# Windows AI開発環境構築ガイド（入門編）

AI開発環境（入門編）の一括インストール・クイックスタートガイド

本文のクイックスタートガイドです。本文でのインストールを素早く開始したい場合のために、コマンド部分のみを抜き出して集めたものです。 詳細な説明、各ツールの概要、トラブルシューティング等については、本文をご参照ください

目的

コマンドでAI開発ツールを一括インストールするスキルを習得し、開発環境の基盤を構築する。AI開発では多数のツールとライブラリが必要であり、パッケージマネージャーによる一括管理は必須技能である。

実行前の確認事項

\* ディスク空き容量が20GB以上あることを確認

\* すでにPythonがインストール済みの場合は、基本的にそのまま使用する（これまでの各自の研究・学習環境を維持するため）

手順

1. 管理者権限でコマンドプロンプトを起動

Windowsキーまたはスタートメニュー、「cmd」と入力、「管理者として実行」を選択

2. 以下のコマンドを1つずつ実行

wingetコマンドは一つずつ実行し、完了を待つ（実行前に内容を理解しておくことが望ましい）。

reg add "HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Control\FileSystem" /v LongPathsEnabled /t REG\_DWORD /d 1 /f

reg query "HKLM\SYSTEM\CurrentControlSet\Control\FileSystem" /v LongPathsEnabled

winget install --scope machine --id Nvidia.CUDA --version 12.6 -e

winget install --scope machine --id Python.Python.3.12 -e --silent

winget install --scope machine --id Python.Launcher -e --silent

winget install --scope machine --id Git.Git -e --silent

winget install --scope machine --id 7zip.7zip -e --silent

winget install --scope machine --id Microsoft.VisualStudioCode -e --silent

winget install --scope machine --id Codeium.Windsurf -e --silent

3. 新しくコマンドプロンプトを管理者として実行

Windowsキーまたはスタートメニュー、「cmd」と入力、「管理者として実行」を選択

4. 次のコマンドを実行

まとめて実行しても良い（次の実行において、特に結果は表示されない。エラーメッセージが出ない場合は実行に成功している）

set "INSTALL\_PATH=C:\Program Files\Python312"

echo ;%PATH%; | find /i ";%INSTALL\_PATH%;" >nul

if errorlevel 1 setx PATH "%PATH%;%INSTALL\_PATH%" /M >nul

echo ;%PATH%; | find /i ";%INSTALL\_PATH%\Scripts;" >nul

if errorlevel 1 setx PATH "%PATH%;%INSTALL\_PATH%\Scripts" /M >nul

set "CUDA\_PATH=C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.6"

if exist "%CUDA\_PATH%" setx CUDA\_PATH "%CUDA\_PATH%" /M >nul

if exist "%CUDA\_PATH%" setx CUDNN\_PATH "%CUDA\_PATH%" /M >nul

set "NEW\_PATH=C:\Program Files\Git\cmd"

if exist "%NEW\_PATH%" echo %PATH% | find /i "%NEW\_PATH%" >nul

if exist "%NEW\_PATH%" if errorlevel 1 setx PATH "%PATH%;%NEW\_PATH%" /M >nul

set "NEW\_PATH=C:\Program Files\7-Zip"

if exist "%NEW\_PATH%" echo %PATH% | find /i "%NEW\_PATH%" >nul

if exist "%NEW\_PATH%" if errorlevel 1 setx PATH "%PATH%;%NEW\_PATH%" /M >nul

set "NEW\_PATH=C:\Program Files\Microsoft VS Code"

if exist "%NEW\_PATH%" echo %PATH% | find /i "%NEW\_PATH%" >nul

if exist "%NEW\_PATH%" if errorlevel 1 setx PATH "%PATH%;%NEW\_PATH%" /M >nul

set "NEW\_PATH=C:\Program Files\Windsurf"

if exist "%NEW\_PATH%" echo %PATH% | find /i "%NEW\_PATH%" >nul

if exist "%NEW\_PATH%" if errorlevel 1 setx PATH "%PATH%;%NEW\_PATH%" /M >nul

if exist "C:\Program Files\Microsoft VS Code\bin" cd "C:\Program Files\Microsoft VS Code\bin"

if exist "C:\Program Files\Microsoft VS Code\bin" code --install-extension ms-python.python

if exist "C:\Program Files\Microsoft VS Code\bin" code --install-extension ms-python.vscode-pylance

if exist "C:\Program Files\Microsoft VS Code\bin" code --install-extension MS-CEINTL.vscode-language-pack-ja

if exist "C:\Program Files\Microsoft VS Code\bin" code --install-extension dongli.python-preview

if exist "C:\Program Files\Windsurf\bin" windsurf --install-extension MS-CEINTL.vscode-language-pack-ja

実行時のヒント

* wingetコマンドは一つずつ実行し、完了を待つ
* コマンドの途中で止まったら、まずEnterキーを1回押す。そして、エラー内容をよく読む
* Pythonのインストールのときに「Modify Repair Uninstall」画面が出た場合は、Pythonがインストール済みなので「Cancel」をクリック

よくある間違いの回避

* コマンドの途中で改行しない（1つのコマンドは1行で実行）
* 管理者権限で実行していることを確認

## 1. はじめに：AI開発環境の全体像

このガイドは，**Windows PC上でAI開発を始めたいと考えている学生や技術者**を対象としています．本ガイドでは，AI開発で広く使われるプログラミング言語Python，基本的なデータ処理ライブラリ，そして開発を助けるコードエディタを中心に，**シンプルかつ実践的なAI開発環境**を構築することを目指します．このガイドの手順に沿って環境を構築すれば，簡単なデータ分析や機械学習モデルの実行，ディープラーニングの入門的なコードを試すことができるようになります．

対象とするPC環境はWindows 11です．NVIDIA製GPUがある場合はGPUを活用した開発環境も構築できますが，GPUがない場合でもCPUのみで多くの学習・開発を行うことは可能です．AI開発はCPUだけでも可能ですが，深層学習ではGPUを使用することで計算速度が向上します．

インストールの実行には，PCのSSD／ディスクの空き容量が約20GB必要です．事前にご確認ください．空き容量が無い人は実施しないこと．

このガイドが，Windows AI開発環境構築とその後の学習・開発活動に役立つことを期待しています．

## 2. ガイドが対象とするAI開発環境（入門編）

本ガイドでは，AI開発環境（入門編）として，以下のツール群の役割，メリット，そしてインストール手順を解説します．9章には，より高度なステップについても説明しています．

* **プログラミング言語:** Python
* **バージョン管理:** Git
* **GPU利用基盤:** NVIDIA CUDA
* **Pythonライブラリ:** PyTorch, NumPy, Pandas, Scikit-learn, Matplotlib, Seabornなど
* **開発エディタ:** Visual Studio Code, Jupyter Notebook, Windsurf
* **ユーティリティ:** 7-Zip

+-----------------------------------------------------------------+
| 開発エディタ (VS Code / Jupyter Notebook / Windsurf) |
| +--------------------------------------------------------------+ |
| | Git (バージョン管理) | |
| +--------------------------------------------------------------+ |
| |
| +--------------------------------------------------------------+ |
| | Python + AIコアライブラリ (PyTorch, NumPy, Pandasなど) | |
| +--------------------------------------------------------------+ |
| |
+-----------------------------------------------------------------+
| 基盤 (OSとハードウェアの橋渡し) |
| +---------------------------+ <--連携--> +--------------------+ |
| | PyTorch内のCUDAライブラリ | | NVIDIA CUDA | |
| +---------------------------+ +--------------------+ |
+-----------------------------------------------------------------+
| OS・ハードウェア (Windows, NVIDIA GPU) |
+-----------------------------------------------------------------+

## 3. 導入にあたっての推奨事項 (段階的アプローチ)

スムーズな環境構築のため，以下の段階的な手順を推奨します．

1. **環境構築の基礎知識を学ぶ:** まず，コマンドプロンプトの基本的な使い方など，Windowsでの環境構築に必要な基礎知識を確認します（4章参照）．
2. **AI開発環境（入門編）のインストール:** Python本体，Git，7-Zip，開発エディタ必要であればNVIDIA CUDAをPCにインストールします（5.1節参照）．
3. **Pythonライブラリの導入:** PyTorchなどのAI開発等に必要なPythonライブラリをインストールします（5.2節参照）．
4. **開発エディタ:** 3つの開発エディタ（Visual Studio Code, Jupyter Notebook, Windsurf）を知り，状況に応じて活用します（5.3節参照）．

## 4. 環境構築の前に知っておきたいWindowsの基礎

この章では，環境構築の手順を進める上で必要となるWindowsの基本的な操作や概念について解説します．ここで学ぶ知識は，続くインストール手順で頻繁に登場するため，内容を理解しておくとスムーズに進められます．

### 4.1. フォルダ（ディレクトリ），パス

**フォルダ**（ディレクトリ：ファイルを整理するための仕組み）は，複数のファイルやフォルダを格納します．そのことで，大量のファイルを整理して格納できます．

**パス**は，コンピュータ上でファイルやフォルダの位置を示す文字列です．例えば，「C:\Users\金子さん\Documents\研究\データ.xlsx」は，Cドライブ（コンピュータ内の記憶装置）にUsersフォルダがあり，その中に金子さんフォルダがあり，更に，その中にDocumentsフォルダがあり，その中に「データ.xlsx」があります．このファイル「データ.xlsx」のパスは「C:\Users\金子さん\Documents\研究\データ.xlsx」になります．

### 4.2. 環境変数とユーザープロファイル

**環境変数**はシステム全体で共有される設定値を格納するものです．**ユーザープロファイル**は各ユーザーアカウントに割り当てられる専用ディレクトリ（C:\Users\ユーザー名）であり，個人データと設定が格納されます．

#### 4.2.1. 環境変数

環境変数は，システム全体で共有される設定値を保存する変数です．例えば，環境変数%USERPROFILE%には現在のユーザーのプロファイルディレクトリ名（例：C:\Users\金子さん）が格納されています．コマンドプロンプトなどで「%USERPROFILE%\Desktop」と記述することで，そのユーザーのデスクトップフォルダを参照できます．

#### 4.2.2. ユーザープロファイル

ユーザープロファイルは，個人専用のフォルダ領域であり，Windowsの各ユーザーアカウント（コンピュータを使用する個人の登録情報）に対して個別に割り当てられる専用ディレクトリです．ユーザープロファイルにはユーザー固有のデータと設定が格納されます．

* Windows 11ではユーザープロファイルは，**C:\Users\ユーザー名**になります（ユーザー「金子さん」の場合は C:\Users\金子さん）．
* **コマンドプロンプト**（コマンドでシステムを操作するツール）を起動すると，作業ディレクトリ（コマンド実行時の基準となる場所）が，「C:\Users\金子さん>」のように表示されます．コマンドプロンプトの起動直後は，作業ディレクトリがユーザープロファイルに設定されます．コマンドプロンプトの作業ディレクトリは，**cdコマンド**で変更できます．

### 4.3. コマンドプロンプトの基本

**コマンドプロンプト**はコマンドでシステムを操作するツールです．システム管理，ファイル操作，設定変更，インストール作業などの様々な作業をコマンドで実行でき，GUI（グラフィカルインターフェース）では手順が多くなる操作や，自動化したい操作の実行に便利です．通常起動と管理者権限起動が可能で，cdコマンドで作業ディレクトリを変更できます．**ワイルドカード**（\*.tmp等）でファイルを一括指定し，dir，copy，del等のコマンドでファイル操作を実行できます．

#### 4.3.1. コマンドプロンプトとは

* **別の言葉で例えると:** 文字でPCと会話するための「画面」のようなものです．dir（ファイル一覧表示）やcd（フォルダ移動）といった「コマンド」を入力して，PCに直接命令を送ります．
* **コマンドプロンプトとは:** コマンドと呼ばれる文字列を入力してOSの機能を直接実行するための，キャラクタユーザインターフェース（CUI）です．GUI（Graphical User Interface）と異なり，キーボード入力が中心になります．
* **使用理由:** 複雑な作業を効率的に行ったり，繰り返しの作業を自動化したりするのに役立ちます．ソフトウェアのインストールや設定変更など，GUIでは対応が難しい操作もコマンドで行えます．

#### 4.3.2. 一般ユーザーと管理者

**一般ユーザー権限**では自分のプロファイル内操作が可能であり，**管理者権限**ではシステム全体を制御できます．このように，一般ユーザー権限であるか，管理者権限であるかによって，Windowsにおけるアクセス制御（誰が何を操作できるかを管理する仕組み）が行われます．つまり，実行できる操作が異なります．アクセス制御は，重要なシステムファイルが誤って変更されることを防ぐのに役立ちます．

* **一般ユーザー権限:** 日常的な作業に必要な基本的な操作権限です．自分のユーザープロファイル内でのファイル操作，インストール済みアプリケーションの実行などが可能です．
* **管理者権限:** システム全体を制御できる最高レベルの権限であり，システムの重要な設定やファイルを変更できる特別な権限です．一般ユーザー権限では C:\Windows\System32 などのシステムディレクトリへのファイル作成や，システム設定の変更ができませんが，管理者権限では可能です．重要なシステム操作には管理者権限が必要となります．

#### 4.3.3. コマンドプロンプトの起動

* **コマンドプロンプトの起動（通常起動）**

 　いくつかの方法があります．「cmd」は，コマンドプロンプトを起動するためのプログラム名です．

* + 方法① Windowsキーまたはスタートメニュー，cmdと入力，コマンドプロンプトを選ぶ，Enter
	+ 方法② Windowsキー + R（同時押し），cmdと入力，Enter

**実行例**．次により，環境変数 %USERPROFILE% の値が表示されます．

* + 1. コマンドプロンプトを起動（手順：Windowsキーまたはスタートメニュー，cmdと入力，コマンドプロンプトを選ぶ，Enter）
		2. 次のコマンドを実行．echo を用いて環境変数 USERPROFILEの値を表示．
		- echo %USERPROFILE%
* **コマンドプロンプトを管理者として実行**

システム設定やソフトウェアのインストールなど，システムレベルの変更が必要な操作をコマンドプロンプトで実行するときは，Windowsのコマンドプロンプトを管理者として実行します（手順：Windowsキーまたはスタートメニュー，「cmd」と入力，右クリックメニューなどで「管理者として実行」を選択）．コマンドプロンプトの通常実行では，Windowsのアクセス制御により，一部の操作が制限される場合があります．

**コマンドプロンプトの起動（管理者権限で起動）**

* 1. Windowsキーまたはスタートメニュー，cmdと入力
	2. 「コマンドプロンプト」を右クリックし，「管理者として実行」を選択．あるいはメニューに「管理者として実行」が表示されている場合は，それを選択
	3. セキュリティ警告：「このアプリがデバイスに変更を加えることを許可しますか？」というメッセージが表示されたら「はい」をクリックします．

#### 4.3.4. コマンド実行のパラメータ（オプション）

**パラメータ**や**オプション**（コマンドの動作を細かく制御するための追加指示）により，コマンドに条件や設定を追加できます．例えば「dir /a」コマンドにおいて，dir（dir：ディレクトリの内容を表示するコマンド）は「ディレクトリ内容の表示」という指示であり，/a（/a：すべてのファイルを「隠しファイルも含めて全て表示する」という追加指示）です．

#### 4.3.5. ワイルドカード

コマンド実行では，**ワイルドカード**と呼ばれる「何でも当てはまる」という意味を持つ特殊文字を用いて，複数のファイルを一度に指定できます．例えば，「\*.tmp」は「拡張子が.tmpであるすべてのファイル」を意味し，テスト.tmp，12345.tmp などのファイル名とマッチします．別の例としては「data\_\*.xlsx」は「data\_で始まるすべてのExcelファイル」を指定します．これは，ファイル名の一部について，任意の文字列にマッチする**パターンマッチング**（条件に合致するものを探す仕組み）を行うものです．

#### 4.3.6. コマンドプロンプトによるファイル操作

コマンド入力後，Enterキーで実行が開始されます．プロンプトには現在の作業ディレクトリ（現在操作対象となっているフォルダ）のパスが表示されます．コマンドプロンプトでは，現在作業しているフォルダを「**カレントディレクトリ**」と呼びます．

基本的なファイル操作コマンド

* **dir**: ディレクトリ（フォルダ）の内容を表示するコマンド．中のファイルやサブディレクトリ（サブフォルダ）などを一覧表示します．
	+ dir: 現在のディレクトリ内容を詳細表示します
	+ dir /a: 隠しファイルとシステムファイルを含む全ファイルを表示します
	+ dir /s: サブディレクトリを含む階層的な内容を表示します
	+ dir /w: ファイル名を横方向に整列表示します
* **cd**: カレントディレクトリを確認，移動するコマンド
	+ 使用例: cd Documents または cd ..（cd ..：一つ上の親ディレクトリへ移動するための特殊な指定）
	+ cd ディレクトリパス: カレントディレクトリを変更します．例：cd C:\Users
	+ cd ..: カレントディレクトリを，一つ上の階層のディレクトリに移動します．
	+ cd（引数なし）：現在のカレントディレクトリを表示（ただし，プロンプトに既に表示されている情報と同じ）
* **mkdir**: 新規ディレクトリ（フォルダ）を作成するコマンド．
	+ 使用例: mkdir project\_folder または mkdir "My Project"
* **rmdir**: ディレクトリ（フォルダ）を削除するコマンド
	+ 使用例: rmdir /s /q old\_project（/s：サブディレクトリも含めて削除するオプション，/q：確認メッセージを表示しないオプション）
	+ rmdir ディレクトリ名: 空のフォルダーを削除します．（/s オプションで中身ごと削除）
* **del**: 指定したファイルを削除するコマンド．
	+ 使用例: del \*.tmp （拡張子が「tmp」であるすべてのファイルを削除）
	+ del ファイル名: ファイルを完全削除します（ごみ箱には移動されません）．/f オプションで読み取り専用ファイルも削除

ファイル内容の表示

* **type ファイル名**: テキストファイルの内容を表示します
* **more ファイル名**: 内容をページ単位で表示します
	+ スペースキー：次ページへ移動
	+ Enterキー：次行を表示
	+ Qキー：表示を終了

ファイルとフォルダーの操作

* **move 元名 新名**: ファイルまたはフォルダーの名称変更や移動
* **copy 元名 先名**: ファイルの複製を作成
* **xcopy /s /e /h 元フォルダー 先フォルダー**
	+ /s: 非空のサブディレクトリを含めて複製
	+ /e: 空のサブディレクトリを含めて複製
	+ /h: 隠しファイルとシステムファイルも複製

**実行例**

1. コマンドプロンプトを管理者として実行（手順：Windowsキーまたはスタートメニュー，「cmd」と入力，「管理者として実行」を選択）
2. 次のコマンドを実行．notepad はテキストエディタ（メモ帳）を起動するコマンド．
* dir
cd ..
mkdir test\_folder
notepad

### 4.4. エラーメッセージとの付き合い方

エラーメッセージは「失敗」ではなく，**「問題解決へのヒント」**です．エラーメッセージには，どのファイル（File）の何行目（line）で，どのような種類のエラー（例: ModuleNotFoundError）が起きたかなどが記載されています．エラーメッセージの一部や全体をコピーしてインターネットで検索したり，AIアシスタントで調査すると，解決策が見つかることが多いです．

## 5. Windows でのAI開発環境（入門編）インストール手順

**注意:**

* 以下の手順では主に**コマンドプロンプト（cmd.exe）**を使用します（4.3節参照）．一部の操作では「管理者として実行」が必要です（4.3.2節参照）．
* wingetコマンドが利用できない場合は，Microsoft Storeから「アプリ インストーラー」をインストールする必要があります．
* 以下のコマンド例では，コメントを示すために REM を使用しています．

### 5.1. AI開発環境（入門編）のインストールと設定

#### 5.1.1. AI開発環境（入門編）の一括インストール

PythonはAI開発で広く使われる言語です．wingetはWindowsのコマンドラインからアプリをインストールできるツールです．

**Pythonバージョンについて：** 2025年6月現在，Python 3.13.4が最新安定版です．ただし，AI開発の入門としてはPython 3.12系列の利用を推奨します．これは多くのAIライブラリがPython 3.12での動作確認が行われており，安定性が確保されているためです． AnacondaやMinicondaなど，他の配布版もありますが，このガイドでは，wingetを用いてPythonを本体をインストールします．

**インストール**は，ソフトウェアをコンピュータで実行可能な状態にするための処理のことです．プログラムファイルの配置，システムへの登録，設定などを行います．

**winget**（Windows Package Manager：Microsoft公式のソフトウェア管理ツール）は，ソフトウェアのインストールや管理を行うための**コマンドラインツール**（文字コマンドで操作するプログラム）です．インストール作業を簡単なコマンドで実行可能になります．

**インストールスコープ**は，ソフトウェアインストール時の対象ユーザーの範囲の指定です．wingetを用いたインストールにおいても，重要な設定です． 例えば，winget では，「--scope machine オプション（コマンドの動作を変更するための追加設定）」を指定してPythonをインストールすると，そのコンピュータのすべてのユーザーがPythonを使用できます．このオプションを指定しない場合，インストールを実行したユーザーのみがPythonを使用可能となります．

* --scope machine: すべてのユーザーが使用可能（インストール時に管理者権限必須）
* --scope user: 現在のユーザーのみが使用可能

以下のコマンドで，AI開発環境（入門編）を一括でインストールします。

AI開発環境（入門編）の一括インストール

1. **コマンドプロンプトを管理者として実行**（手順：Windowsキーまたはスタートメニュー，「cmd」と入力，「管理者として実行」を選択）
2. 次のコマンドを１つずつ実行（長いコマンドは，表示が複数行に分かれていますが，１つのコマンドとして実行してください）

**REM Windows のパス長制限を緩和**

**reg add "HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Control\FileSystem" /v LongPathsEnabled /t REG\_DWORD /d 1 /f**

**reg query "HKLM\SYSTEM\CurrentControlSet\Control\FileSystem" /v LongPathsEnabled**

**REM winget コマンドは必ず１つずつ実行すること．（実行完了を待つため）**

**REM Nvidia CUDA 12.6 をシステム領域にインストール**

**winget install --scope machine --id Nvidia.CUDA --version 12.6 -e**

**REM Python 3.12 をシステム領域にインストール**

**winget install --scope machine --id Python.Python.3.12 -e --silent**

**winget install --scope machine --id Python.Launcher -e --silent**

**REM Git, 7zip, Visual Studio Code, Windsurf をシステム領域にインストール**

**winget install --scope machine --id Git.Git -e --silent**

**winget install --scope machine --id 7zip.7zip -e --silent**

**winget install --scope machine --id Microsoft.VisualStudioCode -e --silent**

**winget install --scope machine --id Codeium.Windsurf -e –silent**

**コマンドの説明：**

* **パス長制限の緩和:** Windowsの既定では260文字のパス長制限がありますが，AI開発では深いフォルダ階層を使うことがあるため，この制限を緩和します．
* **CUDA 12.6:** NVIDIA GPUを使用する場合に必要な計算基盤．GPUがない場合はスキップ可能です．
* **–scope machine:** システム全体（全ユーザー）で使えるようにインストールします．
* **–silent:** インストール中の対話的な質問をスキップし，既定の設定で自動的にインストールします．
* -e: パッケージIDの完全一致検索（曖昧な検索を避けます）．

**よくある間違いの回避策**

* コマンドの途中で改行しない
* 管理者権限で実行していることを確認
* インストール中にPCをスリープモードにしない
* Python がすでにインストール済みのときは，「winget install --scope machine --id Python.Python.3.12...」の実行のときに，「Modify Repair Uninstall」画面が表示されることがあります．この画面が出た時は，この画面では「Cancel」をクリックして次に進んでください．
* wingetコマンドが認識されない場合は，Windows 10の場合，Microsoft StoreからApp Installerをインストールする必要があります．

次に，パスを通す設定，環境変数の設定，Python開発に必要なVS Code拡張機能のインストールを行います．最初に**必ず新しいコマンドプロンプトを管理者として実行**します．

**AI開発環境（入門編）の各種設定**

1. **コマンドプロンプトを管理者として実行**（手順：Windowsキーまたはスタートメニュー，「cmd」と入力，「管理者として実行」を選択）
2. 次のコマンドを実行

**set "INSTALL\_PATH=C:\Program Files\Python312"
echo %PATH% | find /i "%INSTALL\_PATH%" >nul
if errorlevel 1 setx PATH "%PATH%;%INSTALL\_PATH%" /M >nul
echo %PATH% | find /i "%INSTALL\_PATH%\Scripts" >nul
if errorlevel 1 setx PATH "%PATH%;%INSTALL\_PATH%\Scripts" /M >nul

set "CUDA\_PATH=C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.6"
if exist "%CUDA\_PATH%" setx CUDA\_PATH "%CUDA\_PATH%" /M >nul
if exist "%CUDA\_PATH%" setx CUDNN\_PATH "%CUDA\_PATH%" /M >nul

set "NEW\_PATH=C:\Program Files\Git\cmd"
if exist "%NEW\_PATH%" echo %PATH% | find /i "%NEW\_PATH%" >nul
if exist "%NEW\_PATH%" if errorlevel 1 setx PATH "%PATH%;%NEW\_PATH%" /M >nul

set "NEW\_PATH=C:\Program Files\7-Zip"
if exist "%NEW\_PATH%" echo %PATH% | find /i "%NEW\_PATH%" >nul
if exist "%NEW\_PATH%" if errorlevel 1 setx PATH "%PATH%;%NEW\_PATH%" /M >nul

set "NEW\_PATH=C:\Program Files\Microsoft VS Code"
if exist "%NEW\_PATH%" echo %PATH% | find /i "%NEW\_PATH%" >nul
if exist "%NEW\_PATH%" if errorlevel 1 setx PATH "%PATH%;%NEW\_PATH%" /M >nul

set "NEW\_PATH=C:\Program Files\Windsurf"
if exist "%NEW\_PATH%" echo %PATH% | find /i "%NEW\_PATH%" >nul
if exist "%NEW\_PATH%" if errorlevel 1 setx PATH "%PATH%;%NEW\_PATH%" /M >nul**

**if exist "C:\Program Files\Microsoft VS Code\bin" cd "C:\Program Files\Microsoft VS Code\bin"
if exist "C:\Program Files\Microsoft VS Code\bin" code --install-extension ms-python.python
if exist "C:\Program Files\Microsoft VS Code\bin" code --install-extension ms-python.vscode-pylance**if exist "C:\Program Files\Microsoft VS Code\bin" code --install-extension MS-CEINTL.vscode-language-pack-ja
if exist "C:\Program Files\Microsoft VS Code\bin" code --install-extension dongli.python-preview

if exist "C:\Program Files\Windsurf\bin" windsurf --install-extension MS-CEINTL.vscode-language-pack-ja

**PATH設定コマンドの説明**

* これらのコマンドは，各ツールの実行ファイルが置かれているフォルダをPATH環境変数に追加します．
* if existとif errorlevelを使って，既に設定されている場合は重複して追加しないようにしています．
* /Mオプションによりシステム環境変数として設定されます．
* これらのコマンドを実行において，特に結果は表示されません．エラーメッセージが出ない場合は実行に成功しています．

**Python と Git のインストール確認**

新しいコマンドプロンプトを開き，以下のコマンドでバージョンが表示されれば成功です．

python --version
git --version

**簡単なPythonの動作確認**

コマンドプロンプトでpythonと入力すると，Pythonの対話モードが起動します．以下のコードを入力してエンターキーを押し，結果が表示されるか確認してみます．

print("Hello, AI World!")

>>>と表示されている行はPythonの入力受付中であることを示します．print("Hello, AI World!")と入力してエンターキーを押すと，その下の行にHello, AI World!と表示されます．確認できたらexit()と入力してエンターキーを押すと対話モードを終了できます．もしここでエラーが出る場合は，8章のトラブルシューティングを参照してください．

#### 5.1.2. 各ツールの概要

* **Python:** AIやデータサイエンスの世界で最も広く使われているプログラミング言語です．
* **Git:** プログラムコードの変更履歴を記録・管理するシステム．GitHubとの同期にも使用されます．
* **7-Zip:** ファイルの圧縮・展開を行うツール．
* **Visual Studio Code (VS Code):** 多機能なコードエディタ．
* **Windsurf:** AI支援機能を特徴とする開発環境．

### 5.2. Pythonライブラリの一括インストール

pipを使って，PythonのAI開発でよく使うライブラリを一括でインストールします．**pip**とは: Pythonパッケージマネージャ（Pythonの追加機能を管理するためのツール）です．ライブラリ（プログラムで使用できる機能の集合）のインストールや管理を行うためのツールです．

* 使用例: pip install numpy
* このガイドでは pip installコマンドは，管理者権限で実行することを推奨します．Windowsのユーザ名が日本語であった場合に発生する可能性があるトラブルを防止するためです．

pip基本コマンド

* pip install [パッケージ名]: 指定したパッケージ（機能をまとめたソフトウェア部品）をインストールします．
* pip install --upgrade [パッケージ名]（--upgrade：既存パッケージを最新版に更新するためのオプション）: 指定したパッケージを更新します．
* pip uninstall [パッケージ名]: 指定したパッケージを削除します．
* pip list: インストール済みパッケージを一覧表示します．
* pip show [パッケージ名]: 指定したパッケージの詳細情報を表示します．

**重要：** 「pip install -U torch torchvision torchaudio --index-url <https://download.pytorch.org/whl/cu126>」の実行では、PyTorch公式サイト（<https://pytorch.org/get-started/locally/>）で，自身の環境（Windows, **Pip**, Python, CUDA）に合った**最新のインストールコマンドを必ず確認**すること．特にGPUを利用する場合は，適切なCUDAバージョンの選択が重要です（7.3節参照）

以下のコマンドで，AI開発に必要なライブラリを一括でインストールします（2025年6月版）：

**主要なPythonライブラリの一括インストール**

1. **コマンドプロンプトを管理者として実行**（手順：Windowsキーまたはスタートメニュー，「cmd」と入力，「管理者として実行」を選択）
2. 次のコマンドを１つずつ実行（長いコマンドは，表示が複数行に分かれていますが，１つのコマンドとして実行してください）

**REM pipを最新版に更新
python -m pip install -U pip

REM PyTorchのインストール（CUDA 12.6対応版）
pip install -U torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu126

REM その他のAIライブラリを一括インストール
pip install jupyter ipykernel numpy pandas scikit-learn matplotlib japanize-matplotlib seaborn scipy opencv-python pillow timm pygame streamlit**

**注意：** 上記のコマンドはCUDA 12.6を使用する場合の例です．2025年6月現在，PyTorch 2.6.0および2.7系でCUDA 12.6が正式サポートされています．

**インストールされるPythonライブラリの説明**

* **PyTorch:** ディープラーニング（深層学習）モデルを効率的に作るための主要なライブラリです．
* **torchvision:** 画像処理用のPyTorchライブラリ
* **torchaudio:** 音声処理用のPyTorchライブラリ
* **jupyter:** 開発エディタのJupyter Notebook
* **NumPy, Pandas, Scikit-learn:** 数値計算（NumPy），表形式データの操作・分析（Pandas），古典的な機械学習（Scikit-learn）のための定番ライブラリ
* **Matplotlib / Seaborn:** データの分布や学習結果などをグラフで可視化するためのライブラリで
* **japanize-matplotlib:** Matplotlibで日本語を表示するためのライブラリ
* **scipy**: 科学計算ライブラリです（SciPy：高度な科学計算機能を提供）．
* **opencv-python**: 画像処理・コンピュータビジョン用ライブラリ．
* **pillow**: 画像処理ライブラリです（Pillow：画像の読み込み，編集，保存機能を提供）．
* **timm:** PyTorch Image Models（事前学習済みモデル集）
* **pygame:** ゲーム開発・インタラクティブアプリケーション用ライブラリ
* **streamlit**: Webベースの対話アプリ開発ツール

**インストール後の確認コマンド**

# GPU情報確認
wmic path win32\_VideoController get name

# Pythonバージョン確認
python --version

# PyTorchとCUDA対応確認
python -c "import torch; print(torch.\_\_version\_\_, torch.cuda.is\_available())"

**ライブラリの動作確認**

インストールしたライブラリが正しく使えるか，簡単なPythonコードで確認してみます．コマンドプロンプトでpythonと入力し，対話モードに入って以下のコードを実行します．

# PyTorchのバージョン確認とGPU利用可能性の確認
import torch
print("PyTorch Version:", torch.\_\_version\_\_)
print("CUDA Available:", torch.cuda.is\_available()) # GPUが使えるか（TrueならOK）
if torch.cuda.is\_available():
 print("CUDA Version:", torch.version.cuda) # PyTorchが対応しているCUDAのバージョン

# NumPyのバージョン確認
import numpy as np
print("NumPy Version:", np.\_\_version\_\_)

# Pandasのバージョン確認
import pandas as pd
print("Pandas Version:", pd.\_\_version\_\_)

# Matplotlibのバージョン確認
import matplotlib
print("Matplotlib Version:", matplotlib.\_\_version\_\_)

エラーが出ずにバージョン情報などが表示されれば成功です．特にCUDA Available: Trueと表示されていれば，PyTorchがGPUを認識しており，GPUを使った計算が可能です．Falseと表示された場合は，GPUが搭載されていないか，CUDAドライバやPyTorchのインストールに問題がある可能性があります．確認できたらexit()と入力して対話モードを終了します．動作確認でエラーが出た場合は，8節のトラブルシューティングを参照してください．

## 6. 開発エディタの概要

5.1でインストール・設定したVisual Studio CodeとWindsurf，および5.2でインストールしたJupyter Notebookについて，それぞれの特徴と適した用途を説明します．開発エディタにはそれぞれ特徴があり，併用も可能です．

**① Visual Studio Code (VS Code)**

**特徴:** Microsoftが開発するコードエディタです．「拡張機能」により，Python開発支援など様々な機能を追加してカスタマイズできます．

**適した用途:** プログラム開発，複数のファイルで構成されるプロジェクトの管理．

**主なAI機能:** GitHub CopilotやClaudeなどの拡張機能を導入することで，コード補完やチャットによる質問応答，コード生成といったAI支援機能を利用できます．

**5.1.1でインストールした拡張機能の説明**

* **ms-python.python:** Python言語の基本サポート（シンタックスハイライト，デバッグ，リンティングなど）
* **ms-python.vscode-pylance:** Pythonの高度な言語サポート（型チェック，自動補完，定義へのジャンプなど）を提供する言語サーバー
* **MS-CEINTL.vscode-language-pack-ja:** VS Codeの日本語化パック
* **dongli.python-preview:** Pythonコードの実行結果をプレビュー表示する拡張機能

**② Jupyter Notebook**

Webブラウザ上で動作する対話的な開発環境です．「セル」と呼ばれるブロック単位でコードを書き，実行するとその結果（数値，表，グラフなど）がすぐ下に表示されます．シンプルなインターフェースになっています．より多くの機能が必要になったらJupyterLabへの移行も検討できます．

**適した用途:** データ分析，アルゴリズム，学習過程や結果の可視化，デモンストレーション．

**主なAI機能**

* **コードの自動生成:** 「# titanicデータセットを読み込み，最初の5行を表示」といったコメント（自然言語）をセルに書くと，AIが対応するPythonコードを自動で生成してくれます．
* **コードと結果の解説:** 実行したコードや表示されたグラフが何をしているのか，自然言語で解説を生成させることができ，学習の助けになります．

**③** **Windsurf**

**特徴:** AI統合開発環境（IDE）です．エディタ機能とAIとの連携が特徴です．

**主なAI機能**

* **Cascade（エージェント型チャットボット）:** 「# ファイルアップロード機能を持つWebアプリの雛形を作って」のような指示からコード生成やプロジェクト構築を行うなど，対話によるコード生成が可能です．
* **Context Awareness（文脈認識）:** 開いているプロジェクト全体のコードをAIが自動的に理解し，別のファイルに存在する関数や変数を考慮した上で，適切なコード補完や修正案を提示します．

## 7. このガイドのAI環境開発環境（入門編）のメリットと注意点

### 7.1. メリット

* **シンプルさ:** インストールするツールの数を絞り，AI開発を始めやすい構成です．
* **標準的な構成:** Python + pipの構成は広く利用されています．
* **開発エディタの選択肢:** 開発スタイルに合わせて，VS Code, Jupyter Notebook, Windsurfといったエディタから選択できます．

### 7.2. 注意点と潜在的な課題

* **バージョン互換性:** GPUを使う場合，NVIDIA CUDAとPyTorch（が要求するCUDAバージョン）の組み合わせには注意が必要です（7.3節参照）．
* **コマンド操作への慣れ:** ライブラリのインストールなど，基本的なコマンドプロンプトの操作に慣れる必要があります（4章参照）．
* **複数プロジェクトの管理:** このシンプルな構成では，すべてのライブラリがPC全体で共有されます．複数のプロジェクトで異なるバージョンのライブラリを使いたい場合は，環境管理（例: Pythonのvenv機能やconda）が必要になる点に留意してください．

### 7.3. 注意点：GPU利用時のPyTorchのバージョン互換性

1. 使用したいPyTorchのバージョンを決める: 最新安定版が推奨されます．
2. PyTorch公式サイトのインストールページを確認する．
	* PyTorch公式サイト（<https://pytorch.org/get-started/locally/>）にアクセスします．
	* 「PyTorch Build」（Stable），「Your OS」(Windows)，「Package」(**Pip**を選択)，「Language」(Python)，「Compute Platform」を選択します．
	* 「Compute Platform」で，利用したいCUDAのバージョン（例: CUDA 11.8, CUDA 12.6）を選択します．
	* 選択すると，下にインストールコマンドが表示されます（例: pip install torch ... --index-url .../cu126）．このコマンドの末尾（cu126など）が，PyTorchが対応するCUDAバージョンを示します．
3. 必要なCUDAのバージョンを確認する．

**最新の互換性情報は** [**https://docs.nvidia.com/cuda/cuda-toolkit-release-notes/**](https://docs.nvidia.com/cuda/cuda-toolkit-release-notes/) **で確認してください．**

## 8. 一般的なトラブルシューティング

### 〇　問題: pythonやpipコマンドが認識されない

**エラーメッセージ例:** 'python' is not recognized as an internal or external command, operable program or batch file.

**原因:** wingetでのインストールに失敗したか，パス（PAT）が正しく設定されていない可能性があります．

**対処法:** PCを再起動してみてください．それでも解決しない場合は，Windowsの「設定」→「システム」→「バージョン情報」→「システムの詳細設定」→「環境変数」を開き，システム環境変数のPathにPythonのインストールフォルダ（例: C:\Program Files\Python312）が含まれているか確認してください．

### 〇　問題: ライブラリが見つからない

**エラーメッセージ例:** ModuleNotFoundError: No module named 'xxx'

**原因:** 必要なライブラリ（例: torch, numpy, pandasなど）が正しくインストールされていないか，コマンドを実行しているPython環境とは異なる環境にインストールされている可能性があります．

**対処法:** pip install <ライブラリ名>コマンドを再実行してみてください．また，pip listコマンドでインストール済みのライブラリ一覧を確認できます．

### 〇　問題: GPUが認識されない，またはGPU関連のエラーが出る

**エラーメッセージ例:** CUDA is not availableなど

**原因:** CUDA 12.6と，pipでインストールしたPyTorchのCUDAバージョンの間に互換性がないことが主な原因です（7.3節参照）．その他，CUDAのインストール失敗，GPUハードウェアの問題なども考えられます．

**対処法:**

1. 7.3節の手順を参考に，インストールされているCUDA 12.6と，pipでインストールしたPyTorchのCUDAバージョン（コマンドの--index-url部分やPyTorch公式サイトで確認できる）の互換性を再確認してください．必要であれば，CUDAまたはPyTorchを再インストールしてください．
2. コマンドプロンプトでnvidia-smiを実行し，NVIDIAドライバが正しくインストールされ，GPUがシステムに認識されているか確認してください．GPUの状態やドライババージョンが表示されます．
3. Pythonインタープリタを起動し，以下のコードを実行してPyTorchがGPU（CUDA）を認識しているかプログラムで確認してください．
* import torch
print(torch.cuda.is\_available())
print(torch.version.cuda) # PyTorchが対応しているCUDAバージョンを表示
* Trueが返ってくればPyTorchはGPUを認識しています．Falseの場合は，バージョン不整合の可能性が高いです（7.3節参照）．pip uninstall torchで一度アンインストールし，PyTorch公式サイトで確認した正しいコマンドでPyTorchを再インストールすることを検討してください．
1. PCの再起動も試してみてください．
2. 次の手順で，NVIDIAドライバの更新も試みてください．
* NVIDIA公式サイトのドライバダウンロードページにアクセス．
* GPUモデル、OS、ドライバタイプを選択．
* 最新版をダウンロードし、インストーラーを実行（「クリーンインストール」推奨）．
* 確認 (コマンドプロンプト): nvidia-smi

## 9. このガイドを終えたら：次のステップへ

これで，Windows PC上にAI開発の基本的な環境が構築できました．

この後，AI開発やプログラミング学習をさらに進めるためのステップをいくつか案内します．

* **Pythonの基本を学ぶ:** まだPythonに慣れていない場合は，変数，オブジェクト，メソッド，クラス，インポート，関数，データ型，制御構文（if文，for文）といった基本的な文法や概念を学ぶのが最初のステップです．
* **Pythonの基本的なライブラリを使ってみる:** インストールしたNumPyやPandasなどを使って，データの読み込み，表示，集計などの操作を試してみましょう．データに触れることはAI開発の基礎になります．
* **開発環境を使いこなす:** WindsurfのAI機能やデバッグ機能を使ってみたり，Jupyter Notebookでデータ分析を進めてみたりなど，開発環境の便利な機能を積極的に使ってみましょう．
* **仮想環境について学ぶ:** 複数のプロジェクトで異なるライブラリのバージョンを使いたいなど，より進んだ環境管理が必要になったら，Pythonの仮想環境（venvやcondaなど）について学ぶことをお勧めします．

このガイドで構築した環境を使い，様々なコードを試すことができます．AI開発の学習が充実したものとなることを期待しています．