カリキュラム1
IT リテラシー・AI の基本

第 5 回：簡単なプログラミングと AI アシスタント活用

～プログラミングの基本を学ぶ・AI アシスタントを活用する～

目次

第 1 章プログラミングの基本概念

1. はじめに

この章では、プログラミングの基本概念について、一緒に学んでいきます。

「プログラム」という言葉を聞くと、なんだか難しそうに感じる方も多いかもしれません。

実は、実はプログラムの考え方は、日常生活の中にもたくさん取り入れられています。

私たちの身近な行動の中にも、順番やルール、判断、繰り返しといった要素があります。

そうした考え方こそが、プログラミングの基本です。

2・プログラムとは？

これから「プログラムとは何か？」について学びます。

プログラムという言葉を聞くと、「難しそう」「専門家のもの」と思う方もいるかもしれません。

けれど、実は私たちの身の回りには、プログラムによって動いているものがたくさんあります。

スマートフォンのアプリ、洗濯機の自動洗浄、自動販売機の動作、LINE でのメッセージ送信

―― そのすべてに、裏ではプログラムが働いています。

プログラムは、私たちの生活の中にすでに自然に溶け込んでいるのです。

では、プログラムとは何か？　それは、コンピュータに「やってほしいことを順番に伝えるための命令の集まり」です。人間なら、話しかければある程度察してくれますが、コンピュータはそうはいきません。「一つずつ、正確に、間違いなく」指示してあげる必要があります。そのためには、「考えを整理して順序立てる力」が必要になります。

これを「プログラミング的思考」といいます。

そして、プログラムの設計図としてよく使われるのが「フローチャート」です。※動画参照

これは、流れを図で示したもので、処理の始まりから終わりまで、どのような順序で何をするかをわかりやすく視覚化できます。

「もしこうだったら、こうする」「それ以外だったら、別のことをする」といった分岐処理や、「この作業を繰り返す」といったループ処理なども、図で整理すれば頭の中がすっきりします。

プログラムの基本的な構造は、次の 3 つに分けられます。

1.「順次処理」

上から下へ命令を 1 つずつ実行する基本的な流れです。

2.「分岐処理」

条件によって処理の流れを変えることができます。たとえば、「雨が降っていたら傘を持つ、晴れていたら帽子をかぶる」といった判断です。

3.「反復処理」

同じ処理を繰り返す命令で、たとえば「5 人分のデータを順番に処理する」などの場面で使います。

これらの要素を組み合わせることで、私たちはより柔軟で高度な処理を作ることができます。

複雑に見えるアプリや Web サービスも、実はこの 3 つの組み合わせによって成り立っているのです。たとえば、自動販売機の動作は順次処理と条件分岐、繰り返しが組み合わさった典型例です。

「お金を入れる」「選んだボタンを確認する」「商品を出す」「お釣りを返す」

この一連の流れも、すべてプログラムで制御されています。

もうひとつ例を挙げると、LINE でメッセージを送るとき。ボタンを押すと、相手にメッセージが届き、既読がつきます。このようなシンプルな操作の裏にも、細かく設計された命令の流れがあります。そのひとつひとつが、正しく順序立てて書かれていなければ、動作はうまくいきません。

ここで大切なのは、プログラムは「考える力を鍛える道具」でもあるということです。どんな順序で、どんな条件で、どれだけ繰り返せば目的が達成できるか――それを考えること自体が、論理的思考力や問題解決力を育ててくれます。

また、プログラミングはチームワークや他者との意思疎通にも役立ちます。

フローチャートを共有したり、手順を明文化することは、仕事の効率化にも直結します。「誰が見ても分かる」「あとから見返しても意味が通じる」設計を心がけることが、良いプログラムを作る基本です。

では、AI が進化した今、プログラミングは必要なくなるのでしょうか？

実は、逆に「AI 時代だからこそ必要」とも言われています。

ChatGPT のような生成 AI に「こんなコードを書いて」と頼むことは可能になりましたが、そのためには「何をしたいのか」「どうしてそうしたいのか」を明確に伝える必要があります。つまり、AI に対して“意図を正確に説明する力”が、今まで以上に重要になっているのです。

この力は、まさにプログラミング的思考の延長線上にあります。

AI は、こちらの曖昧な考えを勝手に汲み取ってくれるわけではありません。「入力があいまいなら、出力もあいまいになる」というのが AI の基本です。だからこそ、「処理をどう組み立てれば目的が達成されるか」を自分で考える力を持っていると、AI に対しても効果的に指示を出せるようになるのです。

このように、プログラムを書くことは「コードを書く」ことにとどまらず、「構造的に考える」「目的から逆算する」「正しく伝える」という、どんな分野でも活かせる力を育てることにつながります。

そしてこれは、仕事だけでなく、家庭でも役に立つ力です。たとえば、家事の手順を効率よく組み立てたり、子どもに物事を順序立てて説明したり。日常の中にある「プチ・プログラミング」は、思っているよりずっと多いのです。

さらに、年齢や職種を問わず、プログラミングは誰にとっても必要な「21 世紀型スキル」と言われるようになりました。文字が読めて書けるようになるように、思考を順序立てて伝えられる力も、現代社会では基本的なリテラシーとされています。学ぶタイミングに遅すぎるということはありません。むしろ、これからの社会を生き抜くために、誰もが知っておいて損はない考え方なのです。

プログラムとは、コンピュータに命令を出す手段であり、順次処理・条件分岐・繰り返し処理という 3 つの構造で成り立っています。

それを分かりやすく整理するために使うのがフローチャートです。

そして、プログラミングを学ぶことは、単に技術を習得するだけでなく、考える力、伝える力、そして問題を解決する力を育てることに直結しています。

このあと実際にコードを書いていく中でも、「考えを順序立てて伝える」ことを意識してみてください。それができれば、きっと AI 時代の強力な武器となって、皆さんの味方になってくれるはずです。

3・Python の基本構文

それでは、ここからは実際にプログラミングの基本となる構文について、Python という言語を使いながら、学びましょう。

Python、パイソンと読みますが、これは世界中で非常に人気のあるプログラミング言語の一つで、特に初心者にとって学びやすい構文であることから、教育現場やデータ分析の分野、さらには AI や Web アプリの開発まで、非常に幅広く使われています。

では、なぜ Python がこんなにも選ばれているのでしょうか。

それは「コードの見た目がシンプル」「書く量が少ない」「文法が直感的」など、プログラミングに不慣れな人にとってのハードルを低くしてくれる工夫が多く含まれているからです。

さて、プログラムというのは、基本的に「何かの処理を順番に書き並べたもの」です。その処理には、「データを保存する」「条件によって処理を分ける」「何度も繰り返す」など、いくつかの基本的なパターンがあります。

Python では、これらを非常にシンプルな書き方で表現することができます。

ここでは、3つの基本構文、「変数」「条件分岐」「繰り返し（ループ）」について順番に見ていきましょう。

1.「変数」

「変数」という概念について見てみましょう。

変数とは、データを一時的に保管しておくための箱のようなものです。

日常生活に置き換えるなら、「財布」がわかりやすい例です。

財布にはお金を入れておけますよね。財布そのものは「変数」、中に入っているお金は「値」だと考えるとイメージしやすくなります。

Python では、変数に値を入れるときは次のように書きます。

```
name = "Taro"

age = 18
```

この例では、「name」という変数に「Taro」という名前の文字列を入れ、「age」という変数に「18」という整数を入れています。

ここで重要なのは「=（イコール）」です。

算数で使う「等しい」という意味ではなく、Python では「代入」を意味します。

つまり「右の値を左の変数に入れる」という命令です。

このように変数を使っておくことで、あとからプログラム内で何度でも再利用することができます。

たとえば、こんなふうに使います。

```
print("こんにちは、", name, "さん！")

print("あなたの年齢は", age, "歳ですね。")
```

こうすると、「こんにちは、Taro さん！あなたの年齢は 18 歳ですね。」というような出力がされます。

2.「条件分岐」

次は、「条件分岐」、つまり「もし～ならば～する」という処理です。

これは実生活でもとても身近な考え方です。

たとえば、「もし雨が降っていたら傘を持っていく」「もし冷蔵庫に卵がなかったら買いに行く」といったように、私たちは常に条件によって行動を変えています。

Python でこのような条件分岐を表すには、「if」というキーワードを使います。

次のように書いてみましょう。

```
if age >= 18:
    print("大人です")
else:
    print("未成年です")
```

このコードは、「もし age が 18 以上なら『大人です』と表示し、それ以外なら『未成年です』と表示する」という意味です。

このように、条件によって処理を変えることができるのが条件分岐です。

また、条件が複数ある場合には「elif（else if）」という構文も使えます。

```
if age >= 65:
    print("高齢者です")
elif age >= 20:
    print("成人です")
else:
    print("未成年です")
```

このように、「上から順に条件を評価し、最初に当てはまったものだけを実行する」という動きになります。

条件式の中でよく使われる演算子には以下のようなものがあります。

- ==：等しい
- !=：等しくない
- >：より大きい
- <：より小さい
- >=：以上
- <=：以下

これらを組み合わせて、さまざまな条件を設定することができます。

3.「繰り返し」

次に、「繰り返し」、つまり「ループ」について紹介しましょう。

プログラムでは、「同じ処理を何度も実行する」ことがよくあります。

たとえば、「10 回拍手する」「100 人の名前を順番に表示する」など、手作業では大変なことも、繰り返しの命令を使えば一瞬です。

Python には大きく分けて2種類の繰り返し構文があります。

ひとつ目は「for 文」、もうひとつは「while 文」です。

まずは、代表的な「for 文」を見てみましょう。

```
for i in range(5):
    print("こんにちは")
```

このコードでは、「こんにちは」というメッセージが 5 回表示されます。

「range(5)」というのは、0 から 4 までの 5 つの数字を順番に生成してくれる関数です。

つまり、i という変数に 0,1,2,3,4 が順番に入り、そのたびに print の中身が実行されます。

繰り返し処理の中で変数 i を利用すれば、次のような出力も可能です。

```
for i in range(1, 6):
    print(i, "回目のご挨拶です")
```

この記述で、「1 回目のご挨拶です」から「5 回目のご挨拶です」までが表示されます。

もう一つの繰り返し構文である「while 文」は、ある条件が成り立つ限り、処理を繰り返すという形式です。

たとえば、次のようなコードがあります。

```
count = 0

while count < 3:

    print("ループ中です")

    count += 1
```

この例では、「count」が 3 未満の間はループを続け、毎回「ループ中です」と表示して、「count += 1」で 1 ずつ増やしていきます。

条件が満たされなくなると、ループは自動的に終了します。

「while 文」は「いつ終わるかわからないような繰り返し」に使われることが多く、例えばユーザーの入力を待ち続けるプログラムや、ある条件を満たすまで処理を続けたい場合に便利です。

以下は、これらすべてを使って簡単なミニプログラムを作った見本です。

このプログラムは、1 から 20 までの数字のうち、3 の倍数に対して特別なメッセージを表示するものです。

```
for i in range(1, 21):

    if i % 3 == 0:
```

```
        print(i, "は 3 の倍数です！")
    else:
        print(i)
```

このように、変数、条件分岐、ループを組み合わせることで、処理を自在に制御することができます。

プログラミングは一見複雑に見えるかもしれませんが、このように基本的なパターンを押さえておけば、意外とすぐに使いこなせるようになります。

Python の良いところは、文法がとにかくシンプルで、英語に近い感覚で書けるという点です。

「何をしたいか」がそのままコードに表れるので、まるでコンピュータと会話しているような感覚になります。

ぜひ、今日学んだ変数・条件分岐・ループの使い方をベースに、自分なりのミニプログラムを作ってみてください。

4. まとめ

ここまで、プログラミングの基本について、一緒にじっくりと学んできました。

初めての方にとっては、聞き慣れない言葉や新しい考え方も多かったかもしれません。

ですが、今日学んだ内容は、プログラミングを理解するうえで、とても大切な「土台」となる部分です。

最初にご紹介したように、プログラムとは、コンピュータに対して「何をどうしてほしいか」を順番に命令として書き並べたものです。
この考え方は、実は私たちの生活の中にもたくさんあります。
たとえば、朝の支度、料理の手順、機械の操作手順なども、プログラムのように順を追って動いているわけです。

その流れを整理する方法として、「フローチャート」という図を使えば、複雑な手順も目で見て理解しやすくなります。
自分の頭の中のアイデアを図にして整理できるのは、プログラミングだけでなく、日常の問題解決にも役立ちます。

そして、今回は Python というプログラミング言語を使って、変数・条件分岐・ループという三つの基本構文を見てきました。

変数はデータを一時的に記憶しておく「箱」のようなもの。
条件分岐は「もし〜なら〜する」といった判断。
ループは「同じことを何度も繰り返す」処理でした。

この 3 つを組み合わせることで、より柔軟で現実的な処理ができるようになります。

大切なのは、「まずやってみること」です。
小さなコードでも、自分の手で書いて、実際に動かしてみることで、「なるほど、こうなるのか」と実感できます。
それはまるで、自転車に初めて乗ったときのような感覚かもしれません。
転びながらも少しずつ前に進んで、やがてスムーズに走れるようになる。
プログラミングも、それにとてもよく似ています。

今回の内容を参考に、ぜひご自宅や空いた時間に、簡単なプログラムを書いてみてくださ

い。

たとえば、「こんにちは」と表示させるだけでも、立派な第一歩です。

そして次回は、いよいよ Python のコードを実際に実行する体験に入ります。

学んだことが目の前で動き出す瞬間は、とても楽しく、感動もひとしおです。

プログラムが動いたときの喜びを、ぜひ一緒に味わいましょう。

第2章2・簡単なコードを実行

1.はじめに

ここから、Python というプログラミング言語を使って、「コードを書く」そして「実際に動かしてみる」体験を一緒にしていきましょう。

「え、自分でコードなんて書けるのかな…」と不安な方も、まったく問題ありません。

今回は、初心者の方でも簡単に使える、無料でインストール不要のツール「GoogleColab」と、コードの相談相手としてとても便利な「chatGPT」を使います。

準備は不要。ブラウザさえあれば OK です。

さあ、リラックスして、ちょっとしたワクワクを感じながら、一緒にコードの世界をのぞいてみましょう！

2. Google Colab で Python を実行しよう

これから、Python というプログラミング言語を使って、「コードを書く」そして「実際に動かしてみる」の実習をしていきましょう。

「え、自分でコードなんて書けるのかな…」と不安な方も、まったく問題ありません。

今回は、初心者の方でも簡単に使える、無料でインストール不要のツール「GoogleColab」と、

コードの相談相手としてとても便利な「chatGPT」を使います。

「Google Colab を使った Python の基本実行」について学んでいきましょう。

プログラミングが初めての方や、「プログラミングってなんだか難しそう…」と感じている方も、安心してください。ゆっくり、ひとつずつ、実習を交えながら進めていきます

最近、テレビやネットのニュース、SNS などで「AI（人工知能）」や「Python（パイソン）」という言葉を耳にする機会が増えていませんか？

「なんだかすごそう…」「便利らしいけど、自分には関係なさそう…」そう思った方もいらっしゃるかもしれません。

実は、この Python というプログラミング言語は、今、世界中のあらゆる業界で注目されている、いわば“現代の共通言語”のような存在です。

AI の開発はもちろん、会社の業務を効率化したり、ホームページやアプリを作ったり、さらにはグラフや表を自動でつくってくれるツールの中にも Python が使われています。

とはいえ、これまで「プログラミングなんて触ったこともない」という方からすると、「コードを書く」「命令を出す」「関数ってなに？」といった言葉を聞いただけで、すこし身構えてしまうかもしれません。

「真っ黒な画面に、ずらっと並んだ英語と数字…。なんだか映画のハッカーみたいで、自分には無理かも…。」

そんなイメージをお持ちの方も、どうぞご安心ください。

なぜなら、今日使うのは「Google Colab（グーグル・コラボ）」という、誰でも無料で使える便利なツールを使用します。Google Colab は、パソコンに特別なソフトをインストールする必要がなく、インターネットにさえつながっていれば、すぐに Python のコードを書いて、動かしてみることができます。

たとえば、ワードやエクセルを開くのと同じくらい気軽に、Python をはじめられる。

それが、この Google Colab の大きな魅力です。

しかも、実行した結果がすぐにその場で表示されるので、「自分で書いた命令が、ちゃんと動いた！」という実感をすぐに味わうことができます。

みなさんは人生で初めて、プログラムを書いて実行する、という新しい一歩を踏み出します。

それはとても小さな一歩かもしれませんが、「できた！」という実感と、「もっとやってみたい！」という気持ちは、これからのみなさんの学びにとって大きな原動力になるはずです。

難しいことは、あとでゆっくり覚えていけば大丈夫。

まずは、やさしい言葉とシンプルなコードで、“プログラミングってこんな感じなんだ”という感覚を味わってみましょう。

それでは、はじめての Python 体験。いよいよスタートです！

こからは実際に Python を動かしてみるためのツール、「Google Colab（グーグル・コラボ）」についてご紹介していきます。

Google Colab は、Google が提供している無料のサービスで、正式名称は

「Google Colaboratory（グーグル・コラボラトリー）」といいます。

この“Colaboratory”という言葉は、「Collaborate（協働する）」と「Laboratory（研究室）」を組み合わせた造語で、「みんなで一緒に学べる・試せる・作れるノートブック」という意味合いが込められています。

最大の特徴は、なんといっても“インストール不要”であること。

パソコンに特別なソフトウェアを入れたり、難しい設定をする必要は一切ありません。

Google アカウントさえあれば、誰でもすぐに使い始めることができるんです。

たとえば、普段お使いの Chrome などの Web ブラウザさえ開けば、もう準備完了です。

これは、Python の勉強を始めたいと思った方にとって、とても大きなメリットです。

というのも、一般的にプログラミングを始めようとすると、「開発環境の構築」という少し複雑なステップが必要になることがあります。

しかし、Google Colab ではそうした“準備作業”をすっ飛ばして、すぐに「書いて、動かす」という体験に入ることができるんですね。

また、もう一つの便利なポイントは、「クラウド上で動いている」ということ。

つまり、作成したファイルは自動的に Google ドライブに保存されるので、途中でパソコンの電源が切れても安心。

職場や自宅、外出先など、どの場所からでも同じノートブックにアクセスできます。

たとえるならば、「いつでも取り出せる、自分だけの学習ノートがネットの中にある」といったイメージです。

では、実際の画面の使い方をイメージしてみましょう。

Google Colab のノートブックは、上半分に“コードを書く欄”、その下に“実行結果が表示される欄”という構成になっています。

このように、コードを書いたらすぐ下に結果が現れる、というスタイルは、初めてプログラムを触る人にとって非常にわかりやすい構造です。

たとえば、みなさんが何か質問を書いたら、その場で AI がすぐに答えてくれる。

そんな「質問と答えが同じ紙の上に並んで出てくる」ような感覚が、Google Colab の大きな魅力です。

また、ノートブックには「テキストセル」と「コードセル」があり、説明文とプログラムを混ぜて書くことができるため、学習や記録にも非常に適しています。

プログラミングの学習サイトや YouTube などでも、この Google Colab を使った解説が多く見られますので、今後の学習の中でもとても役に立つツールになりますよ。

◆実習

では、ここから実際に、Google Colab を立ち上げてみましょう。

まず、Google 検索で「Colab」と入力してみてください。すると、検索結果の上の方に「Google Colaboratory」というサイトが表示されると思います。

●そのページにアクセスして、「新しいノートブックを作成」というボタンをクリックします。

●しばらくすると、白くてシンプルなノートブック画面が表示されるはずです。

※これが、みなさんがこれから Python を書くことになるページです。

ノートにペンで書くように、ここにコードを入力し、再生ボタンのようなマークをクリックするだけで、Python が動いてくれます。それではいよいよ、最初の Python コードを書いてみましょう。

1,画面のセルと呼ばれる入力欄に、次のように入力してください。

```
print(″こんにちは、Python！″)
```

この print というのは、“表示する”という命令です。

●カギカッコの中に入っている文字列が、画面に出力されます。

●入力ができたら、左側にある再生ボタンを押してみてください。

下に「こんにちは、Python！」と表示されましたか？

これが、はじめての Python プログラムの実行です。

プログラミングの醍醐味は、こうした「自分で指示を出して、思った通りに動かせる」ことです。

2.次は、「変数（へんすう）」です。

変数とは、“値に名前をつけて覚えさせる”機能です。

●次のように書いてみましょう。

```
name = "さくら"
age = 25
```

ここでは、name という名前に「さくら」、age という名前に「25」という情報を保存しています。

Python はこれを記憶してくれるので、あとから name や age と入力するだけで、その情報を呼び出せます。

●次は、この変数を使って、プログラミングしてみましょう。

```
print("こんにちは、" + name + "さん")

print("あなたの年齢は", age, "歳ですね")
```

実行すると…

「こんにちは、さくらさん」「あなたの年齢は 25 歳ですね」と表示されるはずです。

ここで大切なのは、+ や , を使って文字列をつなげたり、変数を差し込んだりできるということです。

コンピュータに“意味”を伝えるために、こうした文法ルールがありますが、ひとつずつ覚えていけば大丈夫です。

3.次は、「条件分岐（じょうけんぶんき）」です。

これは、「もし～だったら、こうしてね」と指示するための仕組みです。

●年齢によって、表示内容を変えたい場合には次のように書きます。

```
if age >= 20:

    print(name + "さんは大人ですね")
```

```
else:
    print(name + "さんは未成年ですね")
```

この if は「もし～ならば」、else は「それ以外なら」という意味です。

Python は age >= 20 を確認して、条件に当てはまる方の命令を実行します。

このように、条件分岐を使うことで、「場合によって動作を変える」という柔軟な指示ができるようになります。

4.次に、「ループ」について学びましょう。

ループとは、“同じ処理を何度も繰り返す”仕組みです。

●Python では、for という命令を使って次のように書きます。

```
for i in range(3):
    print("Python って楽しい！")
```

このコードは、「Python って楽しい！」というメッセージを 3 回表示してくれます。

range(3) という部分が、“0 から 2 までの 3 回”を意味しています。

繰り返しの中で何をしたいかは、インデント（スペースを空ける）で示すのがポイントです。

プログラムを見ただけで、「あ、これは 3 回繰り返すのだな」とわかるようになります。

●応用として、こんな風にもできます。

```
for i in range(5):
    print(f"{name}さん、今日もがんばってください！")
```

これで、さくらさんに 5 回メッセージを送れます。

ここまで、「Google Colab を使って Python を実行する」という体験をしていただきました。

内容をおさらいしましょう。

Google Colab は、ブラウザで使える開発環境です、はじめての人でも簡単に始められます。

print 関数で、画面に文字を表示できました。

変数を使えば、情報を記憶して、呼び出すことができました。

条件分岐で、「もし～だったら…」という判断ができました。

ループで、同じ動作を繰り返す指示もできました。

すべてのコードは、とても短くてシンプルなものでしたが、しっかりと意味を持って動いてくれました。はじめての方でも、「自分で書いたコードが動く」体験は、とても大きな一歩です。

この感覚を大切にしながら、次は「AI アシスタントを活用して文章を作ったり、ウェブを自動化したり」という、もっと実用的な世界にも入っていきます。

Python は、知識というより“道具”です。その使い方に触れたことで、みなさんの選択肢は、ぐんと広がりました。

※これで実習は終了です、実習した内容はスタッフさんに確認してもらいましょう

3. ChatGPT でコードを改善してみよう

次は、AI アシスタントである ChatGPT を使って、コードの改善や修正に挑戦してみましょう。「えっ？自分で書いたコードを AI が直してくれるの？」と、不思議に思った方もいらっしゃるかもしれません。

ChatGPT は、コードのエラーを見つけたり、より良い書き方を提案してくれたりする、非常に優れたツールです。

そもそも ChatGPT とは、OpenAI という企業が開発した AI チャットアシスタントです。人間のような自然な会話ができるのが特徴で、「こんにちは」と話しかけると、「こんにちは」と返してくれる、そんな親しみやすい存在です。日本語にも対応していて、英語が苦手な方でも安心して利用できます。

この AI は、ただの検索ツールではありません。「どう書けばいいのかわからない」「どこが間違っているのか教えてほしい」といった、あいまいな質問にも柔軟に対応してくれます。

たとえば、Python で次のようなコードを書いたとします。

```
name = input(″名前を入力してください″)
print(″こんにちは″ + name)
print(″今日も頑張りましょう！″
```

一見、普通のコードに見えますが、最後のカッコが閉じられていないという、よくあるエラーが含まれています。初心者の方は特に、このような小さな記号のミスに気づきにくいものです。実行すると「SyntaxError」などのエラーメッセージが表示されてしまい、「何がいけないのかわからない」と焦ってしまうかもしれません。

ChatGPT に「この Python コードにエラーがあります。修正してください。」と指示すると、次のように正しいコードを教えてくれるのです。

```
name = input(″名前を入力してください″)
print(″こんにちは″ + name)
print(″今日も頑張りましょう！″)
```

さらに、「このように、最後のカッコが抜けていたのが原因です」といった説明も添えてくれることがあります。まるで、優しい先生が隣でそっと教えてくれるような安心感があります。

また、ChatGPT はただエラーを直すだけではありません。表現をもっと丁寧にしたいときにも力を貸してくれます。たとえば、「もっと丁寧な言い回しにしてほしい」と指示してみましょう。

```
name = input("お名前をご入力ください：")
print(f"{name}さん、こんにちは。本日もどうぞよろしくお願いいたします。")
```

このように、実際のビジネスや接客の現場で使えるような丁寧な文章に書き換えてくれます。チャットボットや接客システムなどにも応用できる、実践的なスキルが自然と身につきます。

さらに、ChatGPT に「このメッセージを 5 回繰り返して」と伝えると、次のようなコードを提案してくれます。

```
for i in range(5):
    print(f"{name}さん、こんにちは。本日もどうぞよろしくお願いいたします。")
```

これは「for 文」と呼ばれる、繰り返し処理の基本的な構文です。ChatGPT は、こうした処理の考え方も教えてくれます。

また、ChatGPT は日本語だけでなく、英語にも対応しています。たとえば、「What does this code mean?（このコードはどういう意味？）」と聞けば、英語で丁寧に解説してくれます。英語の学習ツールとしても活用できるので、将来的にグローバルに活躍したい方にも、とても頼りになる存在です。

コードのエラーにもいろいろな種類があります。

「SyntaxError（構文エラー）」や

「TypeError（型の間違い）」など、

最初は意味がわからず戸惑ってしまうこともあるでしょう。ChatGPT は、そうした専門用語の意味もやさしく説明してくれます。

「SyntaxError は、コードの文法が正しくないときに出るエラーです」

といった形で、初心者にもわかりやすい表現で教えてくれるのです。

さらに、「もっと簡単な言葉で説明してください」とお願いすれば、表現をかみ砕いて説明しなおしてくれます。この柔軟さは、人間の先生にも負けない魅力のひとつです。

ChatGPT は、コードの文法だけでなく、「なぜそのように書くのか」という背景まで教えてくれます。たとえば、「なぜここで if 文を使うの？」と聞くと、「特定の条件のときだけ処理を実行するためです」と、具体例を交えて説明してくれます。このような会話を重ねることで、自然とプログラミングの考え方が身についていくのです。

「人に聞くのはちょっと勇気がいるな…」という方にも、AI である ChatGPT は心強い味方です。AI なら、どんな質問でもバカにせず、何度聞いても嫌な顔をしません。人に聞くときの緊張感がない分、気軽に質問できるのも大きなメリットです。

便利な AI にも注意点があります。ChatGPT は非常に高性能ですが、すべてを完璧にこなせるわけではありません。あいまいな質問には間違った回答をすることもありますし、複雑な問題には誤ったコードを提案することもあります。

そのため、ChatGPT の回答をうのみにせず、

「この内容は正しいかな？」

「別の方法もあるかな？」と、

自分で考える姿勢がとても大切になります。AI はあくまで補助役であり、学びの中心はあくまで自分自身です。

また、ChatGPT が生成したコードをそのまま提出したり、他人のコードとして扱ったりするのは、学習上も倫理的にも問題があります。

大切なのは、「理解して、自分の言葉で書き直すこと」です。

AI が示してくれた内容を参考にしながら、自分なりにアレンジすることで、より深い理解につながります。

ChatGPT の使い方は「コードの修正」にとどまりません。

「簡単なゲームを作ってみたい」と相談すれば、じゃんけんゲームや数字当てゲームのコード例を出してくれることもあります。「この機能を追加したい」と伝えれば、改良の提案もしてくれます。

加えて、ChatGPT を使ってコードのコメントを自動生成したり、コードの処理内容を日本語で要約したりといった応用も可能です。こうした活用法を少しずつ取り入れていくことで、実務にも役立つスキルへとつながっていきます。

たとえば、Excel での VBA マクロ、HTML での Web ページ作成、SQL でのデータベース操作なども、ChatGPT はある程度対応できます。これにより、「AI に相談しながら幅広く学ぶ」とい

う新しい学習スタイルが実現されているのです。

教育現場や職場でも、ChatGPT は徐々に取り入れられています。

学校では、生徒の個別対応が難しい場面で補助的な先生のように使われたり、企業では、新人研修や資料作成、簡単な自動化処理の学習支援にも役立っています。

これからの時代、AI とどう向き合うかが大切なテーマになります。

AI と競うのではなく、AI を味方につけて学びや仕事のパートナーとして活用していく。

そんな姿勢が、これからの私たちに求められるスタイルです。

ChatGPT はプログラミング初学者にとって、非常に心強い味方です。

コードのエラーを見つけてくれたり、丁寧な表現に変えてくれたり、繰り返し処理を提案してくれたりと、さまざまな場面で助けてくれます。しかし、それ以上に大切なのは、それを活かす「学ぶ側の姿勢」です。

わからないことをそのままにせず、自分の言葉で理解しようと努力すること。ChatGPT を単なる便利な道具としてではなく、「学びを支えてくれる相棒」として捉え、上手に付き合っていくこと。その姿勢こそが、これからの時代における「AI リテラシー」の第一歩です。

AI と一緒に学ぶことは、決して難しいことではありません。むしろ、自分のペースで、無理なく、そして楽しく学びを深めることができる、新しい学びの形です。

どうか皆さんも、ChatGPT とともに、安心して一歩ずつプログラミングの世界を楽しんでください。きっとそこには、新しい気づきや、ワクワクする発見がたくさん待っているはずです。

4.まとめ

Python というプログラミング言語を使って、実際にご自身の手でコードを書いてみるという体験をしました。初めての方も多かったと思いますが、

「print 関数」を使ってみる。変数や条件分岐を試してみる。

そうした一つ一つのアクションが、プログラミングの世界への大切な一歩です。

その際に活用した Google Colab は、わざわざソフトを入れる必要もなく、ブラウザを開くだけで Python のコードが書ける。これは、これから学習を始めたい方にとって、非常に大きな味方になるはずです。

もう一つの強力なツール「ChatGPT」も使いました。エラーが出てしまったときや、「もっと丁寧な表現にしたいな」「この部分、効率よく書けないかな」と思ったとき。そんなときに ChatGPT に質問すると、驚くほどスムーズに答えが返ってきました。「コードのどこを直せばよいか」「どうすれば目的の動きに近づけるか」、そうした相談を、気軽にできる存在があるというのは、とても心強いことです。

AI を活用すれば、プログラミングのハードルは一気に下がります。

そして、ただコードを書くという行為だけでなく、そのプロセスの中で「自分で考えて、調べて、修正する」力も身についていきます。

今日の体験を通じて、「自分でもやってみたい」「もっと知りたい」と思えたなら、それは大きな一歩です。小さな成功体験が重なることで、学びはどんどん楽しくなっていきます。

第 3 章 AI アシスタントの活用

1.はじめに

私たちの毎日の生活の中にある「AI（人工知能）」の活用例について、できるだけわかりやすくお伝えしていきます。

AI と聞くと、なんだか難しそうに感じる方も多いかもしれません。ですが実は、皆さんもすでに気づかないうちに AI を使っています。

たとえば、スマホで検索したり、音声アシスタントに話しかけたり、外国語の文章を翻訳したり——それらの裏側では AI がしっかりと働いてくれています。

2. ChatGPT で文章作成・校正を体験しよう

AI アシスタントの代表格ともいえる「ChatGPT」を使って、文書を作る、そしてそれを校正・改善するという 2 つの体験をしていただきます。

「文書作成」と聞くと、何から書き始めればいいのか悩んだり、時間ばかりかかってしまう…そんな経験、ありませんか？そんなときこそ、AI の力を借りてみましょう。

では、「文章を書く」体験を行います。

たとえば、地域のイベントを企画していて、参加者を集めたいとします。

でも、いざ案内文を書くとなると、「どういう文体で？」「長さはどれくらい？」など、迷ってしまうことも多いです。そんなときは、ChatGPT にこう頼んでみましょう。

「地域イベントのお知らせ文を、参加を促すように丁寧に書いてください」

この一文を入力するだけで、ほんの数秒後に提案文が現れます。

たとえば、こんな感じです。

> 「皆さま、こんにちは。
>
> 来たる〇月〇日、地域のふれあいイベントを開催いたします。
>
> ご家族やご友人とぜひお越しください。詳細は以下をご覧ください。」

挨拶から始まり、日時の案内、参加の呼びかけまでが自然につながっていて、とてもわかりやすい構成です。

言葉も丁寧で、相手に対する気配りも感じられる。これなら、地域の広報や学校の通信、あるいは自治体のお知らせにも使えます。

ChatGPT は文章の「調整」や「トーンの変更」にも柔軟に対応してます。

たとえば…

「もっとカジュアルにしてください」

「高齢者向けにやさしい言葉にしてください」

「小学生にもわかるように書き直してください」

この様に伝えると、AI は瞬時に文章を再構成し、より適切な表現に“翻訳”してくれるのです。

次は「文章の校正」です。

自分で書いた文が、「ちょっと幼い感じがする」とか「敬語に自信がない」…そんなこと、ありませんか？

たとえば、こんな文章を ChatGPT に渡してみます。

「このまえのかいぎで、ちょっとしたもんだいがでました。」

これを AI にチェックしてもらうと、次のように修正提案が返ってきます。

「先日の会議で、いくつかの課題が浮かび上がりました。」

あるいは、

「前回の会議では、少し問題が発生しました。」

ただの言い換えではありません。

文法的な正しさはもちろん、相手に失礼がなく、わかりやすく、そして少し改まった印象を与えるように整えてくれます。

このように、ChatGPT は言い回しの調整や敬語表現の変換、語彙のリライトにも優れています。

まだ「使い方が難しそう…」と感じた方もいらっしゃるかもしれません。

ChatGPT の使い方はとてもシンプルで簡単です。

コツはたった 3 つ。

- ●「どんな文章を書きたいのか」を伝える
- ●「どんな印象にしたいのか」を補足する
- ●「例文やキーワード」があれば加える

この 3 つのどれか、または全部を入力するだけで、AI が自動的にベストな提案を返してくれます。そして、ここが重要なのですが、“一度で終わり”じゃないというのが最大の魅力です。

たとえば…

「もっと短くしてください」

「漢字を減らしてください」

「箇条書きにしてください」

こうした細かい修正指示を、あとから何度でも出せるのです。しかもそのたびに、AI が最適な文に“アップデート”してくれます。

では、実践的な「応用編」に進みます。

※実際に ChatGPT を使って見てください

ChatGPT に、以下の指示をしてみます。

「高校生向けのキャリアイベントを案内する文章を、

ワクワクするように書いてください」

これだけで、こんな返答が得られます。

「進路に迷っている高校生の皆さんへ！

未来への一歩を踏み出すチャンスです。先輩たちのリアルな話が聞けるキャリアイベントを開催します。

夢を形にするヒントを、ぜひここで見つけてください！」

どうですか？ 文章から“ワクワク感”が伝わってきませんか？

ChatGPT は、対象読者やシチュエーションに応じて文章のトーンを自在に切り替えられる、そんな柔軟性を持っているのです。

ChatGPT の活用は案内文や文章作成にとどまりません。

たとえば…

●SNS 投稿の文案づくり

●履歴書や職務経歴書の下書き

●説明会の原稿やプレゼン資料の文章整理

こういった用途にも、ChatGPT は強力なアシスタントになってくれます。

次に「校正者」としての機能です。

自分が書いた文章をコピーして、ただ「読みやすくしてください」と指示するだけです。

それだけで、文の構成や語尾の調整、敬語の使い方まで整えてくれます。

さらに、「読みやすさを 5 段階で評価してください」といった、客観的なフィードバックも得られるのです。

ChatGPT を「使う」のではなく、ChatGPT と「一緒に書く」という意識が重要です。

これこそが、これからの時代に求められる AI リテラシーです。

あなたが考えたアイデアを、AI が整理し、構成し、伝わる言葉にしてくれる。

つまり、ChatGPT は“あなたのもうひとつの頭脳”として動いてくれるのです。

この続きでは、さらに AI を活用して、表計算ソフトやデータ分析へと進んでいきます。

あなたの仕事や学びがもっとスムーズに、もっと楽しくなる未来のために。

このスキル、ぜひ身につけましょう！

最後に、ちょっと視点を変えて考えてみましょう。「ChatGPT を使う」という表現、よく耳にしますが、実はもっとしっくりくる言い方があります。

それは…

「ChatGPT と一緒に書く」という感覚。

文章作成という作業は、ときにひとりで抱え込んでしまいがちです。

「うまく言葉が出てこない」

「考えがまとまらない」

「伝えたいけど、どう表現すればいいか分からない」

そんなときこそ、AI は頼れる“もうひとりの相棒”になります。

あなたの頭の中にある“もやもやしたイメージ”。

その断片的な言葉や想いを、ChatGPT がひとつの文章へと編み直してくれる。

例えるなら、それはまるで――

あなたの考えを外に取り出して、「見えるカタチ」にしてくれる“外部の脳”のような存在。

思考の整理、言葉の肉付け、表現のリズムやトーンまで、AI はあなたの“伝えたい気持ち”に寄り添いながら、丁寧にサポートしてくれます。

最初から完璧な答えが返ってくるわけではありません。大切なのは、対話を続けること。

AI に問いかけ、修正を重ね、調整しながら、自分の意図を少しずつ明確にしていく。そのプロセスこそが、

「表現する力」＝「伝える力」

を養う道筋になるのです。

そしてこれは、文章を書くことが苦手だと感じている人にこそ、ぜひ伝えたいポイントです。

うまく書けなくてもいい。まずは、AI に相談してみる。あなたの中にある思いや目的を、言葉にする第一歩として、ChatGPT はきっと力になってくれます。

文章とは、ただの情報ではありません。

それは、“あなたの考え”や“あなたの想い”が込められた、ひとつのコミュニケーションです。

AI と対話を重ねることで、そのコミュニケーションの質は、必ず磨かれていきます。

つまり、ChatGPT は単なるツールではなく、

「一緒に考え、一緒に伝えるパートナー」とも言える存在です。

●あなたの声を、より明確に。

●あなたの言葉を、より届くかたちに。

その先にあるのは、“伝える”という行為の、新しい可能性です。

さあ、今日からはひとりで文章と向き合う必要はありません。AI とともに、あなたの表現を育てていきましょう。

3. Excel Copilot でデータ分析

今回は、AI が Excel の操作を支援してくれる新しいアシスタント「Copilot」を活用して、データ分析をより簡単に、より深く行う方法をご紹介します。

●「Excel はなんとなく使っているけど、複雑な分析までは無理…」

●「関数やグラフの使い方がよく分からない」

●「資料作りのたびに時間ばかりかかってしまう」

そんな悩みをお持ちの方にこそ、今回の内容は大きな助けになるでしょう。

◆Copilot

「Copilot（コパイロット）」は、Microsoft 365 に搭載された AI 機能で、あなたが自然な言葉で

「こうしてほしい」と伝えるだけで、複雑な表計算や分析、グラフ作成を行ってくれます。

まずは、実際の活用シーンをイメージしてみましょう。

あなたのパソコンには、ある売上データの表が表示されています。

たとえば、月別の売上金額、商品ごとの販売数、地域別の売上構成、前年との比較表です。

これまでなら、このデータを分析するには関数を使ったり、表をフィルターで並び替えたり、いろいろと手作業が必要でした。しかし、Copilot があれば、その手間が大きく変わります。

たとえば、こう指示してみてください。

「売上が前年より伸びている地域を一覧にして」

「この中で一番売れている商品を教えて」

「月ごとの売上推移をグラフで表示して」

すると Copilot は、Excel 上でその指示をすぐに理解し、該当データを抽出・計算・可視化してくれます。しかも、関数の知識も不要、グラフの種類を手で選ぶ必要もありません。

ただ「指示だけ」で完了する。まさに未来型の使い方です。

Copilot の最大の特徴は、この“自然言語でのやりとり”です。

つまり、関数名や専門用語を知らなくても、

「～を、してください」「これを～に変えて」とお願いするだけで処理してくれるのです。

たとえば、次のような依頼もできます。

「前年比を追加して、伸び率を示してください」

「このデータからパイチャートを作成して」

「今月と先月の売上差を自動計算して表示して」

Excel 初心者にとって、この自由度の高さは非常にありがたいですよね。

そして上級者にとっても、「時間短縮」や「作業の正確性向上」というメリットがあります。

さらに、Copilot はデータの要点を“まとめて説明”してくれる機能も持っています。

たとえば次のように頼むと…

「この表の内容を一文で説明してください」

「上昇傾向にあるカテゴリーを教えて」

「この表の注目点を 3 つに絞ってまとめて」

まるでレポート作成アシスタントのように、要点を抽出して整理してくれるのです。

これは、報告資料・プレゼン・社内共有など、実務で非常に役立つポイントです。

ここで、Copilot が“グラフ作成”に強いことにも注目しましょう。

従来の Excel では、棒グラフや折れ線グラフを作るために、「挿入」タブを開いて、対象セルを

選んで…といった細かい手順が必要でした。

でも Copilot なら…

「前年と今年の売上を比較できる棒グラフを作成して」

「トップ 5 の商品を目立つ色で表示して」

「月別の売上推移を折れ線グラフで表現して」

このように指示すれば、Copilot が適切な種類のグラフを選んで、自動で作成・整形してくれるのです。しかも、配色、凡例の設定、タイトルの追加まで一括で完了。これは資料作成やプレゼン準備の時間を、格段に短縮してくれます。

たとえば、次のような“改善提案”も行ってくれます。

「このセルは空白になっていますが、どう扱えばいいですか？」

「前月と比べて急に数値が変動しています。理由を調べますか？」

「この列は不要かもしれません。非表示にしましょうか？」

このように、ただ受け身で指示を待つだけではなく、“気づき”をもたらす AI アシスタントとしての役割も果たしてくれるのです。

また、Excel 上の Copilot は、資料を“読みやすく整える”ための支援もしてくれます。

「表の見出しを中央揃えにして」

「背景色を交互に変えて読みやすくして」

「強調すべき数値に太字を使って」

このようなレイアウトの微調整まで、自然な指示で実行できます。

そして、実は Copilot は複数の処理を“まとめて”頼むことも可能です。

「このデータを要約して、前年比を追加して、棒グラフを作って、色も強調してください」

なんとこれだけで、分析・要約・可視化・装飾まで一気に処理されるのです。

これはもう、Excel を超えて“補助的な AI 秘書”と呼べるかもしれません。

では、ここで気になる点を一つお伝えします。

現在この Copilot 機能は、Microsoft 365 の特定のプラン（主に法人向け・教育機関向け）で提供されており、誰でもすぐに使えるとは限りません。

ですが、こうした AI アシスタント機能は急速に普及しており、今後は多くの PC に標準搭載されていく流れが予想されます。

また、Google スプレッドシートの「Duet AI」、Notion の AI アシスト機能など、同様の AI 支援型ツールが各社から次々と登場しています。

つまり、Excel に限らず、「話しかけるだけで操作できる」という時代が、すぐそこまで来ているのです。

これまで、「Excel が苦手」と感じていた方。「分析は専門の人がやるもの」と思っていた方。

そんな方々こそ、Copilot との対話を通じて、“自分の力で分析できる”体験を味わってほしいと思います。

最後に、ひとつだけお伝えしておきたいことがあります。それは、「完璧に使いこなそうとしないでください」ということ。Copilot は、試しに話しかけるところから始めるだけで OK です。

「これ、できる？」と聞いてみる。「この資料、整えて」と頼んでみる。

そこから、あなたと AI の“共同作業”が始まります。

Excel に詳しくなくても構いません。むしろ、Copilot と対話することで“理解が深まっていく”という循環が生まれるのです。あなたの中にある「疑問」や「困りごと」が、AI を通じて形になり、“問題解決の手がかり”となって戻ってくる。

それが、AI と仕事をするということです。

4. まとめ

実際に Python のコードを書いて実行する体験を通じて、

「プログラミングって、案外できるかも！」という感覚を持っていただけたのではないかと思います。

Google Colab というツールを使えば、インストールも不要で、すぐに始められる。

そして、ChatGPT のような AI アシスタントを活用すれば、コードの修正や改善も手助けしてもらえる。

「分からないことがあっても大丈夫。AI に聞けばいい。」

そんな頼もしい存在が、いま私たちのすぐそばにあるのです。

コードを書くことは、コンピュータに“やってほしいこと”を伝える言葉を学ぶこと。

だからこそ、ほんの少しでも「自分の力で動かせた」という実感が得られたなら、それは大きな一歩です。

第 4 章 Q&A

Q1：プログラムとは何ですか？

A1：プログラムとは、コンピュータに対して「何をどうしてほしいか」を明確に伝える命令の集合です。

コンピュータは人間のようにあいまいな指示を理解できないため、すべての処理を正確かつ順序立てて指定する必要があります。たとえば、自動販売機における「お金を受け取る」「商品を出す」といった動作も、プログラムによって制御されています。

Q2：プログラミング的思考とは何ですか？

A2：プログラミング的思考とは、物事を論理的に整理し、順序立てて考える力を指します。

この思考力を身につけることで、自分の考えをわかりやすく伝えることができ、複雑な問題を段階的に解決できるようになります。AI に対して指示を出す際にも、プログラミング的思考があると的確な伝達が可能となり、より良い結果が得られます。

Q3：フローチャートはどのような場面で使いますか？

A3：フローチャートは、処理手順や条件分岐を図で視覚的に表現するために使用されます。

たとえば、「もし雨なら傘を持つ、晴れなら帽子をかぶる」といった判断を、図形と矢印を使って整理することで、処理の流れが一目で理解しやすくなります。

Q4：プログラムの基本構造にはどのような種類がありますか？

A4：プログラムの基本構造は、大きく 3 つに分類されます。

1 つ目は「順次処理」で、命令を上から順に実行します。

2 つ目は「分岐処理」で、条件によって処理の内容を切り替えます。

3 つ目は「反復処理」で、同じ処理を繰り返し実行します。

これら 3 つを組み合わせることで、さまざまなアプリケーションを構築することが可能となります。

Q5：Python はなぜ初心者向けと言われているのですか？

A5：Python は、文法がシンプルで直感的な構文を採用しているため、初学者でも理解しやすいという特徴があります。

英語に近い形で記述できるうえ、他のプログラミング言語と比べて必要なコードの量が少なく済むことから、効率的に学習できます。教育分野や AI 開発など、幅広い領域で活用されています。

Q6：変数とは何ですか？日常にたとえると？

A6：変数とは、値を一時的に保存しておく「名前付きの箱」のようなものです。

たとえば「財布」を例にすると、財布＝変数、中のお金＝値と考えることができます。

Python では『name = ″さくら″』のように、変数名と値をセットで記述し、プログラム内で繰り返し利用します。

Q7：Python で条件分岐を行うにはどうすればよいですか？

A7：Python では「if 文」を使って条件分岐を行います。

たとえば、『if age >= 20:』と記述すると、「age が 20 以上の場合」に限定した処理を実行することができます。

条件に合わない場合には「else」や「elif」を使い、別の動作に切り替えることが可能です。

Q8：Python での繰り返し処理にはどのような種類がありますか？

A8：Python には主に「for 文」と「while 文」の 2 種類のループがあります。

「for 文」は決まった回数の繰り返し処理に適しており、たとえば『for i in range(5):』と書くことで、5 回繰り返す処理が実行されます。

一方、「while 文」は、特定の条件が成立している間、繰り返し続ける処理に適しています。

Q9：Google Colab とは何ですか？

A9：Google Colab は、ブラウザ上で Python のコードを無料で実行できるクラウド型のサービスです。

専用ソフトのインストールが不要で、パソコンとインターネット環境があればすぐに利用できます。実行結果をその場で確認できるため、初心者にも使いやすい学習環境として広く活用されています。

Q10：Python で文字を表示するにはどうすればよいですか？

A10：Python では『print()』関数を使用することで、画面上に文字を表示することができます。

たとえば『print("こんにちは")』と記述すれば、プログラムを実行した際に「こんにちは」と表示されます。これはプログラミングにおける最初の基本操作の一つです。

Q11：AI にコードの修正をお願いすることはできますか？

A11：はい、可能です。たとえば ChatGPT のような AI にエラーのあるコードを提示すると、問題の箇所を指摘し、修正案を提示してくれます。

プログラミングの初学者にとって、安心して試せるサポートツールとなっています。

Q12：AI にわかりやすく言い換えてもらうことはできますか？

A12：できます。ChatGPT などの AI は、文章の文体やトーンを調整するのが得意です。

たとえば、「ビジネス風に」「子ども向けに」といった要望に応じて、適切に言い換えてくれます。「もっとやわらかく」「カジュアルに」といった指示も有効です。

Q13：AI で学ぶことに意味はありますか？

A13：はい、大きな意味があります。AI は 24 時間いつでも質問に応えてくれ、同じことを何度聞いても問題ありません。

初心者がつまずきやすいポイントについても丁寧に解説してくれるため、自分のペースで学習を進めることが可能です。

Q14：AI に頼りすぎると自分で考えられなくなるのでは？

A14：その懸念は理解できますが、AI を「先生」ではなく「相談相手」として活用することが大切です。

AI の回答を参考にしながら、自分の言葉で理解・表現することで、思考力を高めることができます。

Q15：プログラミングは将来どのように役立ちますか？

A15：プログラミングは、論理的思考力や問題解決力を養ううえで非常に有効です。

また、AI や IT ツールを活用するための基礎力としても重要であり、年齢や職業を問わず、将来的に大きな価値を持つスキルです。