

# プログラマ向け 活用テクニック 13

中西 克典

## リスト1 コード生成の指示と結果

全文はログ・ディレクトリの「コード生成.md」

```
[入力] 非負整数のリストを2進数表示したときの1の数が少ないものから順  
にソートする関数を書いてください。  
[GPT]  
def count_ones_in_binary(num):  
    # 2進数表現での1の数を数える関数  
    return bin(num).count('1')  
  
def sort_by_ones_count(lst):  
    # カスタムキー関数を使って1の数が少ない順にソート  
    sorted_list = sorted(lst,  
                          key=count_ones_in_binary)  
    return sorted_list
```

ChatGPTはプログラミングのさまざまな場面で活用できます。要件に合うコードを生成させることはもちろん、解説やドキュメントやテストを生成させたり、ライブラリの使い方を調査させたり、コード・レビューをさせて、コードに含まれる問題点を挙げて修正案を提案させたりすることもできます。

本章では、さまざまな活用方法を生成系/調査系/解説系の3系統に整理し、どのような場面でどのように活用できるかを具体例を交えて紹介します。まずは3系統のそれぞれにフォーカスした基本的な活用方法について紹介し、最後に3系統を組み合わせた複合系について紹介します。

本章のChatGPTとのやり取りはChatGPT (GPT-3.5)のチャット機能(APIでない)を用いたもので、そのやり取りの全文を、

```
https://github.com/kurusugawa-computer/interface-2023-11-chatgpt/prompts/  
各論編
```

に公開しています(ログ・ディレクトリと呼ぶ)。読みながら本文と同じ指示をChatGPTに入力する場合や、全文を確認する場合にご活用ください。

## 生成系

生成系は、ChatGPTにコードを書かせる最も基本的な活用方法です。

## リスト2 コード修正の指示と結果

```
[入力] リストに負の数が入っていないようにassertを追加してください。  
[GPT] assert all(num >= 0 for num in lst), "リストに負  
の数が含まれています。"
```

## ● コード生成

コード生成は、要件を指示してそれを満たすコードを生成する活用方法です。リスト1が指示と結果の例です。また、生成されたコードが期待したものではない場合は修正を指示します。リスト2は修正の指示の例とその結果の抜粋です。

最初の指示に条件を追加して再生成することでも同様のことができます。そうした方が試行錯誤しやすいため、お勧めです。

## 調査系

調査系は、ChatGPTの知識を使って調査する活用方法です。ChatGPTは驚くほど多くの知識を学習しているため、ぜひ試してほしい活用方法です。ただし、次の2点の注意があります。

- 最新情報は学習していない
- ハルシネーション(生成結果に誤りが含まれること)が起きる可能性がある

特に、ハルシネーションには注意が必要です。ChatGPTはもっともらしい文章を生成するため、専門家などの正しい知識を持った人間でない誤りを見抜けない場合があります。そのため、ChatGPTに頼るのは大枠の把握までとし、それ以降は、公式ドキュメントを見たり、専門家に聞いたり、動作確認や証明をしたりして、正確に把握しましょう。インターネット上の記事を参考にする場合も同様のことを行っているはずで、

## ● 検索

検索は、インターネットの代わりにChatGPTでツールやライブラリを検索する活用方法です。ChatGPTを用いると整理された説明がオンデマンド