# explain\_program5.py プログラムマニュアル

## プログラム名

explain\_program5.py

## プログラムの目的

explain\_program5.py は、指定されたプログラムコードをAI（OpenAI または Google Gemini）を用いて自動的に解析し、その目的、原理、使い方、使用例などをMarkdown形式で記述したドキュメントを生成するためのツールです。開発者が手作業で行うドキュメント作成の負担を軽減し、コードベースの理解と共有を促進することを目的としています。

## 原理

本プログラムは、以下の主要なコンポーネントとフローに基づいて動作します。

### 1. AIサービスとの連携

* **APIの利用**: OpenAI (GPT-5) および Google Gemini (Gemini 2.5) の各APIを利用して、コード解析とドキュメント生成を行います。
* **カスタムライブラリ**: tkai\_lib というカスタムライブラリを介して、これらのAIサービスとの通信を抽象化しています。これにより、APIキーの管理やモデルの選択が容易になっています。
* **設定ファイル**: AIサービスの認証情報（APIキー）や使用するモデル名は、ai.env という環境設定ファイルから読み込まれます。

### 2. 設定とプロンプトの管理

* **INIファイルによる設定**: ドキュメント生成のための詳細な指示（プロンプトテンプレートやシステムロール）は、専用のINIファイル（デフォルトは explain\_program5.ini）から読み込まれます。
* **read\_ini 関数**: この関数は、key=value 形式の設定を解析し、複数行にわたるテキストブロック（"""...""" や '''...''' で囲まれた部分）や、INIファイル内で定義された他の変数への参照（$VARIABLE）の展開をサポートしています。
* **プロンプトテンプレート**: INIファイル内には、PROMPT\_MAIN (メインプログラム用) や PROMPT\_LIB (ライブラリ用) といったキーでプロンプトテンプレートが定義されます。これらのテンプレートには、{{script\_name}} (ファイル名)、{{code}} (プログラムコード)、{{lang}} (プログラミング言語) などのプレースホルダーが含まれており、実行時に実際の値に置換されます。
* **システムロール**: AIに対する一般的な指示や役割設定は、SYSTEM\_ROLE キーでINIファイルに定義されます。

### 3. 処理フロー

1. **コマンドライン引数の解析**: argparse モジュールを使用して、対象ファイルのワイルドカードパターン、出力ファイル名、API選択、INIファイルパス、プログラムタイプ、更新/上書きオプションなどのコマンドライン引数を解析します。
2. **AI設定の読み込み**: ai.env ファイルからAI APIのキーとモデル名を読み込みます。
3. **INI設定の読み込み**: 指定されたINIファイルからプロンプトテンプレート、システムロールなどの設定を読み込みます。
4. **対象ファイルの特定**: コマンドラインで指定されたワイルドカードパターン（例: '\*.py'）に基づいて、処理対象となるプログラムファイルのリストを生成します。
5. **ファイルごとの処理**:
   * **スキップ判定**: 出力ファイルが既に存在し、--overwrite オプションが指定されていない、または--update オプションが有効で入力ファイルより新しい場合、そのファイルの処理をスキップします。
   * **コードの読み込み**: 入力プログラムファイルのコード内容をUTF-8またはShift-JISエンコーディングで読み込みます。
   * **言語とプログラムタイプの判別**: ファイルの拡張子 (.py, .c, .js など) からプログラミング言語を自動判別します。また、get\_program\_type 関数やコマンドラインの --program\_type オプションにより、プログラムがメインプログラム (main) なのか、ライブラリ (lib) なのかを判定します。
   * **プロンプトの構築**: 判別されたプログラムタイプに対応するプロンプトテンプレートをINIファイルから選択し、読み込んだコード内容、ファイル名、言語を埋め込んで最終的なプロンプトを構築します。
   * **AI呼び出し**: 選択されたAI API（OpenAIまたはGoogle Gemini）に構築したプロンプトとシステムロールを送信し、ドキュメント生成をリクエストします。
   * **ドキュメントの抽出と保存**: AIから返された応答からドキュメント本文を抽出し、Markdown形式のファイルとして出力先に保存します。
   * **待機**: 各AIリクエスト間に1秒の待機を挿入し、APIレート制限への配慮とサーバ負荷の軽減を図ります。

これらのステップを通じて、開発者は簡単に高品質なコードドキュメントを自動生成することができます。

## 使い方 (Usage)

### 基本構文

python explain\_program5.py <pattern> [output\_filename] [OPTIONS]

### 引数

* <pattern>:
  + 対象とするプログラムファイルのワイルドカードパターンを指定します。
  + 例: '\*.py', './src/\*.c', 'main.js'
* [output\_filename]:
  + 単一ファイルを処理する場合の出力ファイル名を指定します。
  + この引数を省略した場合、入力ファイル名に .md 拡張子を付加したものがデフォルトの出力ファイル名となります（例: my\_script.py → my\_script.md）。
  + 複数のファイルが <pattern> で指定されている場合、この引数は使用できません。各入力ファイルに対応する \*.md ファイルが自動的に生成されます。

### オプション

* --inifile <path>:
  + プロンプト設定を定義するINIファイルのパスを指定します。
  + デフォルトは、スクリプトと同じディレクトリにある explain\_program5.ini です。
* --api {openai,openai5,google,gemini}:
  + 使用するAI APIを選択します。
  + openai: OpenAI API (主にGPT-4系モデル) を使用します。OPENAI\_MODEL 環境変数で指定されたモデルが使われます。
  + openai5: OpenAI API (主にGPT-5系モデル) を使用します。OPENAI\_MODEL5 環境変数で指定されたモデルが使われます。
  + google: Google Gemini API を使用します。GEMINI\_MODEL 環境変数で指定されたモデルが使われます。
  + gemini: google と同じです。
  + デフォルトは google です。
* -t, --program\_type {main,lib}:
  + プログラムの種類を手動で指定します。
  + main: プログラムをメイン実行可能なスクリプトとして扱います。INIファイルから PROMPT\_MAIN が選択されます。
  + lib: プログラムをライブラリとして扱います。INIファイルから PROMPT\_LIB が選択されます。
  + このオプションを省略した場合、ファイル名から自動判定されます（例: .pm 拡張子や tk で始まるファイルは lib と判定されます）。
* -u, --update <0|1>:
  + 更新チェックを行います。
  + 1: 出力ファイルが存在し、かつ入力ファイルよりも新しい場合に処理をスキップします。
  + 0: このチェックを行いません（デフォルト）。--overwrite との組み合わせで挙動が変わります。
* -w, --overwrite <0|1>:
  + 出力ファイルが存在する場合の挙動を指定します。
  + 1: 既存の出力ファイルを上書きします。
  + 0: 既存の出力ファイルを上書きしません（デフォルト）。上書きしない場合、ファイルはスキップされます。

### 設定ファイル

#### ai.env ファイルの例

AIサービスへの認証情報と使用モデルは、スクリプトが実行される環境に ai.env というファイル名で配置するか、環境変数として設定しておく必要があります。

# OpenAI API Keys  
OPENAI\_API\_KEY=sk-xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  
OPENAI\_MODEL=gpt-4-turbo9 # 'openai' APIオプションで使用  
OPENAI\_MODEL5=gpt-5.2 # 'openai5' APIオプションで使用  
  
# Google Gemini API Keys  
GEMINI\_API\_KEY=AIzaSyxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  
GEMINI\_MODEL=gemini-1.5-flash # 'google' または 'gemini' APIオプションで使用

#### explain\_program5.ini ファイルの例

プロンプトテンプレートやAIへの役割指示は、INIファイルに記述します。デフォルトのファイル名は explain\_program5.ini ですが、--inifile オプションで変更できます。

# システムロール（AIへの指示）  
SYSTEM\_ROLE=You are a helpful AI assistant that specializes in explaining code.  
# または、より詳細に:  
# SYSTEM\_ROLE="""  
# あなたは熟練したソフトウェアエンジニアであり、AIアシスタントです。  
# どんなプログラミング言語のコードでも、その目的、原理、使い方、そして具体的な使用例を、  
# 初心者でも理解できるように明確かつ詳細に説明することができます。  
# 出力はMarkdown形式でお願いします。  
# """  
  
# メインプログラム用のプロンプトテンプレート  
PROMPT\_MAIN="""  
以下の{{lang}}プログラム'{{script\_name}}'について、以下の項目で詳細なドキュメントを作成してください。  
Markdown形式で出力し、目的、原理、使い方、使用例のセクションを含めてください。  
特に、目的と原理は、初心者にも理解できるように詳細かつ具体的に記述してください。  
使用例は、実際のコマンドライン引数を含めて記述し、簡単な実行方法を示してください。  
  
---  
  
{{code}}  
  
---  
  
"""  
  
# ライブラリ用のプロンプトテンプレート  
PROMPT\_LIB="""  
以下の{{lang}}ライブラリ'{{script\_name}}'について、以下の項目で詳細なドキュメントを作成してください。  
Markdown形式で出力し、概要、機能、使い方、関数/クラス一覧のセクションを含めてください。  
特に、使い方では、このライブラリを他のプログラムからどのようにインポートし、使用するかを具体的なコード例を交えて説明してください。  
  
---  
  
{{code}}  
  
---  
  
"""

**INIファイルの変数展開:**

INIファイル内で KEY=$ANOTHER\_KEY のように記述することで、他のキーの値を参照できます。 例: COMMON\_PROMPT=This is a base., PROMPT\_MAIN=$COMMON\_PROMPT for main.

## 使用例

以下の使用例では、カレントディレクトリに ai.env と explain\_program5.ini が適切に設定されていることを前提とします。

### 1. 単一Pythonファイルのドキュメントを生成

my\_script.py というファイルの内容をAIで解析し、my\_script.md というMarkdownファイルとして出力します。

# my\_script.py がカレントディレクトリにある場合  
python explain\_program5.py my\_script.py

特定の出力ファイル名を指定する場合:

python explain\_program5.py my\_script.py my\_document.md

### 2. 複数のCファイルのドキュメントを生成（既存ファイルはスキップ）

src ディレクトリ内の全ての .c ファイルを処理し、各ファイルのドキュメントを src/\*.md として生成します。既に存在する .md ファイルはスキップされます。

# src/ ディレクトリに複数の .c ファイルがある場合  
python explain\_program5.py 'src/\*.c' --update 1 --overwrite 0

### 3. 既存ファイルを上書きして強制的にドキュメント生成

old\_program.py のドキュメントを再生成し、既存の old\_program.md を強制的に上書きします。

python explain\_program5.py old\_program.py --overwrite 1

### 4. Google Gemini APIを使って特定のプログラムタイプでドキュメント生成

tk\_ai\_core.py というファイル（ライブラリと自動判定される可能性が高いが、明示的に lib と指定）をGoogle Gemini Proモデルで解析し、ドキュメントを生成します。

python explain\_program5.py tk\_ai\_core.py --api google -t lib

### 5. OpenAI GPT-4 APIを使って特定のINIファイルでドキュメント生成

main.py をOpenAI GPT-4で解析し、custom\_prompts.ini に定義されたプロンプトを使用してドキュメントを生成します。

python explain\_program5.py main.py --api openai --inifile custom\_prompts.ini