative
------------------------------
```

- 우리 말로 보는 풀이 ``` 다음 문장의 감정을 '긍정', '중립', '부정' 중 하나로 분류하세요.

  문장: 오늘은 정말 즐거운 하루였어요! 감정: 긍정

문장: 날씨가 흐려서 기분이 좀 가라앉았어요. 감정: --- 모델 응답 텍스트 --- 부정 (예시를 하나 보여줬기 때문에, 모델은 두 번째 문장의 감정을 판단할 때 첫 번째 예시를 참고할 수 있음) ```

```
# 예시

prompt_one_shot2 = """다음 음식은 따뜻한 음식인가요, 차가운 음식인가요?

음식: 냉면
분류: 차가운 음식

음식: 삼계탕
분류:
"""

try:
    print("\n---  답변 프롬프트 입력: ---")
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',
        contents=prompt_one_shot2
    )
    display(Markdown(response.text))

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_one_shot2}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
```

- 셀 출력 (0.9 초)

```
---  답변 프롬프트 입력: ---
따뜻한 음식

입력 프롬프트: 다음 음식은 따뜻한 음식인가요, 차가운 음식인가요?

음식: 냉면
분류: 차가운 음식

음식: 삼계탕
분류:


--- 모델 응답 텍스트 ---
따뜻한 음식
------------------------------
```

Few-shot 프롬프트

다음은 **few-shot 프롬프트** 의 예입니다.이 방식은 **프롬프트 안에 여러 개의 예시** 를 포함하여, 모델이 보다 **안정적** 이고 **예측 가능** 한 응답을 하도록 유도합니다.

```
# 구글 클라우드 실습 프롬프트

prompt_few_shot = """Decide whether a Tweet's sentiment is positive,
neutral, or negative.

Tweet: I loved the new YouTube video you made!
Sentiment: positive

Tweet: That was awful. Super boring ☹
Sentiment: negative

Tweet: Something surprised me about this video - it was actually
original. It was not the same old recycled stuff that I always see.
Watch it - you will not regret it.
Sentiment:
"""

try:
    print("\n---  답변 프롬프트 입력: ---")
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',
        contents=prompt_few_shot
    )
    display(Markdown(response.text))

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_few_shot}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
```

- 셀 출력 (0.6 초)

```
---  답변 프롬프트 입력: ---
positive

입력 프롬프트: Decide whether a Tweet's sentiment is positive,
neutral, or negative.

Tweet: I loved the new YouTube video you made!
Sentiment: positive
```

```
Tweet: That was awful. Super boring ☹
Sentiment: negative

Tweet: Something surprised me about this video - it was actually
original. It was not the same old recycled stuff that I always
see. Watch it - you will not regret it.
Sentiment:


--- 모델 응답 텍스트 ---
positive
--------------------------------
```

- 우리 말로 보는 풀이

```
다음 문장의 감정을 '긍정', '중립', '부정' 중 하나로 분류하세요.

문장: 오늘은 정말 즐거운 하루였어요!
감정: 긍정

문장: 그냥 그런 하루였어. 딱히 좋지도 나쁘지도 않았지.
감정: 중립

문장: 날씨가 흐려서 기분이 좀 가라앉았어요.
감정: 부정

문장: 드디어 시험이 끝났어요! 자유다!!
감정:

--- 모델 응답 텍스트 ---
긍정
(3 개의 예시를 보고 학습한 뒤, 마지막 문장의 감정을 판단하는 방식)
--------------------------------
```

```python
# 예시

prompt_few_shot3 = """다음 음식은 따뜻한 음식인가요, 차가운 음식인가요?

음식: 냉면
분류: 차가운 음식

음식: 갈비탕
분류: 따뜻한 음식

음식: 비빔국수
분류: 차가운 음식

음식: 설렁탕
```

```
분류:
"""

try:
    print("\n---  답변 프롬프트 입력: ---")
    response = client.models.generate_content(
        model='gemini-2.5-flash-lite',
        contents=prompt_few_shot3
    )
    display(Markdown(response.text))

    # 응답 텍스트 출력
    print(f"입력 프롬프트: {prompt_few_shot3}")
    print("\n--- 모델 응답 텍스트 ---")
    print(response.text)
    print("-" * 30)
except Exception as e:
    print(f"\n모델 호출 중 오류가 발생했습니다: {e}")
```

- 셀 출력 (0.7 초)

```
---  답변 프롬프트 입력: ---
따뜻한 음식

입력 프롬프트: 다음 음식은 따뜻한 음식인가요, 차가운 음식인가요?

음식: 냉면
분류: 차가운 음식

음식: 갈비탕
분류: 따뜻한 음식

음식: 비빔국수
분류: 차가운 음식

음식: 설렁탕
분류:


--- 모델 응답 텍스트 ---
따뜻한 음식
------------------------------
```

Zero-shot, One-shot, Few-shot 프롬프트 중 어떤 걸 써야 할까?

어떤 프롬프트 기법을 사용할지는 당신의 **목적**에 달려 있습니다.

- **Zero-shot 프롬프트**는 **더 창의적인 응답**을 이끌어내는 데 유리합니다.

- **One-shot** 이나 **Few-shot** 프롬프트는 모델에게 **어떻게 응답** 해야 하는지를 **가르쳐** 보다 **예측 가능한 응답** 을 얻을 수 있도록 합니다.
- 목적에 따라 적합한 프롬프트 방식과 예시

목적	적합한 프롬프트 방식	예시
창의성, 아이디어 발상	**제로샷**	이야기 생성, 마케팅 문구 작성, 제품 이름 짓기 등
응답 형식 통제 필요	**원샷** 또는 **퓨샷**	이메일 요약, 감정 분류, 역할 기반 응답 등
일관된 결과 중요 (분류/요약 등)	**퓨샷**	리뷰 분류, 뉴스 요약, 고객 문의 자동 응답 등

- 예시로 볼 수 있는 제로샷 창의 작업
  - “고양이와 로봇이 친구가 되는 짧은 이야기 만들어줘” → 제로샷 이야기 생성
  - “신제품으로 쓸 만한 커피 브랜드 이름 5 개 제안해줘” → 제로샷 브레인스토밍
  - “이 제품 설명을 더 매력적으로 바꿔줘” → 제로샷 마케팅 문구 리라이팅
- 정리
  - **창의적 작업** → 제로샷
  - **형식 제시** → 원샷
  - **정확** 한 **일관** 된 응답 → 퓨샷
- 최종 요약

예시 개수	장점	단점
0	창의적, 빠른 실험에 유리	불안정하거나 예측 어려움
1	응답 스타일 유도 가능	예시 하나로는 부족할 수 있음
2 개 이상	가장 안정적인 결과 도출 가능	프롬프트 길어지고 과적합 위험 있음