自家用 AI でコード生成（Tabby）

目次

自家用 AI でコード生成（Tabby）

はじめに

機械学習（ML:Machine Learning）、深層学習（DL:Deep Leaning）、神経回路網（Neural Network）モデル、大規模言語モデル（LLM:Large Language Models）等々 AI 関連の言葉が日常で聞かれるようになり、プログラミングでも自然言語処理や自然言語生成（詳細は BERT や GPT の解説を参照）から派生したコード生成が実用化しています。

コード生成の論文・実装は GPT 以降も新しい発表が続いていますが、大まかには、コメントを含む大量の式や文をトークン（英文テキストでは≒単語）に分解し、神経回路を模したネットワークを通して特徴量や関連度<Self-Attention>を算出し（深層学習）、トークン間を関係付け正規化した事前学習済みの大規模言語モデルを作成します。これらの一連の作業が機械学習で、これに強化学習を加えてプログラミング言語の特徴を習得させます。

強化学習済の言語モデルから情報を取得するには利用側（クライアント）からプロンプトを与え、サーバ上でプロンプトをトークンに分解して言語モデルの中から関連度の高い情報群を選びます。

現在は各種のプログラミング言語に強化学習した言語モデルと、プロンプトをサーバに送るためのクライアント機能がオープンソースで利用できます。

1. コード生成の問題点と自家サーバ

【問題点①：情報セキュリティ】一般に、サーバに送られるプロンプトとしてエディターで開いているソースとコメントがそのまま使われ、言語モデルへの問合せに使われるとともに言語モデルの学習（データ収集）にも使われます。外部に設置されている言語モデル（サーバ）を使用する場合、これが情報保護上の問題になる場合があります。

【問題点②：内部ルール】GitHub のリポジトリ等から学習している言語モデルは自社内で開発したソフトウェアやコーディングルールを知らないので、これを基にしたコード生成をしてくれません。また、プログラミング言語の新しい機能や非推奨等のバージョン差異は関知しません（できません）。

【問題点③：日本語処理】言語モデルやプロンプトはトークンという単位で言語処理を行いますが、例えば英語であれば空白で区切られた単語をトークンにできるのに対して日本語は形態素解析が必要になります。日本語の形態素解析はオープンソースのソフトウェアがあり誰でも利用できますが、最初からサーバに組み込まれていなければ日本語の利用は困難です。

《解決策》①に関しては LAN にサーバを用意することで解決でき、経験値が上がるにつれて②の改善も期待できます。③に関してはオープンソースであれば自分で組み込むことも可能かもしれませんがリポジトリが頻繁に更新されるので自前のソース管理は現実的ではなく、国際化が考慮されたソフトウェアを選ぶ必要があります（アジア系の開発者が多いためか、試した範囲ではプロンプトに空白区切りの欧米系言語を推奨している言語モデルでも日本語のコメントでコード生成ができました）。

2. Tabby

機械学習だけでなくプロンプトの解析と大規模言語モデルの処理に強力な CPU/GPU とメモリが必要になるためコード生成のサーバをインターネット上で提供することが多いのですが、Tabby というソフトウェアは「ローカルホストに設置できる AI」を謳ったオープンソースです（商用利用や改変を認めている Apache-2.0 ライセンス）

2.1. サーバ

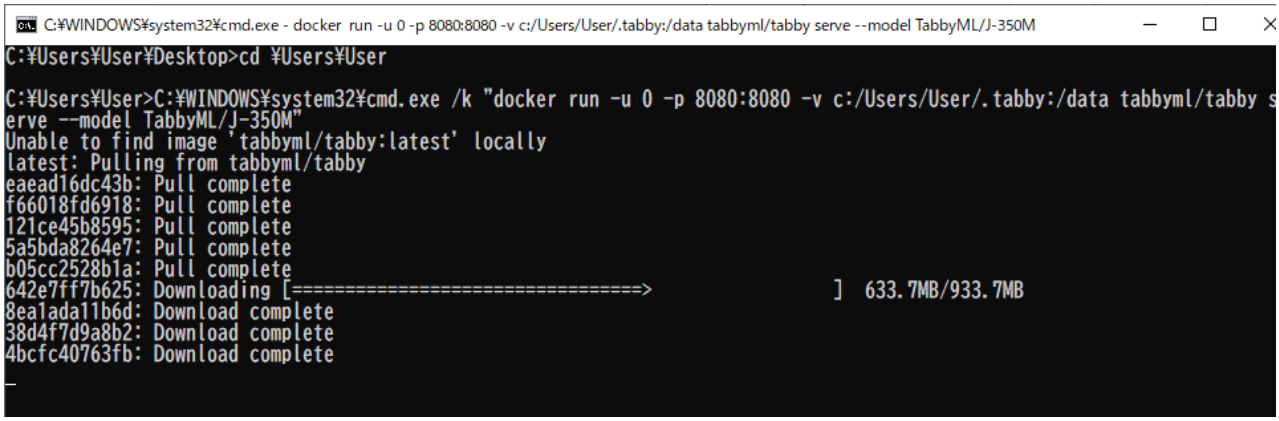
GitHub のリポジトリ[1]TabbyML/tabby で開発されています（2023/6 月時点でまだα版です）。

Tabby のドキュメントにはリポジトリの複製を作る説明もありますが、Windows 上の Docker で動作させる場合は以下のコマンドを実行するだけです。（公式ドキュメント[2]の Linux 用を読み替え）

```
docker run -u 0 -p 8080:8080 -v c:/Users/User/.tabby:/data tabbyml/tabby serve --model TabbyML/J-350M
```

User :利用者 ID を指定（c:/Users/*User* **言語モデルや設定の保存場所**…*ホームディレクトリ*¥.tabby）

TabbyML/J-350M:強化学習済の言語モデルを指定します（J-350M は動作確認用の軽量モデル）

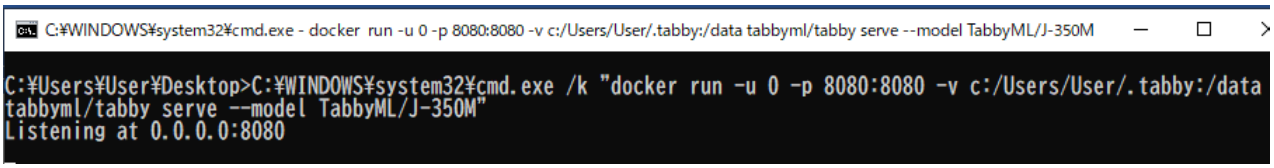


※ Docker と互換のある Rancher Desktop でも動作し、起動スクリプト例は以下になります。

```
@echo off
set HDP=%HOMEDRIVE%%HOMEPATH%
cd %HDP%
tasklist | find "Rancher Desktop.exe" > NUL
if %ERRORLEVEL% == 0 (
  REM Rancher Desktop.exe 実行中
) else (
  echo 先に rdctl.exe start を実行します。Rancher Desktop 起動後に再実行してください。
  echo Rancher Desktop の起動が完全に終わるまで数分待ってください。
  rdctl start
  goto END
)
C:¥WINDOWS¥system32¥cmd.exe /k "docker run -t -u 0 -p 8080:8080 -v %HDP%¥.tabby:/data
tabbyml/tabby serve --model TabbyML/J-350M"
:END
```

Docker イメージのダウンロード（初回のみ）とコンテナ起動後に Tabby のサーバが起動します。

ローカルホスト（IP アドレス 0.0.0.0）のポート 8080 でリクエストの待ち受けを開始します



[1] https://github.com/TabbyML/tabby

[2] Tabby Docker https://tabbyml.github.io/tabby/docs/self-hosting/docker

2.2. 強化学習済言語モデル

Tabby が使用するモデルは、公式ドキュメントに記載（2023/6 月末）の TabbyML/SantaCoder-1B と動作確認用の TabbyML/J-350M です（他にも Hugging Face のサイトに登録されています[3]）。

見た目の違いはファイルサイズで、これがサーバの応答時間に大きく影響します。

Microsoft Office とテキストエディター（VSCode）がまぁまぁ動作する程度で Windows 11 の対象にならない PC にサーバとクライアントを入れて J-350M を使った場合、サーバの応答に 10 秒程度かかります。SantaCoder-1B ではその倍程度を要しますが、ファイルサイズが大きなモデルを使った方がコード生成の多様性や返信のコード量が多くなります。

＜試行で使った PC のプロパティ＞

・プロセッサ　　Intel(R) Core(TM) i5-3230M CPU @ 2.60GHz　2.60 GHz

・実装 RAM 8.00 GB (7.88 GB 使用可能)

・システムの種類　　64 ビット オペレーティング システム、x64 ベース プロセッサ

＜言語モデルの仕様等＞

どのようなプログラミング言語に対応しているか等は、モデルの説明ページ[4]に掲載されています。

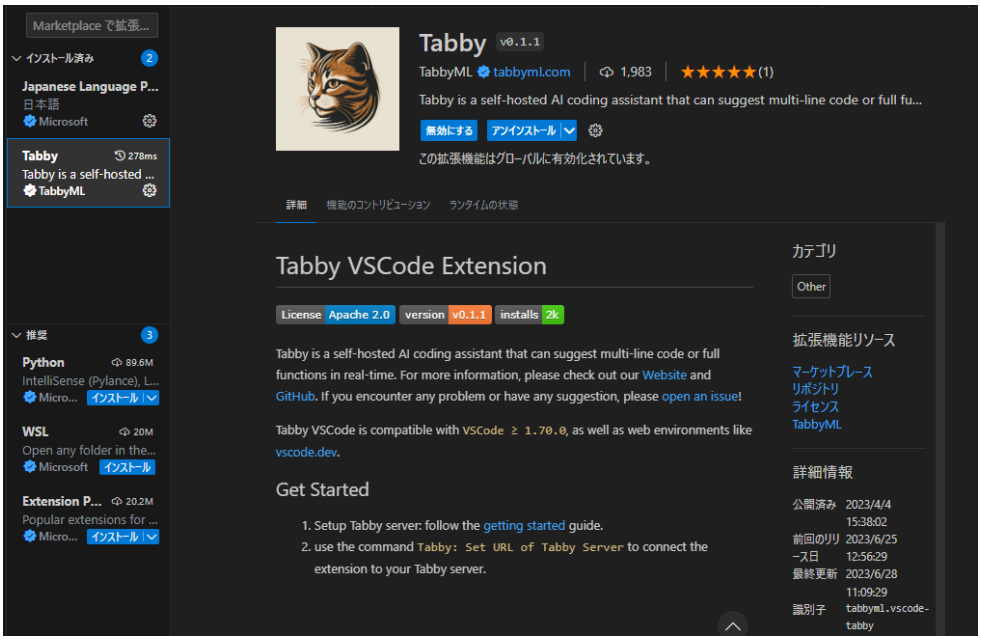
以下、SantaCoder-1B の概要から抜粋

・言語: Python、Java、JavaScript

・使用目的（一部）：コメント（例: # the following function computes the sqrt）などのソース コード内でコマンドを表現するか、関数シグネチャと docstring を記述してモデルに関数本体を完成させる必要があります。

2.3. クライアント

クライアント機能は幾つか開発中ですが、Visual Studio Code(VSCode)の機能拡張が一番簡単です。



[3] Tabby 用の言語モデル https://huggingface.co/TabbyML

[4] モデルの概要 https://huggingface.co/TabbyML/SantaCoder-1B

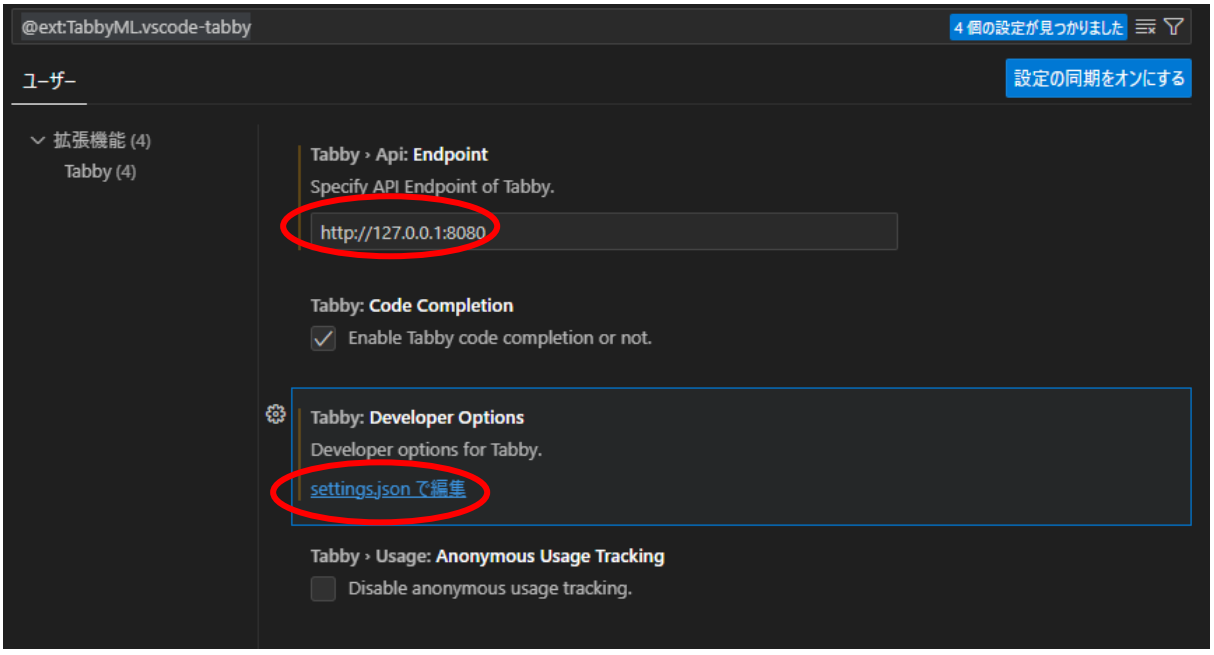
自家用 AI でコード生成（Tabby）

＜VSCode ver.0.1.1 の環境設定＞… 表示項目はバージョンにより変わる可能性があります。

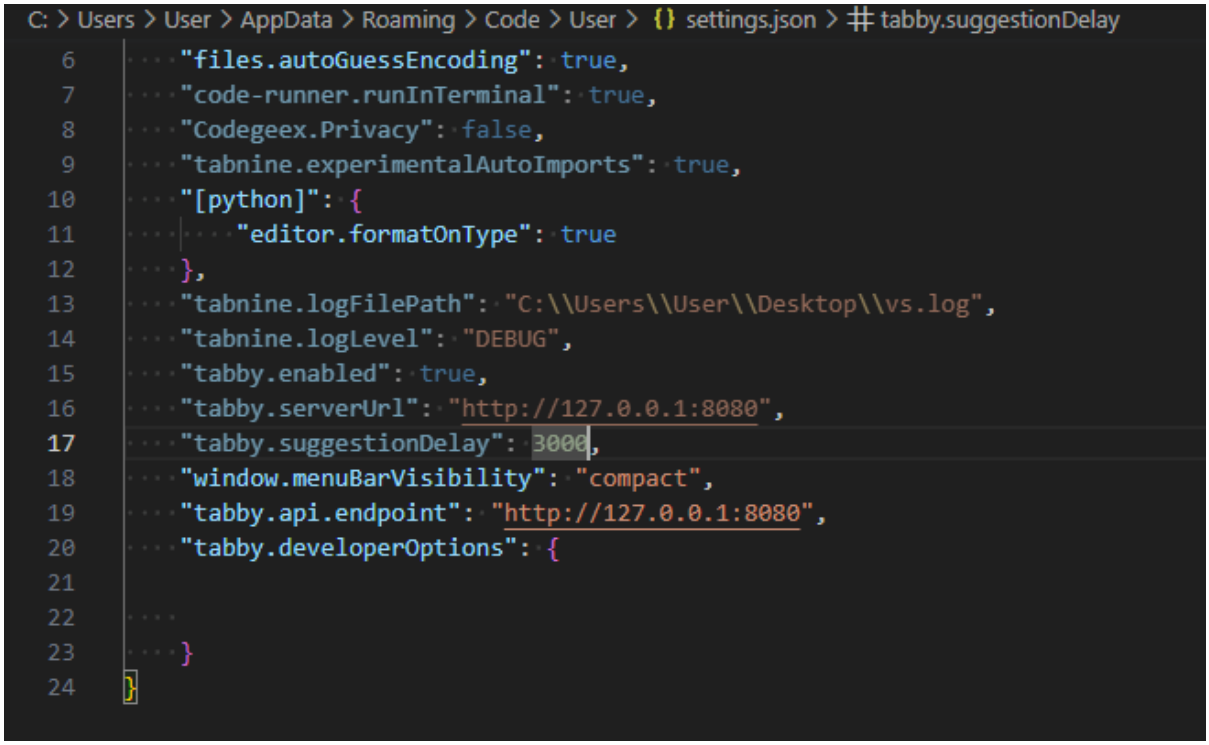
・Tabby・Api:Endpoint　…サーバの起動 URL とポート番号を指定します

・Developer Options　…tabby.suggestionDelay にサーバ問合せ間隔をミリ秒単位で指定します

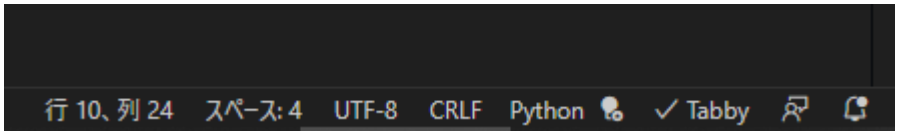
※ デフォルトは 1 秒間に数回サーバ問合せを行い、サーバ性能によってリクエストが滞留します



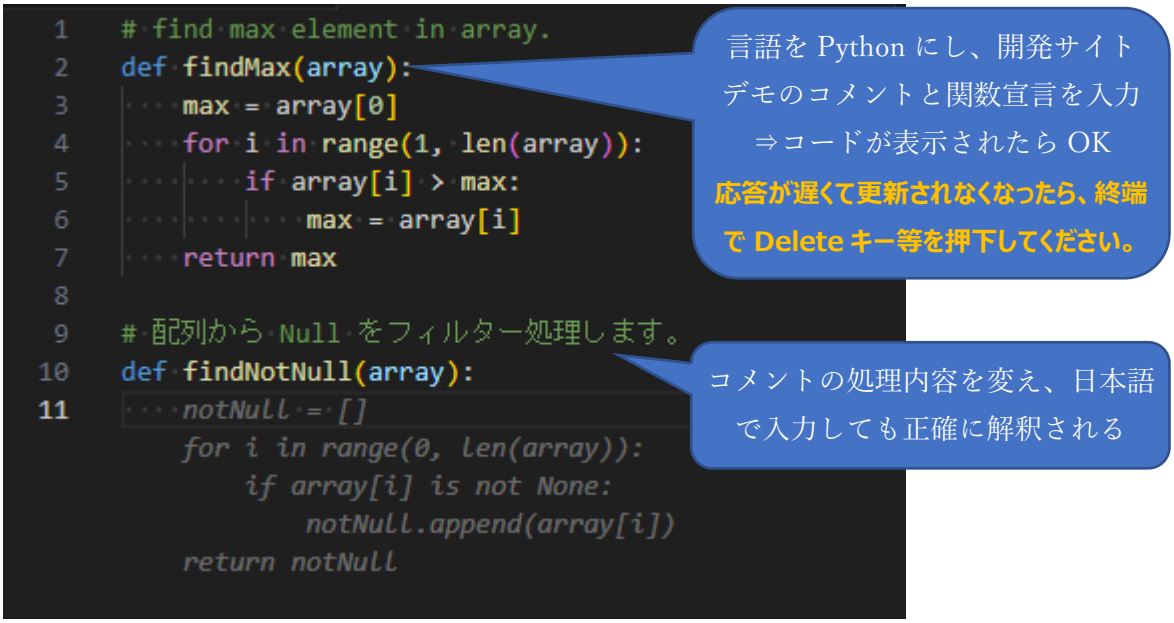
<Tabby:Developer Options　setings.json で編集 のリンク先> …問合せ間隔を 3 秒に設定した例



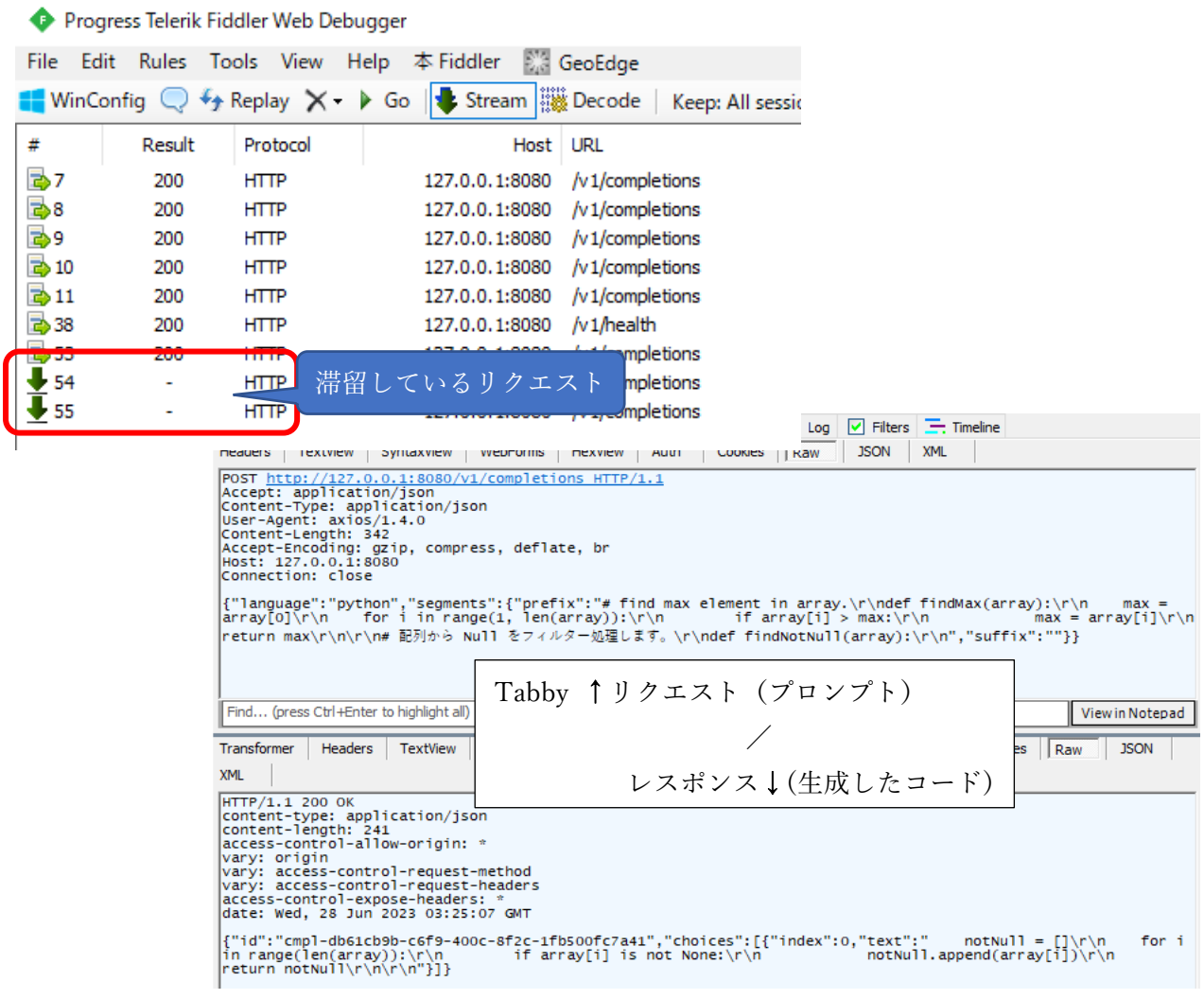
ステータスバーに 「✓Tabby」が表示されていればサーバとの接続ができた状態です



3. 動作確認



【注意】tabby.suggestionDelay の数値が小さすぎると、HTTP リクエストが滞留して結果が VSCode に反映されなくなります。（下図はフリーウェアの Fiddler4 でサーバ接続状態を確認したものです）



自家用 AI でコード生成（Tabby）

4. コード生成のパターン

① シグネチャによるコードの変化

```
String[] findNotNullStrings(String[] array){
    // 配列から Null をフィルター処理します。
    for (int i = 0; i < array.length; i++) {
      if (array[i] != null) {
        return array;
      }
    }
    return null;
  }
```

言語を Java に変えて動作したが、この例では処理内容が間違ってる
処理ブロックが閉じている"}"とコードは追加されません。

```
List<String> findNotNullStrings(String[] array){
    // 配列から Null をフィルターし、Listで返却します。
    List<String> result = new ArrayList<String>();
    for(String s : array){
      if(s != null){
        result.add(s);
      }
    }
    return result;
  }
```

return 型を変えたらイイ感じに…

```
List<String> findMaxElemet(String[] array){
    // 配列から Null をフィルターし、Listで返却
    List<String> list = new ArrayList<String>();
    for (int i = 0; i < array.length; i++) {
      if (array[i] != null) {
        list.add(array[i]);
      }
```

コメントはそのままメソッド名を変えたら処理方法が変わり、処理ブロックの半ばでコードの入力待ちになる

```
List<String> findMaxElemet(String[] array){
    // 配列から Null をフィルターし、Listで返却します。
    List<String> list = new ArrayList<String>();
    for (int i = 0; i < array.length; i++) {
      if (array[i] != null) {
        list.add(array[i]);
      }
    }
    return list;
  }

  public static void main(String[] args) {
    String[] array = { "a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "
```

Tabby の提案を受け入れ、処理ブロックを完結させると継続する提案が表示される

自家用 AI でコード生成（Tabby）

② コメント（日本語）の解釈

```
String findMaxElemet(String[] array){
    // 配列から最大の要素を選択して返却します。
    int max = 0;
    for (int i = 0; i < array.length; i++) {
      if (array[i].length() > max) {
        max = array[i].length();
      }
    }
```

“最大”の意味が期待した「値」ではなく「長さ」になっている

③ コメントの詳細化

```
// 配列の要素を数値に変換して比較し、最大の要素を返却します。
String findMaxElemet(String[] array){
    String max = "";
    int maxIndex = 0;
    for(int i = 0; i < array.length; i++){
        if(array[i] > max){
            max = array[i];
            maxIndex = i;
        }
    }
    return max;
}

// 比較します。
int compare(String a, String b){
    int aIndex = 0;
    int bIndex = 0;
```

コメントを詳細化すると少し見栄えが変わるが、それでも期待した数値での比較にはならない（文字列で比較するにしても、比較演算子使っちゃマズイ！）

5. コードの学習

コメントとコードの関係は保存されます。したがって、サーバを開発グループ全員が参照できる場所に設置すればコードパターンを共有することが可能になります。

```
// 配列の要素を数値に変換して比較し、最大の要素を返却します。
String findMaxElemet(String[] array){
    String max = array[0];
    BigDecimal maxDec = new BigDecimal(array[0]);
    for(int i = 1; i < array.length; i++){
        BigDecimal elmDec = new BigDecimal(array[i]);
        if(elmDec.compareTo(maxDec) > 0){
            max = array[i];
            maxDec = elmDec;
        }
    }
    return max;
}

// 配列の要素を数値に変換して比較し、最大の要素を返却します。
String findMaxElemet2(String[] array){
    String max = array[0];
    BigDecimal maxDec = new BigDecimal(array[0]);
    for(int i = 1; i < array.length; i++){
        BigDecimal elmDec = new BigDecimal(array[i]);
        if(elmDec.compareTo(maxDec) > 0){
            max = array[i];
            maxDec = elmDec;
        }
    }
    return max;
}
```

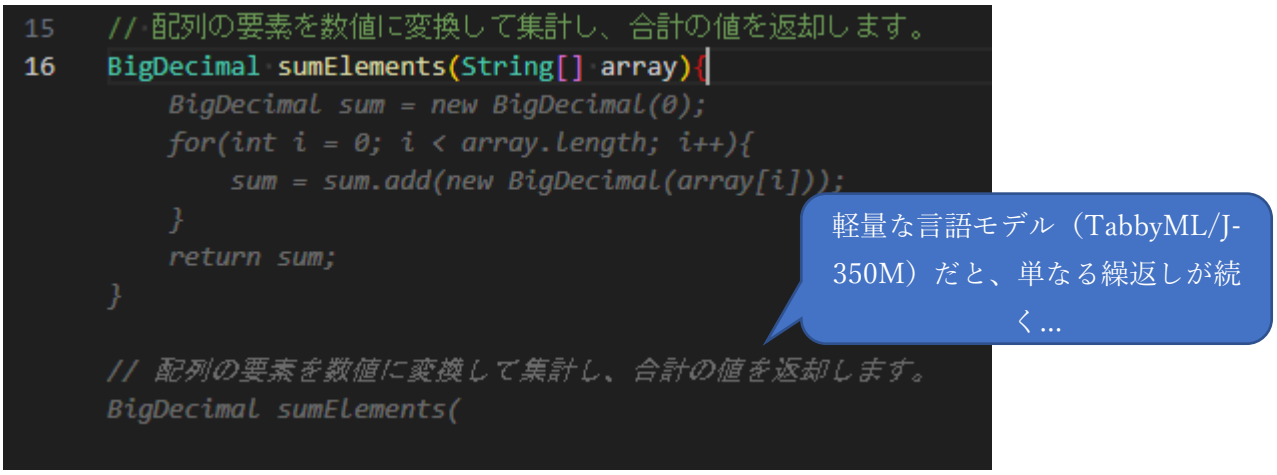
小数点付きの数値も扱えるコードに上書きすると…

同一コメントを使ったときに入力したコードが再現される（.cache_info.json が更新され、VSCode と Tabby を再起動しても残ります）

自家用 AI でコード生成（Tabby）

6. 言語モデルによる違い

使用する言語モデルの情報量が増えると提案されるコードも高度になり、AI 感が出てきます。



7. 使用上の注意点

Tabby は開発途上のため更新すると生成するコードも変わることがありますが、時には内容に問題を含む場合があります。例えば、以下の様なコードが出力される場合があります。

① 15 行目：compare メソッドが提案されているが使われていない。内容が文字列の比較になっているが、これなら String.compareTo()でより簡単に同様の結果が得られる

② コード内の以下の条件に矛盾があり、正常に判定されない（バグ）

ⅰ. 引数の a と b を文字列左端から比較し、a の方に小さい文字を見つけると-1 を返す（22 行目）

ⅱ. ⅰ.に該当せず、文字列長が同じか a の方が文字列が長かった場合に-1 を返す（31 行目）

```
1   // 配列の要素を数値に変換して比較し、最大の要素を返却します。
2   String findMaxElemet(String[] array){
3       String max = "";
4       int maxIndex = 0;
5       for(int i = 0; i < array.length; i++){
6           if(array[i] > max){
7               max = array[i];
8               maxIndex = i;
9           }
10      }
11      return max;
12  }
13
14  // 比較します。
15  int compare(String a, String b){
16      int aIndex = 0;
17      int bIndex = 0;
18      int aLength = a.length();
19      int bLength = b.length();
20      while(aIndex < aLength && bIndex < bLength){
21          if(a.charAt(aIndex) < b.charAt(bIndex)){
22              return -1;
23          }
24          else if(a.charAt(aIndex) > b.charAt(bIndex)){
25              return 1;
26          }
27          aIndex++;
28          bIndex++;
29      }
30      if(aIndex < aLength){
31          return -1;
32      }
33      if(bIndex < bLength){
34          return 1;
35      }
36      return 0;
37  }
```

自家用 AI でコード生成（Tabby）

コード生成ツールはエディターで編集中のコメントやコードと最も関連が近い情報を言語モデルで探して出力します。コメントを深読みしたり出力するアルゴリズムが正しいか否かの判断はしていません。今後 言語モデルのブラッシュアップで出力の精度や複雑さは増していくでしょうが、コードの適否は最終的にツール利用者が保障する必要があります。

利点としては、世界的なコードの保存場所（GitHub 等）で事前学習しているので情報交換サイトでコード例（スニペット）を探すのに近いことが手元でできる上に、使い続ければ自動的に組織固有の知識ベースとして充実していきます。また、キーワード等を計画的に使えば、仕様をコメントに書くだけでコードへ展開させるようなことができるかもしれません。

組織規模に応じた性能のサーバが必要なこと以外はマイナスがないので、ツールが常に最適な回答を出すわけではないことを理解したうえで個人でも組織でも早めに試す価値があります。

以上