

モジュール4

Azure OpenAI Serviceを使用してコードを生成する



Azure OpenAI Service を使用してコードを生成する

50 分 • モジュール • 7 ユニット

[フィードバック](#)

中級

AI エンジニア

開発者

ソリューション アーキテクト

学生

Azure OpenAI Service

Azure

このモジュールでは、Azure OpenAI Service を使用してコードを生成および改善する方法をエンジニアに示します。

学習の目的

このモジュールを完了すると、次のことができるようになります。

- 自然言語プロンプトを使用してコードを記述する
- AI モデルを使用して単体テストを構築し、複雑なコードを理解する
- 既存のコードのコメントとドキュメントを生成する

✓ 800 XP

モジュール4 コードを生成する

- コードの記述作業でAIはどのように役に立つのか？
- セッティング（モデル、システムメッセージ）
- (1)コードを記述してもらう
- (2)別のプログラミング言語で書き直してもらう
- (3)単体テストを作成してもらう
- (4)コードを説明してもらう
- (5)コメントを追加してもらう
- (6)コードのバグを指摘・修正してもらう
- (7)コードの効率を改善してもらう
- 知識チェック
- 講師デモ（省略）

私はプログラマです。私の作業を、あなたはどのように支援できますか？ できることを教えてください。



もちろんです。私はあなたのプログラミング作業を様々な方法で支援できます。以下にいくつかの例を挙げます：

1. **コードのデバッグ**：プログラムにエラーがあった場合、その原因を特定し、修正するための提案を行うことができます。
2. **コードの最適化**：あなたのコードを分析し、パフォーマンスを向上させるための提案を行うことができます。たとえば、無駄なループを削除する、より効率的なデータ構造を提案するなどです。
3. **新しい技術の学習**：新しいプログラミング言語やフレームワークを学ぶ際に、その基本的な概念や使用方法を教えることができます。
4. **コードレビュー**：あなたのコードをレビューし、可読性やメンテナンス性を向上させるための提案を行うことができます。
5. **自動化**：定型的なタスクを自動化するスクリプトを作成するお手伝いをすることができます。
6. **ドキュメンテーション**：あなたのコードのドキュメンテーションを作成するお手伝いをします。
7. **プロジェクト管理**：タスクの管理やスケジューリング、進行状況のトラッキングなどを支援することができます。

以上のような事柄に対して支援を行うことが可能です。具体的に何かお困りのことがあれば、お気軽にお申し付けください。

あなたは、どのようなプログラミング言語、スクリプト言語、フレームワーク、IaCなどに対応していますか？



...

私は多くのプログラミング言語、スクリプト言語、フレームワーク、IaCに対応しています。具体的には以下の様なものを含みます：

1. プログラミング言語: Python, Java, C++, C#, JavaScript, TypeScript, Go, Rust, Ruby, Swiftなど。
2. スクリプト言語: Bash, PowerShell, Perl, PHPなど。
3. フレームワーク: Django, Flask, Spring, Angular, React, Vue.js, Ruby on Rails, Laravelなど。
4. IaC(Infrastructure as Code)ツール: Terraform, Ansible, Chef, Puppet, AWS CloudFormationなど。

また、これらの技術に関する質問に答えるだけでなく、コードのデバッグ、コードレビュー、コード最適化の提案などの支援も提供できます。それぞれの言語やツールに関する具体的な知識は常にアップデートされており、新たな技術にも対応できるようになっています。

モジュール4 コードを生成する

- コードの記述作業でAIはどのように役に立つのか？
- セッティング（モデル、システムメッセージ）
- (1)コードを記述してもらう
- (2)別のプログラミング言語で書き直してもらう
- (3)単体テストを作成してもらう
- (4)コードを説明してもらう
- (5)コメントを追加してもらう
- (6)コードのバグを指摘・修正してもらう
- (7)コードの効率を改善してもらう
- 知識チェック
- 講師デモ（省略）

コードの生成に使用できるモデル

- Codex
 - code-cushman-001 / 002
 - code-davinch-001 / 002

- GPT-3.5

- gpt-35-turbo
- gpt-35-turbo-16k
- gpt-35-turbo-instruct

- GPT-4

- gpt-4
- gpt-4-32k

Completions API
(Completionsプレイグラウンド) で
利用可
※モデルバージョン0301 は、
2024/7/5に廃止予定

Completions API
(Completionsプレイグラウンド) で
利用可

コードの生成に使用できるモデル

- Codex
 - code-cushman-001 / 002
 - code-davinci-001 / 002

- GPT-3.5

- gpt-35-turbo
- gpt-35-turbo-16k
- gpt-35-turbo-instruct

- GPT-4

- gpt-4
- gpt-4-32k

これらはより新しい
Chat Completions API
(Chatプレイグラウンド) で
利用できるのでオススメ

以降、本資料では以下を使用します。

- モデル: gpt-4-32k (0613)
- プレイグラウンド: Chatプレイグラウンド

システムメッセージの設定例

システム メッセージ ⓘ

あなたはC#のプログラマです。|

本資料ではC#を例としますが、
生成AIモデル自体はPython、Java、JavaScriptなど、非常に多
くのプログラミング言語のコードを生成できます。

モジュール4 コードを生成する

- コードの記述作業でAIはどのように役に立つのか？
- セッティング（モデル、システムメッセージ）
- (1)コードを記述してもらう
- (2)別のプログラミング言語で書き直してもらう
- (3)単体テストを作成してもらう
- (4)コードを説明してもらう
- (5)コメントを追加してもらう
- (6)コードのバグを指摘・修正してもらう
- (7)コードの効率を改善してもらう
- 知識チェック
- 講師デモ（省略）

C#で二分探索を行うメソッドを記述してください。第1引数にint型の配列、第2引数に探索する数値、戻り値は探索する数値が見つかった配列の位置としてください。





以下にC#で二分探索を行うメソッドの例を記述します。

```
public static int BinarySearch(int[] array, int target)
{
    int start = 0;
    int end = array.Length - 1;

    while (start <= end)
    {
        int middle = (start + end) / 2;

        if (array[middle] == target)
        {
            return middle;
        }
        else if (array[middle] < target)
        {
            start = middle + 1;
        }
        else
        {
            end = middle - 1;
        }
    }

    return -1; // target not found
}
```



このメソッドは二分探索アルゴリズムを用いて配列内の目標値の位置を探します。目標値が見つかった場合はその位置（インデックス）を返し、見つからなかった場合は-1を返します。

なお、このメソッドを使用するためには配列がソートされている必要があります。

C#で、引数で与えた時刻と現在時刻との差を計算し、その差が分単位なら「N分前」、時間単位なら「N時間前」、日単位なら「N日前」といったように文字列化するメソッドを書いてください





以下に該当のメソッドを記述します。このメソッドは、指定した日時`pastTime`と現在時刻との差を計算し、適切な形式で返します。

```
using System;

public static string TimeDifference(DateTime pastTime)
{
    TimeSpan difference = DateTime.Now - pastTime;

    if (difference.TotalDays >= 1)
    {
        return $"{Math.Floor(difference.TotalDays)}日前";
    }
    else if (difference.TotalHours >= 1)
    {
        return $"{Math.Floor(difference.TotalHours)}時間前";
    }
    else if (difference.TotalMinutes >= 1)
    {
        return $"{Math.Floor(difference.TotalMinutes)}分前";
    }
    else
    {
        return "ちょうど今";
    }
}
```

このメソッドでは、まず`DateTime.Now`と`pastTime`との差（`TimeSpan`オブジェクト）を取得します。その差が1日以上なら"日前"、1時間以上なら"時間前"、1分以上なら"分前"という文字列を返します。それ以外の場合（つまり、指定された時刻が現在時刻とほぼ同じである場合）は"ちょうど今"という文字列を返します。

モジュール4 コードを生成する

- コードの記述作業でAIはどのように役に立つのか？
- セッティング（モデル、システムメッセージ）
- (1)コードを記述してもらう
- (2)別のプログラミング言語で書き直してもらう
- (3)単体テストを作成してもらう
- (4)コードを説明してもらう
- (5)コメントを追加してもらう
- (6)コードのバグを指摘・修正してもらう
- (7)コードの効率を改善してもらう
- 知識チェック
- 講師デモ（省略）

引数で指定された年齢が成年（18歳）以上かどうかを判定し、「成年」または「未成年」と表示するC#のコードを書いてください。





以下にC#を使用して年齢を判断するプログラムを作成しました。このコードでは、メソッドCheckAgeが年齢を引数として受け取り、18歳以上なら"成年"を、それ未満なら"未成年"を出力します。

```
using System;

class Program
{
    static void Main(string[] args)
    {
        Console.Write("Please enter your age: ");
        int age = Convert.ToInt32(Console.ReadLine());
        CheckAge(age);
    }

    static void CheckAge(int age)
    {
        if (age >= 18)
        {
            Console.WriteLine("成年");
        }
        else
        {
            Console.WriteLine("未成年");
        }
    }
}
```

このコードを実行すると、"Please enter your age: "というプロンプトが表示され、ユーザーが年齢を入力します。その年齢がCheckAgeメソッドに渡され、成年か未成年かが出力されます。

このコードをPythonで書き直してもらえますか？



もちろんです。以下に、Pythonを使用して同じ機能を実装したコードを示します。このコードでは、関数check_ageが年齢を引数として受け取り、18歳以上なら"成年"を、それ未満なら"未成年"を出力します。

```
def check_age(age):  
    if age >= 18:  
        print("成年")  
    else:  
        print("未成年")  
  
age = int(input("Please enter your age: "))  
check_age(age)
```



このコードを実行すると、"Please enter your age: "というプロンプトが表示され、ユーザーが年齢を入力します。その年齢がcheck_age関数に渡され、成年か未成年かが出力されます。

モジュール4 コードを生成する

- コードの記述作業でAIはどのように役に立つのか？
- セッティング（モデル、システムメッセージ）
- (1)コードを記述してもらう
- (2)別のプログラミング言語で書き直してもらう
- (3)単体テストを作成してもらう
- (4)コードを説明してもらう
- (5)コメントを追加してもらう
- (6)コードのバグを指摘・修正してもらう
- (7)コードの効率を改善してもらう
- 知識チェック
- 講師デモ（省略）

int型の配列の数値の合計を返すメソッドを書いてください。



以下に要求されたメソッドを示します。このメソッドでは、int型の配列を引数として受け取り、その要素の合計を返します。

```
public static int SumArray(int[] numbers)
{
    int sum = 0;
    for(int i = 0; i < numbers.Length; i++)
    {
        sum += numbers[i];
    }
    return sum;
}
```



このコードの単体テストを生成してください。テストケースは1件でOKです。



単体テストを作成する際には、通常はJUnitやJUnit5などのテストフレームワークを使用します。以下に、JUnitを使用した単体テストの例を示します。

```
using JUnit4;

[TestFixture]
public class SumArrayTests
{
    [Test]
    public void Test_SumArray()
    {
        // Arrange
        int[] numbers = {1, 2, 3, 4, 5};
        int expectedSum = 15;

        // Act
        int actualSum = SumArray(numbers);

        // Assert
        Assert.AreEqual(expectedSum, actualSum);
    }
}
```

このテストでは、まず期待される結果（expectedSum）を設定します。次に、テスト対象のメソッド（SumArray）を呼び出し、その結果を取得します（actualSum）。最後に、期待される結果と実際の結果が一致することを確認します。

なお、このテストを実行するにはJUnitの環境が必要となります。

モジュール4 コードを生成する

- コードの記述作業でAIはどのように役に立つのか？
- セッティング（モデル、システムメッセージ）
- (1)コードを記述してもらう
- (2)別のプログラミング言語で書き直してもらう
- (3)単体テストを作成してもらう
- (4)コードを説明してもらう
- (5)コメントを追加してもらう
- (6)コードのバグを指摘・修正してもらう
- (7)コードの効率を改善してもらう
- 知識チェック
- 講師デモ（省略）

以下のコードが何をするのか説明してくれますか？



```
for (int i = 1; i <= 100; i++)  
{  
    if (i % 3 == 0 && i % 5 == 0)  
    {  
        Console.WriteLine("FizzBuzz");  
    }  
    else if (i % 3 == 0)  
    {  
        Console.WriteLine("Fizz");  
    }  
    else if (i % 5 == 0)  
    {  
        Console.WriteLine("Buzz");  
    }  
    else  
    {  
        Console.WriteLine(i);  
    }  
}
```



このコードは、一般的に「FizzBuzz」と呼ばれるプログラムです。このプログラムは、以下のルールで1から100までの数を入力します：

- 数字が3の倍数であれば "Fizz" を出力します。
- 数字が5の倍数であれば "Buzz" を出力します。
- 数字が3と5の両方の倍数であれば "FizzBuzz" を出力します。
- それ以外の数字（つまり、3の倍数でも5の倍数でもない数字）はそのまま出力します。

この問題は、プログラミングの面接やプログラミングの初学者が論理的思考や基本的なプログラミング構造（ループ、条件分岐等）を理解しているかをテストするためによく用いられます。

C#で、入力された文字列がメールアドレス形式になっているかどうかを判定するメソッドを書いてください。



以下のように正規表現を用いたメソッドを書くことができます。このメソッドでは、「System.Text.RegularExpressions」を使用して、指定されたパターンがメールアドレスの形式に一致するかどうかを確認します。

```
using System.Text.RegularExpressions;

public bool IsEmailAddress(string email)
{
    // Email address pattern
    string pattern = @"^([\w\.-]+)@([\w\.-]+)(\.\w{2,3})$";

    // Check if the email matches the pattern
    if (Regex.IsMatch(email, pattern))
    {
        return true;
    }
    else
    {
        return false;
    }
}
```

このメソッドを使用して、入力された文字列がメールアドレス形式になっているかを判定できます。メールアドレスの正規表現パターンは非常に多様で、上記のパターンは一般的なものの一つです。必要に応じてパターンを変更してください。

このコードで使われている正規表現を分解してわかりやすく解説してもらえますか？



もちろんです、以下に各部分の説明を書きます。

正規表現は、`^([\w\.\-]+)@([\w\-]+)((\.\(\w\){2,3})+)$`となっています。

- `^`: 文字列の開始を表します。
- `([\w\.\-]+)`: `\w`は任意の単語文字（英数字とアンダースコア）を、`\.`はピリオド（ドット）を、`\-`はハイフンを表します。`[]`内の任意の文字が一致することを示します。`+`は直前の要素が1回以上繰り返すことを示します。つまり、この部分は単語文字、ピリオド、ハイフンが1回以上繰り返す部分に一致します。
- `@`: そのままの意味で、`"@"`文字に一致します。
- `([\w\-]+)`: ここも同様に、単語文字とハイフンが1回以上繰り返す部分に一致します。
- `((\.\(\w\){2,3})+)`: `\.`はピリオド（ドット）を表し、`(\w\){2,3}`は単語文字が2~3回繰り返す部分に一致します。さらにその全体が1回以上繰り返す部分に一致します（つまり、`.com`や`.co.jp`など）。
- `$`: 文字列の終了を表します。

全体として、この正規表現は一般的なメールアドレスの形式（例：`example@example.com`）に一致します。ただし、メールアドレスの形式は多様であり、すべての形式に対応する正規表現を作成するのは難しいです。また、メールアドレスの実在性を確認するには、別途サーバー側での検証が必要となります。

モジュール4 コードを生成する

- コードの記述作業でAIはどのように役に立つのか？
- セッティング（モデル、システムメッセージ）
- (1)コードを記述してもらう
- (2)別のプログラミング言語で書き直してもらう
- (3)単体テストを作成してもらう
- (4)コードを説明してもらう
- (5)コメントを追加してもらう
- (6)コードのバグを指摘・修正してもらう
- (7)コードの効率を改善してもらう
- 知識チェック
- 講師デモ（省略）




以下のコードの各行にコメントを追加してもらえますか？

```
OpenAIClient client = new(new Uri(openAIEndpoint), new AzureKeyCredential(openAIKey));  
var completionsOptions = new ChatCompletionsOptions()  
{  
    DeploymentName = engine,  
    Messages = {  
        new ChatMessage(ChatRole.User, prompt)  
    },  
    MaxTokens = 500,  
};
```



もちろんです、以下に各行に対する説明を追加します。

```
// OpenAIClient インスタンスを作成します。ここでは OpenAI のエンドポイントと認証キーを引数に指定しています。  
OpenAIClient client = new(new Uri(openAIEndpoint), new AzureKeyCredential(openAIKey));  
  
// ChatCompletionsOptions オブジェクトを作成します。これは OpenAI の会話モデルの設定を定義します。  
var completionsOptions = new ChatCompletionsOptions()  
{  
    // 使用する OpenAI のエンジン（モデル）の名前を指定します。  
    DeploymentName = engine,  
  
    // 会話の開始メッセージを設定します。ここでは、ユーザーからのプロンプト（入力）を指定しています。  
    Messages = {  
        new ChatMessage(ChatRole.User, prompt)  
    },  
  
    // 会話モデルの応答として生成される最大トークン数を設定します。トークンとは、テキストを分割した単位のことです。  
    MaxTokens = 500,  
};
```

このコードは、OpenAI の会話型AIモデルを使用して、ユーザーからのプロンプトに対するAIの応答を生成します。

モジュール4 コードを生成する

- コードの記述作業でAIはどのように役に立つのか？
- セッティング（モデル、システムメッセージ）
- (1)コードを記述してもらう
- (2)別のプログラミング言語で書き直してもらう
- (3)単体テストを作成してもらう
- (4)コードを説明してもらう
- (5)コメントを追加してもらう
- (6)コードのバグを指摘・修正してもらう
- (7)コードの効率を改善してもらう
- 知識チェック
- 講師デモ（省略）

以下のコードの問題点を指摘してもらえますか？

====

```
// 引数で与えられたint型配列の平均値を計算する
public static double CalculateAverage(int[] numbers)
{
    double sum = 0;
    for (int i = 0; i < numbers.Length; i++)
    {
        sum += numbers[i];
    }
    return sum / numbers.Length;
}
```





このコードは、配列が空（つまり長さが0）の場合に例外（DivideByZeroException）をスローする可能性があります。numbers.Lengthが0の場合、0で除算しようとするためです。

それを防ぐには、メソッドの始めに配列が空でないことを確認するチェックを追加すると良いでしょう。

以下に修正案を示します。

```
public static double CalculateAverage(int[] numbers)
{
    if (numbers == null || numbers.Length == 0)
    {
        throw new ArgumentException("numbers cannot be null or empty.");
    }

    double sum = 0;
    for (int i = 0; i < numbers.Length; i++)
    {
        sum += numbers[i];
    }
    return sum / numbers.Length;
}
```

上記のコードでは、配列がnullまたは空の場合にArgumentExceptionをスローします。これにより、メソッドを呼び出す際に配列が正しい形式であることが保証され、0での除算が回避されます。



以下のコードにはどのような問題点がありますか？

====

```
Console.Write("IDを入力してください: ");
string id = Console.ReadLine();

// 接続文字列
string connectionString = "...";
string sql = "SELECT * FROM YourTable WHERE ID = " + id;

using (SqlConnection connection = new SqlConnection(connectionString))
{
    connection.Open();
    using (SqlCommand command = new SqlCommand(sql, connection))
    {
        using (SqlDataReader reader = command.ExecuteReader())
        {
            while (reader.Read())
            {
                Console.WriteLine(String.Format("{0}, {1}", reader[0], reader[1]));
            }
        }
    }
}
```



...

このコードの主な問題点はSQLインジェクションの脆弱性です。ユーザーが入力したIDをそのままSQL文に組み込んでいるため、悪意のあるユーザーが特定の文字列を入力することで、予期しない動作を引き起こしたり、データベースの内容を読み取ったり、変更したりできてしまいます。

また、このコードではユーザがIDを正しい形式で入力することを前提としていますが、そうでない場合のエラーハンドリングがありません。ユーザーが文字列や特殊文字を入力した場合、プログラムがクラッシュする可能性があります。

これらの問題を解決するためには、SQLパラメータを使用してユーザ入力を適切にエスケープし、SQLインジェクションを防ぐべきです。また、ユーザー入力の検証とエラーハンドリングを追加することも重要です。

修正後のコードは以下のようになります：

```
Console.Write("IDを入力してください: ");
string id = Console.ReadLine();

// 接続文字列
string connectionString = "...";
string sql = "SELECT * FROM YourTable WHERE ID = @Id";

using (SqlConnection connection = new SqlConnection(connectionString))
{
    connection.Open();
    using (SqlCommand command = new SqlCommand(sql, connection))
    {
        command.Parameters.AddWithValue("@Id", id);

        using (SqlDataReader reader = command.ExecuteReader())
        {
            while (reader.Read())
            {
                Console.WriteLine(String.Format("{0}, {1}", reader[0], reader[1]));
            }
        }
    }
}
```

この修正により、ユーザー入力が安全にSQL文に組み込まれ、SQLインジェクション攻撃のリスクが低減されます。

モジュール4 コードを生成する

- コードの記述作業でAIはどのように役に立つのか？
- セッティング（モデル、システムメッセージ）
- (1)コードを記述してもらう
- (2)別のプログラミング言語で書き直してもらう
- (3)単体テストを作成してもらう
- (4)コードを説明してもらう
- (5)コメントを追加してもらう
- (6)コードのバグを指摘・修正してもらう
- (7)コードの効率を改善してもらう
- 知識チェック
- 講師デモ（省略）

1から100までの合計を求めるコード片を書いてください。



C#で1から100までの合計を求めるためのコードは次のようになります:

```
int sum = 0;
for (int i = 1; i <= 100; i++)
{
    sum += i;
}
Console.WriteLine("The sum of 1 to 100 is: " + sum);
```



このコードは、1から100までのすべての数値を加算し、結果を表示します。

このコードの計算をもっと効率的になるよう書き直せますか？



はい、実は数学的な公式を使って計算を効率的に行うことができます。1からnまでの合計は、次の公式で計算できます: $n*(n+1)/2$

この公式を使って、1から100までの合計を計算するC#のコードは次のようになります:

```
int n = 100;  
int sum = n * (n + 1) / 2;  
Console.WriteLine("The sum of 1 to 100 is: " + sum);
```



このコードは、ループを使わずに直接合計を計算しているため、効率的です。

以下のコードをもっと高速に実行できるように書き直せますか？



====

```
using System;
```

```
class Program
```

```
{
```

```
    static void Main()
```

```
    {
```

```
        float[] a = new float[4] { 1.0f, 2.0f, 3.0f, 4.0f };
```

```
        float[] b = new float[4] { 1.0f, 2.0f, 3.0f, 4.0f };
```

```
        float[] c = new float[4];
```

```
        for (int i = 0; i < 4; i++)
```

```
        {
```

```
            c[i] = a[i] + b[i];
```

```
        }
```

```
        Console.WriteLine($"({c[0]}, {c[1]}, {c[2]}, {c[3]})");
```

```
    }
```

```
}
```



...

以下は、C#のSystem.Numericsベクトルライブラリを使用して、ベクトル加算を行う高速なバージョンのコードです。ベクトルライブラリはSIMD命令を活用するため、一度に複数の計算を行うことができ、パフォーマンスが向上します。

注意：System.Numericsベクトルライブラリは.NET Framework 4.6以降で使用できます。

```
using System;
using System.Numerics;

class Program
{
    static void Main()
    {
        Vector4 a = new Vector4(1.0f, 2.0f, 3.0f, 4.0f);
        Vector4 b = new Vector4(1.0f, 2.0f, 3.0f, 4.0f);
        Vector4 c = a + b;
        Console.WriteLine($"({c.X}, {c.Y}, {c.Z}, {c.W})");
    }
}
```

このコードでは、ベクトルaとbの各要素を一度に追加しています。その結果を格納する新たなベクトルcが作成されます。

SIMDとはどのようなものですか？



SIMDは"Single Instruction, Multiple Data"の略で、一つの命令で複数のデータを同時に処理するというコンピュータの処理方式を指します。この方式は特に、大量のデータに対する同じ操作を一度に行う必要がある場合、例えば画像処理や音声処理、数値計算などにおいて高い効率を発揮します。

SIMD命令セットは、CPUのアーキテクチャによって提供され、例えばIntelのSSEやAVX、ARMのNEONなどがあります。これらの命令セットを直接利用することは難しいですが、多くのプログラミング言語やライブラリがこれらの命令セットを抽象化し、開発者が簡単に利用できるようにしています。

具体的には、C#のSystem.Numericsベクトルクラスなどがこれに該当します。これらを使用することで、開発者はSIMDを意識することなく、一度に複数のデータを処理する高速なコードを書くことができます。

モジュール4 コードを生成する

- コードの記述作業でAIはどのように役に立つのか？
- セッティング（モデル、システムメッセージ）
- (1)コードを記述してもらう
- (2)別のプログラミング言語で書き直してもらう
- (3)単体テストを作成してもらう
- (4)コードを説明してもらう
- (5)コメントを追加してもらう
- (6)コードのバグを指摘・修正してもらう
- (7)コードの効率を改善してもらう
- 知識チェック
- 講師デモ（省略）

知識チェック 第1問

Azure OpenAI を使用してコードを生成する利点は何ですか?

- a. 効率と生産性を向上させる
- b. バグと読みやすさの向上
- c. コーディングに費やされる時間の増加

知識チェック 第1問

Azure OpenAI を使用してコードを生成する利点は何ですか?

a. 効率と生産性を向上させる

b. バグと読みやすさの向上

c. コーディングに費やされる時間の増加

知識チェック 第2問

あるコードの続きを生成AIモデルに記述させる際、モデルにより多くの情報を提供すると、どうなりますか？

- a. より多くの情報を提供すると、モデルの正確性が低下します。
- b. より多くの情報を提供しても、モデルの正確性には影響しません。
- c. モデルに提供される情報が多いほど、応答の正確性が高くなります。

知識チェック 第2問

あるコードの続きを生成AIモデルに記述させる際、モデルにより多くの情報を提供すると、どうなりますか？

- a. より多くの情報を提供すると、モデルの正確性が低下します。
- b. より多くの情報を提供しても、モデルの正確性には影響しません。
- c. モデルに提供される情報が多いほど、応答の正確性が高くなります。

知識チェック 第3問

以下のうち正しい記述はどれですか？

- a. Azure OpenAI モデルは、コードが C# で記述されている場合にのみ、ある言語から別の言語にコードを翻訳できます。
- b. ある言語で記述されたコードがあり、それを別の言語にする必要がある場合、Azure OpenAI を使用するといくつかの言語にそれを翻訳できます。
- c. Azure OpenAI モデルでは、コードを Python から C# または Java にのみ翻訳できます。

知識チェック 第3問

以下のうち正しい記述はどれですか？

a. Azure OpenAI モデルは、コードが C# で記述されている場合にのみ、ある言語から別の言語にコードを翻訳できます。

b. ある言語で記述されたコードがあり、それを別の言語にする必要がある場合、Azure OpenAI を使用するといくつかの言語にそれを翻訳できます。

c. Azure OpenAI モデルでは、コードを Python から C# または Java にのみ翻訳できます。

モジュール4は以上です。

- 休憩しましょう。