スクリプト言語プログラミング Pythonによる数値解析

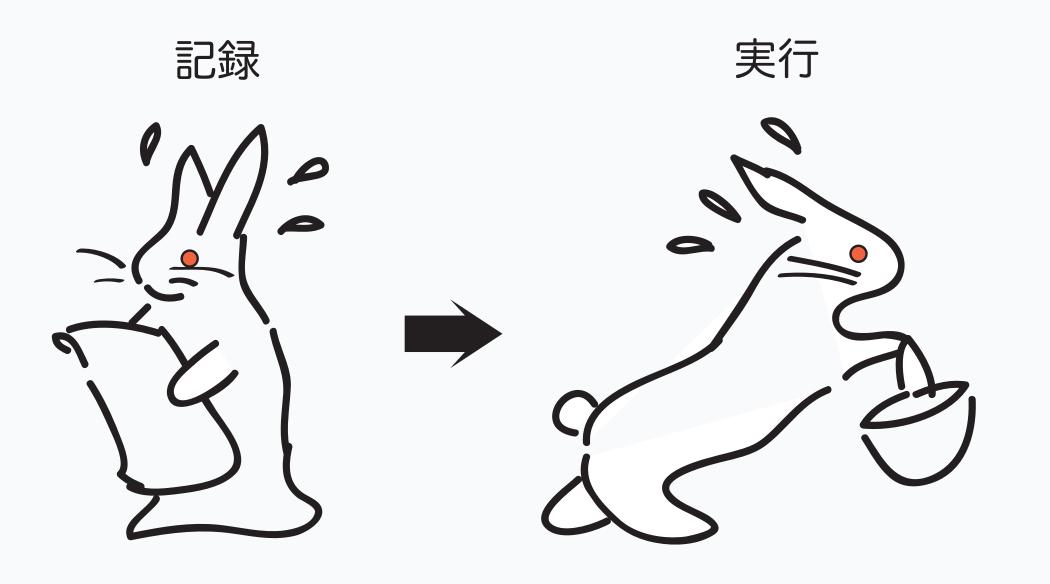
第1回講義資料 箕原辰夫

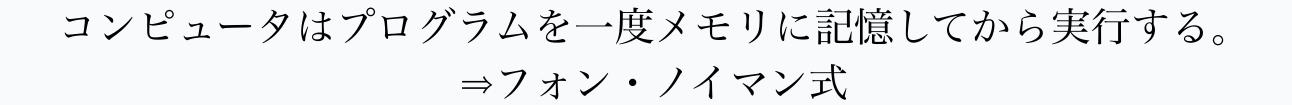
プログラミングの世界



テキストとして記述したものがコンピュータの中 では実行することができる

コンピュータの動き







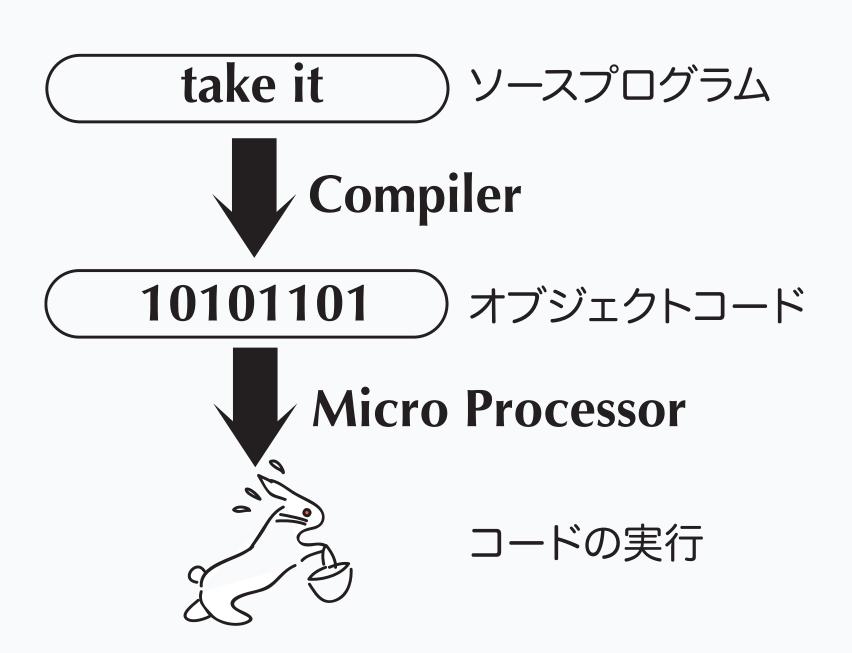
プログラムのレベル



人間が記述するプログラムとコンピュータが実行でき るプログラムの表現方式が異なる。

そのギャップを埋めるために3つの実行方式がある

コンパイラ方式

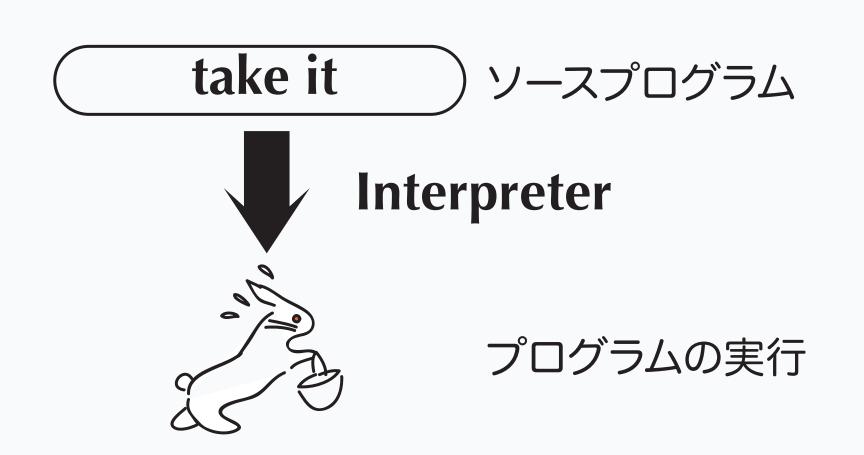


予めコンパイラでコンピュータレベルのプログラム(マシン語のプログラム)に 変換しておく。

⇒高速に実行できるが、各マシン (CPU) やOSごとに変換しなければならない

例:C/C++, Rust, Fortran, Swift, Julia, Dartなど

インタープリタ方式

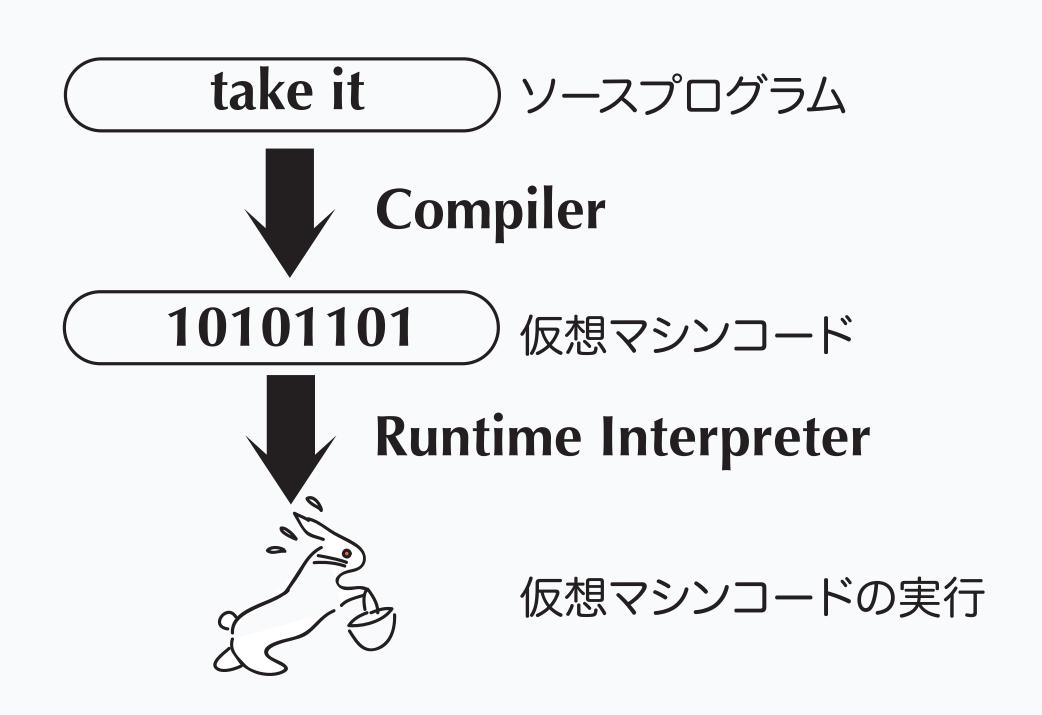


インタープリタが、いちいち解釈しながら実行する。インタープリタさえあればどこでも実行される。

⇒実行が低速になる。

例: Python, Lua, Ruby, Perl, C-shell, Lisp, JavaScript, Swift, Juliaなど

中間(仮想)コード方式



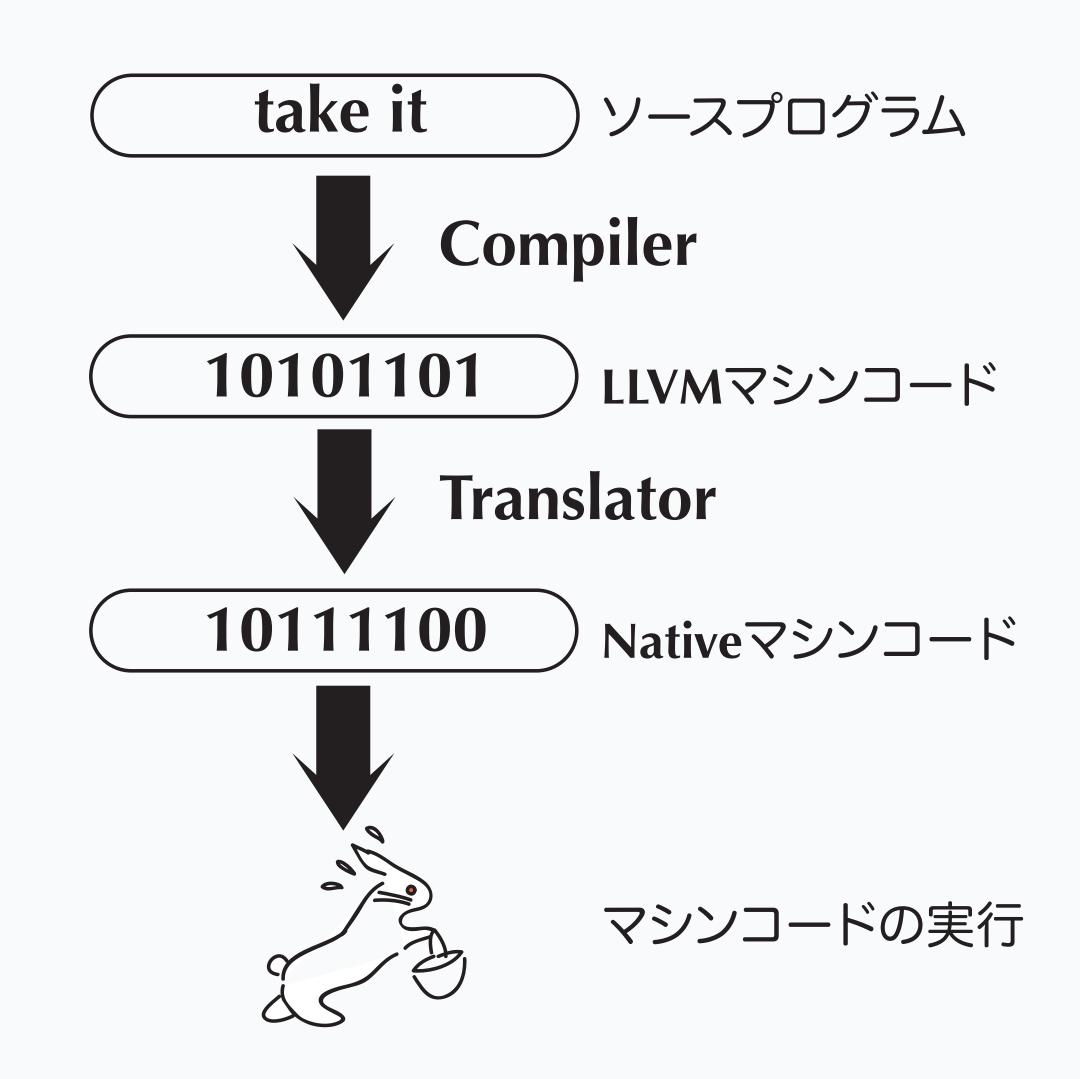
両方式の中間的なもの。特定のマシン語のプログラムではなく、仮想マシンのプログラムに変換する

⇒ランタイム・インタープリタは、割と小さく高速に動くプログラムなので、そこそこの速度で実行可能

例:Java, Pascal, Smalltalk, C#など

コンパイラ方式も中間コード方式を採るようになった

- プログラミング言語のコンパイラが、 LLVM(Low Level Virtual Machine)の命令 コードにコンパイルする
- LLVMの命令コードをtranslatorが、それ ぞれのコンピュータのCPUの命令コード (Native Code) に変換する
- ソースプログラムをLLVMまでコンパイル するコンパイラだけを持ったプログラミン グ言語が増えてきている



速度の差

- 繰返しを使った、簡単な積和計算の実行時間
- https://github.com/Acmion/ ComparisonCythonPythonJuliaCSharpC より
- Cythonは、Pythonと似ているが文法が少し異なるコンパイル言語で、C言語と同様の速度で実行が可能になっている
- Juliaも、Pythonと似ている関数型プログラミング言語だが、コンパイルしてからインタープリタで実行する
- 仮想コード方式の言語 (C#) は、コンパイル言語の10倍ぐらい
- インタープリタ言語は、50~200倍ぐらい時間が掛かっている
- ただし、Pythonの場合は、C言語でコンパイルされたライブラリを使うと、2~3倍程度の時間で実行することが可能

Language	Mean Execution Time (s)
С	0.032200
Cython	0.028999
Julia	0.053200
C#	0.299488
Python	6.353125

Pythonの系譜

Python

▶ Modula-3などの言語仕様を受けて、インタープリタとして作られている。最初 のバージョンは、Python 0.9→1.0としている。

• Python2

Python 1の後継として2000年にリリースされて爆発的に使われるようなった。
 そのときのライブラリがPython2をベースにしたものが多かったのと、print文が使いやすいので生き残っている。2.7版が最新

• Python3

▶ Python 2に新しい言語要素を足したもの。各種ライブラリが対応するようになった。3.12版が安定版(最新版で3.13)

macOS Sierra(10.13)以降の場合

- インストールの制限が掛かっています
- ターミナルで、以下のコマンドを打って下さい。
- sudo spctl --master-disable
- システム環境設定 >> プライバシーとセキュリティ
- 一般のタブで、ダウンロードしたアプリケーションの実行許可
- ・すべてのアプリケーションをOKにします

Python IDLEのインストール

- Python.orgのDownloadsのページからPython3.12.2のIDLEをインストールしてください。Python 3.13はライブラリが対応していない場合がありますので、ダウンロード・インストールしないで下さい。
 - https://www.python.org/downloads/
- Pythonの版の読み方(xやyには該当する数字が入ります)
 - ▶ Python 2.7.x → Python2 あるいは Python 2.7版
 - ▶ Python 3.x.y → Python3 あるいは Python 3.x版
 - ▶ Python 3.13.x → Python3 あるいは Python 3.13版

Windowsのインストーラでの設定

- 最初の画面で、PATHに入れる項目にチェックを入れる
- Customize installationにおいて、以下の項目にチェックを入れる
 - pip...これにチェックを入れておくと、ライブラリをダウンロード・インストールするためのpipコマンドがインストールされる
 - ▶ Install python for all users...これにチェックを入れておくと、Pythonの実行ファイルがC:\Program Files\Python313のフォルダにインストールされる、そうでないと個人用のフォルダの奥深くにインストールされる
 - Add Python to environment variables...これにチェックをいれておくと、 pythonやpipといったコマンドがPowerShellから直接実行することができる

Windowsでのインストーラ画面 (3.13の場合)

- 最初の画面では、2つのオプションにチェックマークを入れます。
- 特に、Add python.exe to PATHにチェックを入れるの 忘れないでください。
- Python 3.13.3 (64-bit) Setup

 Install Python 3.13.3 (64-bit)

 Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

 Install Now

 C:¥Users¥minoh¥AppData¥Local¥Programs¥Python¥Python313

 Includes IDLE, pip and documentation

 Creates shortcuts and file associations

 Customize installation

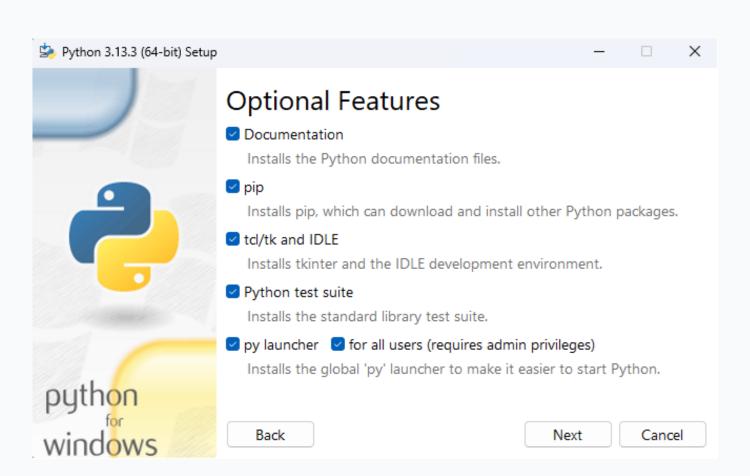
 Choose location and features

 Use admin privileges when installing py.exe

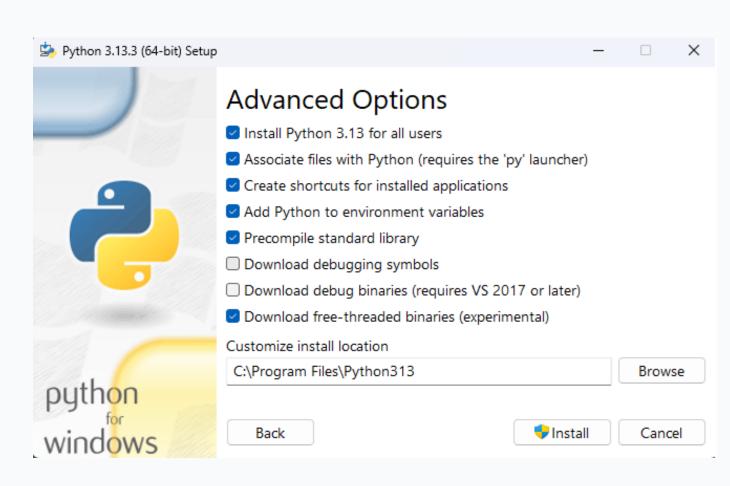
 Add python.exe to PATH

 Cancel

- Customize Installationをクリックします。
- 次の画面では、すべてのオプションにチェックマークを入れます。
- for all usersとpipにチェック マークを入れるのを忘れない でください。



- 最後の画面では、Install
 Python 3.xx for all usersにチェックを入れるのを忘れないでください。
- C:\Program Files\Python313 にインストールされるのを確 認して、Installボタンを押し ます。



Mac OSでpythonを3.0を標準にする

- Mac OS では、Pythonのコマンドは、/usr/local/bin/python3に配置される。
- ターミナルを開き、ps コマンドを入力⇒実行されているシェルの名前が表示される
- bashを使っている場合は、.profileファイルに、zshを使っている場合は、.zprofileに以下を追加(bashおよびzshに共通)なお、bashの場合は、.bashrcファイルに追加するのでも良い
 - export PATH="/usr/local/bin:\$PATH"
- zshを使っている場合で.zprofileがない設定のときは、.zshrcあるいは、.zshenvファイルに以下を追加
 - set path=(/usr/local/bin \$path)

- tcshあるいはcshを使っている場合は、~/.cshrcに以下の設定を追加
 - set path=(/usr/local/bin \$path)
- pythonコマンドを/usr/local/binの下に作成する
 - cd /usr/local/binsudo ln -s python3 python
- シェルで次のことを実行
 - ▶ source .cshrc (csh/tcshの場合)
 - ▶ source .profile (bashの場合)
 - ▶ source .zprofile (zshの場合)
 - ▶ source .zshrc(zshの場合)
- キャッシュ・インデックスの再初期化
 - ▶ rehash (zsh/csh/tcshの場合)
 - ▶ export (bashの場合)

M1~M3などのARM (Apple Silicon) でのPython

- Pythonをインストールすると、Intel CPU用のPythonとARM CPU(M1以降)用のPythonのUniversal2(Mach-o)で2つのPythonがインストールされる
- Python IDLEでは、ARM用のPythonが稼働する
- ターミナルで、コマンドとして起動するときは、以下のように場合分けされる
 - ▶ python3 (/usr/local/bin/python3) …標準版のPythonが起動される (ShellがIntelモードで動いているときはIntel版が起動され、ARMモードで動いているときは、ARM版が起動される)
 - ▶ python3.13 (/usr/local/bin/python3.13) …同上
 - ▶ arch -arm64 python3...強制的にARM版のPythonを起動したいとき
 - ▶ python3.13-intel64(/usr/local/bin/python3.13-intel64)...強制的にIntel版のPythonを起動したいとき

PythonのIDLEを起動する

- Mac OS Xの場合:
 - ▶ Applications(アプリケーション) >>> Python 3.13 >>> IDLE.appを起動する
- Windowsの場合:
 - スタートアップメニューのすべてのアプリ>>>> Python 3.13>>>> IDLE(Python 3.13)

環境設定

- Macintosh: IDLEメニュー>>> Preferences
- Windows: Optionsメニュー>>> Configure IDLE
- ・エディタのフォント
 - ▶ タブ: Fonts/Tabs
 - Font Face
 - Meiryo UI、メイリオ、Lucida Grandeなどに
 - Size
 - ▶ 22 pt以上にする

インタープリタの実行

• 立ち上げたウィンドウ(shellという名前が出ている)の中で>>>の出ているところで、何かの計算式を入れたりして、最後に改行(Enter/Return)キーを押すと、その場で実行が可能になっている。

高速なPython実行系

- PyPy
 - ▶ Pythonより7倍ぐらい高速に実行してくれる。ただし、Num.pyなどの数値計算ライブラリに対応が不充分
- Numba (Anaconda)
 - ▶ @jitを付けるだけで、その関数をコンパイルするので、C言語ぐらい高速に実行 してくれる。
- Cython
 - Pythonそのままでなくて書き直しを迫られる。ただし、コンパイルするので、 C言語と同様の速さで実行してくれる。

他のPython開発環境(1)

- BBEdit (フリーMacOS版)
 - ▶ Xcode がインストールされていると、Xcodeのインタープリタを使う
 - ▶ 一行目に#!/usr/local/bin/pythonを書くとインストールされたPythonになる
- Visual Studio Code (フリー版)
 - ▶ Python用の拡張設定をダウンロードする必要がある。
- Visual Studio Express (フリーWindows版)
 - ▶ PTVSというプラグインで対応している

他のPython開発環境(2)

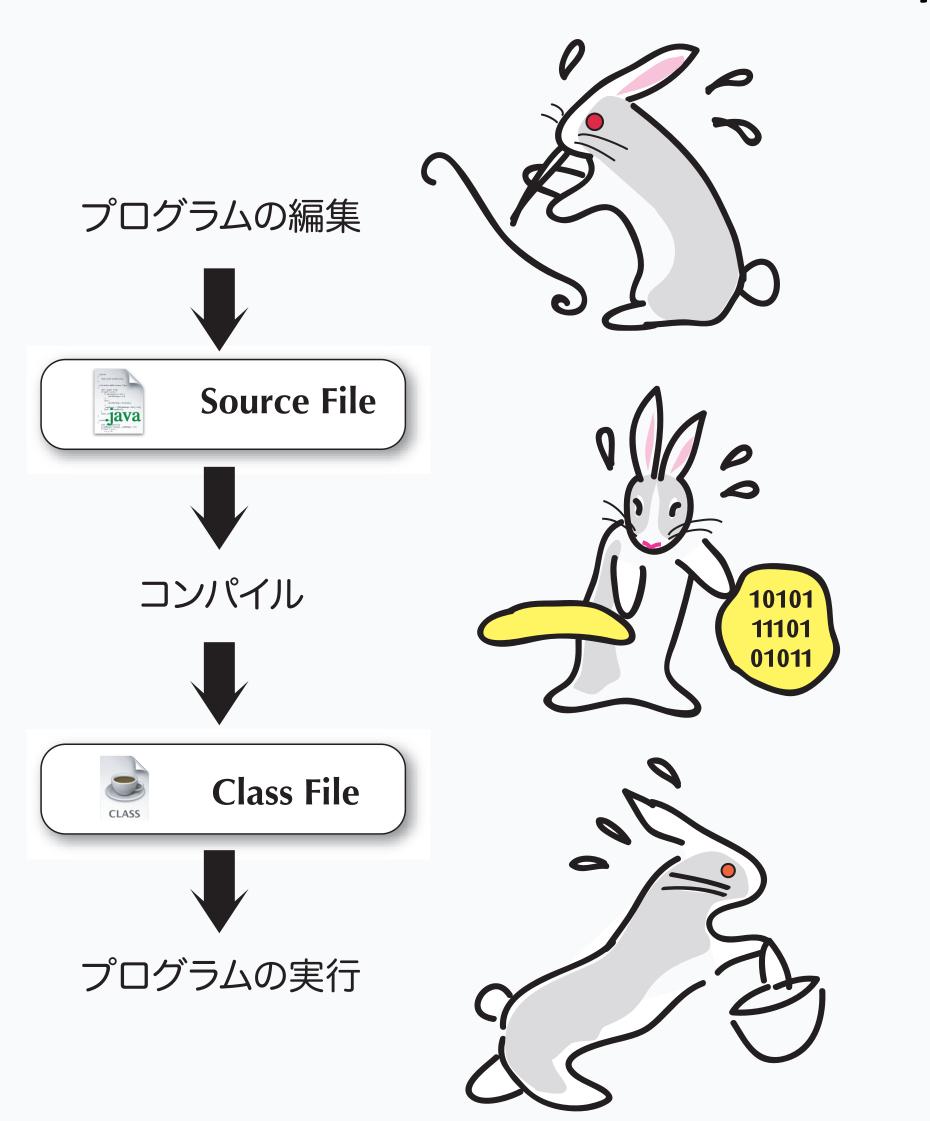
- Eclipse (フリー版)
 - ▶ Pydevというプラグインで対応している
- Xcode
 - ▶ 独自にPythonのインタープリタを用意している
 - /Applications/Xcode.app/Contents/Developer/usr/bin/python3
- PyCharm
 - ▶ 良く使われている開発環境、プロジェクトベース

Webベースの開発環境

- ローカルな環境で実行させる場合は、コピー&ペーストでテキストエディタなどで、.pyファイルとして保存する必要がある
- サーバー側のPythonの環境で実行され、ライブラリなどもある程度揃っている
- Jupyter
 - Jupyter notebook と Jupyter Labがある
 - ▶ 2025年4月の段階では、Python 3.12.7までをサポートしている
 - ▶ sympyなど数式などをきれいに表示してくれる
- Google Colaboratory
 - https://colab.research.google.com/notebooks/welcome.ipynb?hl=ja
 - ▶ 2025年4月の段階でも、Python 3.11.11までしかサポートされていない
 - ▶ サーバー側を3.12や3.13にアップデートできるが、いちいち.pyファイルをドライブに転送する必要がある
 - ▶ Jupyterと同様なWebベースの開発環境(Googleアカウント必要)

プログラムの開発

- 編集
- 保存
- 実行



プログラムの保存と実行

- MacOSX: Finder →「新規フォルダ」で新規フォルダをデスクトップ上に作成、フォルダ名は、Script2025
- Windows, MacOSX:デスクトップ上で右クリックでメニューを出して、「新規フォルダ」で新規フォルダをデスクトップ上に作成、フォルダ名は、Script2025
- IDLE上で「File」 → 「New File」を押す
- ・ 作られたウィンドウ上で「File」→「Save」でファイル名を入力(大文字から始めて、.pyの拡張子で終わるように)
- 編集、保存(コンパイルで自動的に)
- コンパイル・実行は、「Run」→「Run Module」で。F5がショートカットキーになっている

エラーがあったら

- ・該当個所を直して、
 - ▶ 保存して
 - > 実行
- ・プログラムの置かれる場所
 - ▶ デスクトップ >> Script2025
 - ▶ .py (ソースプログラム) ファイル
 - ▶ .pyc (コンパイルされた) ファイル

拡張子を表示

- Macintosh: Finderの環境設定
 - ▶ 「詳細」タブで「すべてのファイル名の拡張子を表示」にチェックを入れる
- Windows: ファイルエクスプローラ
 - ▶ 表示ツールバー>> ... >> オプション
 - ▶ 「表示」タブで「登録されている拡張子を表示しない」についているチェック マークをクリックで外す

Pythonの主要なライブラリ

- 標準ライブラリ
- Numpy...行列計算用のライブラリ
- Scipy...数値解析用のライブラリ(Numpyを利用)
- Matplotlib…グラフ表示用
- Sympy...数式処理用
- pandas...統計処理用
- PyQt5...2次元GUIライブラリ
- wxWidgets... 2 次元GUIライブラリ
- Panda3D...3次元グラフィックス
- scikit-learn...機械学習(クラスタリング・主成分分析・回帰など)
- PyTorch...ニューロンをシミュレートした深層学習(Deep learning)

Pythonライブラリの場所

- Windowの場合
 - ▶ 全体: C:\Program Files\Python313
 - 個人: C:\Users\ユーザ名\AppData\Local\Programs\Python\Python313

• Mac OS Xの場合

- 上全体: /Library/Frameworks/Python.framework/Versions/3.13
- 個人:/Users/ユーザ名/Library/Python/3.13

数値計算用のライブラリのインストール

- Python3だけがある場合、ターミナルから以下を実行する。sudoをしておくと、共通のライブラリのフォルダにインストールされる
 - sudo pip3 install numpy scipy matplotlib sympy
 - ▶ pip3が動かない場合は、以下のようにする
 - sudo python3 -m pip install numpy scipy matplotlib sympy
- M1以降のMacの場合、ARM64用のライブラリをインストールする必要がある。そうでないと、ターミナルの種類によっては、Intel64版がインストールされてしまうことがある。強制的にarm版にするには、arch -arm64の指定をする
 - arch -arm64 pip3 install numpy scipy matplotlib sympy
 - sudo arch -arm64 pip3 install numpy scipy matplotlib sympy
- Python2と併用の場合
 - pip3 install numpy scipy matplotlib sympy
- Windowsは、コマンドプロンプト・PowerShellから以下を実行(Windowsキー+Xで、ターミナル(管理者)でを選んで以下のことを実行)
 - pip install numpy scipy matplotlib sympy

pipでのインストール場所

- pipでインストールされたライブラリは、下記のフォルダに入っている(3.10以降は個人でライブラリをインストールした場合は、標準で下記のフォルダに入るようになった)
- Mac OSの場合(個人)
 - ▶ /Users/ユーザ名/Library/Python/3.13/site-package
- Windowsの場合 (個人)
 - ▶ C:\Users\ユーザ名\AppData\Roaming\Python\Python313\site-package
- 全体でのインストールされた場合は、以下のフォルダになる
- Mac OSの場合(全体)
 - ▶ /Library/Frameworks/Python.framework/Versions/3.13/lib/python3.13/site-packages
- Windowsの場合
 - C:\Program Files\Python313\lib\python313\site-packages

M1/M2/M3/M4 (ARMアーキテクチャ) のPython

- Mac OS上のPython自体は、Mach-o形式になっていて、Intel版のバイナリとARM版のバイナリの両方を持っている
- Python IDLEは、ARM版のPythonを起動するが、pipでライブラリをインストールしたときに、Intel版のシェル上でインストールしようとした場合などは、Intel版のバイナリをインストールする可能性がある。
- ターミナルからpythonやpipを起動する場合は、以下のように対応すること

• Pythonの起動

- ▶ /usr/local/bin/python ...Shell (zsh等) のアーキテクチャに従う (Shellが Intelアーキテクチャで起動されていたら、Intel版のPython が起動され、ARMアーキテクチャで起動されていたらARM版のPythonが起動される)
- ▶ /usr/local/bin/python3 ...同上
- ▶ arch -arm64 python …強制的にARMアーキテクチャのPythonを起動したい場合
- ▶ /usr/local/bin/python3.13-intel64 ...強制的にIntelアーキテクチャのPythonが起動される

• pip3の起動

- ▶ pip3 install ライブラリ名 ...Shellに従ったアーキテクチャのライブラリがインストールされる
- ▶ arch -arm64 pip3 install ライブラリ名 ...ARMアーキテクチャのライブラリがインストールされる
- ▶ 一度、Intelアーキテクチャのライブラリをインストールしてしまっていた場合は、--force-reinstallのオプションをつけると、上書きされる

pyenvでの設定

- Pythonプログラムのテキストの一行目に以下の一文をいれるとPythonを実行するインタープリタを指定することができる (macOSの場合)
 - #!/usr/local/bin/python
- pyenvがインストールされている場合は、以下のコマンドが使える
 - ▶ pyenv version → 現在のPythonのバージョンを表示
 - ▶ pyenv install --version →インストール可能なPythonインタープリタ・ライブラリを表示
 - ▶ pyenv install 3.13.3 → Pythonインタープリタとして、3.13.3をインストール
 - ▶ pyenv global 3.13.3 → PythonインタープリタをPCの全ユーザに3.13.3に設定
 - ▶ pyenv local 3.13.3 → Pythonインタープリタを自分だけ3.13.3に設定
- pyenvのバージョンが古くて、最新のPythonのインタープリタがリストにないときは、pyenv-updateプラグインをインストール
 - pit clone https://github.com/pyenv/pyenv-update.git \$(pyenv root)/plugins/pyenv-update
 - ▶ pyenv update → pyenv自体が最新のものに更新される
 - ▶ 参考:https://zenn.dev/utah/articles/6b4c5cec60c45b

数値計算ライブラリのインストール完了の確認

- Python IDLEを立ち上げて、以下を入力
 - import numpy
 - import scipy
 - import matplotlib
 - import sympy
- エラーが出なければインストールされている

VSCodeのインストール

- 以下からVSCodeの最新版をダウンロードする
 - https://code.visualstudio.com
 - ▶ Download for Mac / Windowsのボタンを押す
- ・いくつかの初期設定を行なう
 - ▶ ø「歯車の記号」ボタンを押し、「設定」を選ぶ
 - ▶ 「よく使用する項目」から、
 - テキストエディター>>フォント
 - ・ Font Sizeを20pt以上に
 - ワークベンチ >> 外観
 - ・Color Themeを明るいものに「Default light+」など

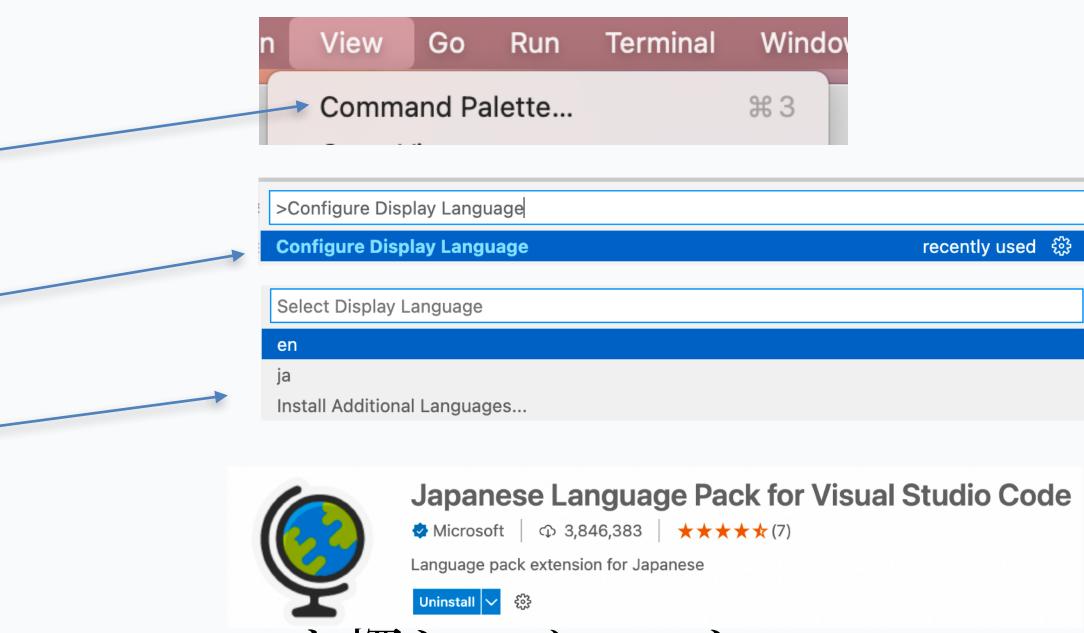


VSCodeの日本語化

- Visual Studio Codeを開く
- ・メニューバーからviewを選択
- command palette を選択
- configure display languageを選択
- install additional languageを選択
 - ▶ 機能拡張のパレットが開くので、

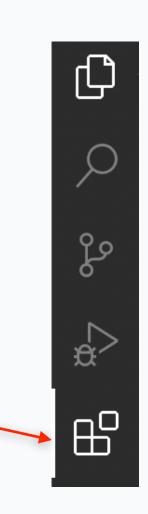
Japanese Language Pack for Visual Studio Codeを探してインストール

トすべてが終わったらVScodeを再起動する



VSCodeでのPythonの機能拡張をインストール

- ・拡張機能のボタンから
- Microsoft Pythonの拡張機能をインストールする
- Pythonで検索
- インストールされているPythonが利用することができる
- 実行のShort Cutキーの設定
 - ▶ 表示のコマンドパレットから、「shortcut」を入力して 「ショートカットキーの設定」を選んで表示させる
 - ▶ 最上段の「検索」のところで、「python run」を入力してフィルタリングする
 - ▶ 「pythonファイルをターミナルで実行」の欄を選んで、ダブルクリックして、「Control + R」を入力してEnterを押す。(Macの場合は「跆+R」でReturn)
- 最初に実行したときに、Pythonのインタープリタのバージョンを尋ねてくるので、3.12.3など該当するものを選ぶ
 - ▶ 後から表示メニューの「コマンドパレット」から「Python: Select Interpreter」でインストールされているインタープリタなどを選ぶことができる





VSCodeでフォルダを開く

- ファイルメニューの「フォルダーを開く」でデスクトップ上のScript2024のフォルダを開く
- 作ったファイルの一覧が、左側のタブの「エクスプローラ」のタブを選べば、表示される
- ファイルメニューの「ワークスペースを保存」で このPythonのワークスペースを保存できる
- practice0101.pyを開いて、レボタンで実行させてみる

VSCodeのインストールの場所

- macOSの場合
 - /Applications/Visual Studio Code.app
- Windowsの場合(個人インストールの場合)
 - ▶ C:\Users\ユーザ名\AppData\Local\Programs\Microsoft VS Code\Code.exe

ドキュメントの場所

- 日本語ドキュメントは、
 - ▶ http://docs.python.jp/3/ 英語版は、https://docs.python.org/3/
 - ▶ チュートリアルや、標準ライブラリの項目をを観て欲しい
- ・教科書は、以下の通り
 - ▶ 英語版:http://www.scipy-lectures.org
 - ▶ 日本語版:http://www.turbare.net/transl/scipy-lecture-notes/index.html(一部翻訳されていない箇 所もある)
- データ分析のための統計学入門は、以下のページに日本語訳のPDFファイルがある
 - http://www.kunitomo-lab.sakura.ne.jp/2021-3-3Open(S).pdf
- MatLabを使った機械学習については、以下のページからeBookがダウンロードできる
 - https://jp.mathworks.com/content/dam/mathworks/ebook/gated/jp-machine-learning-ebook-all-chapters.pdf

iOS用のPythonアプリ

- App Storeからダウンロードする
 - ▶ Carnets Jupyter with Scipy 無料 2025年4月時点では、Python 3.11 sympy, numpy, pandas, scipy 使用可能



▶ Pythonista 3 ¥1,500 評価は非常に良い 2024年4月時点で、Python 3.10まで対応。 sympy, numpy, pandas 使用可能 ただし、scipyは使用不可 Macでも使用可能



- ▶ python3IDE 無料 標準ライブラリのみ
- ▶ python3 IDE Fresh edition 無料
- ▶ Pyto 無料

Android用のPythonアプリ

- Google Playからダウンロードする
 - Pydroid 3 IDE for Python 3 無料主要なライブラリがサポートされているとのこと
 - ▶ QPython3 Python3 for Android 無料 主要なライブラリがサポートされているとのこと
 - ▶ Python Programming Interpreter 無料
 iOS版もあるが、Unicode(日本語も含む)が使えないというユーザからのコ メント有り

その他の開発環境(Anaconda/conda)

- Anaconda自体のWebサイトは以下のところにある
 - https://www.anaconda.com
- Anacondaの個人用の無料版は以下からダウンロードできる(2020年版でも460MB ぐらいある)

https://anaconda.com/products/individual

• 以下のWindows/Mac OS X/Linuxの該当のものを選ぶ

Anaconda Installers		
Windows =	MacOS É	Linux 🗴
Python 3.8	Python 3.8	Python 3.8
64-Bit Graphical Installer (466 MB)	64-Bit Graphical Installer (462 MB)	64-Bit (x86) Installer (550 MB)
32-Bit Graphical Installer (397 MB)	64-Bit Command Line Installer (454 MB)	64-Bit (Power8 and Power9) Installer (290 MB)

Anacondaにおけるconda コマンドラインのパスの設定

- ターミナルのcondaコマンドで、Anaconda Navigatorでできるモジュールのインストール・アンインストール・アップデートなどを行なうことができる。ただし、標準ではcondaコマンドのパスが通っていないので、以下のようにして追加する。
 - ▶ tcsh, cshを使っている場合は .cshrcに、zshを使っている場合は、.zshrcに以下を追加 set path=(\$path /opt/anaconda3/bin)
 - ▶ bashを使っている場合 .bash_profileに、zshを使っている場合(で.zshrcがない場合)は、.zprofileに以下を 追加
 - export PATH=/opt/anaconda3/bin:\$PATH
- Windowsの場合
 - Windowsメニュー >> Windows システムツール >> Control Panel >> システムとセキュリティ >> システム >> システムの詳細設定 >> 環境変数 >> Pathを選び編集ボタンを押す
 - 以下を追加する(なおインストールのときにPATH設定にチェックをしておけば自動的にPathに入ってくる)C:\ProgramData\Anaconda3\Library\bin

Anaconda Navigatorから起動できるIDE

Spyder

- ▶ 昔ながらの標準的なエディタ、実行はIPythonで行なっている
- Jupyter Notebook, JupyterLab
 - ▶ Jupyter系列は基本的にWebブラウザ上でも稼働する。数式などが設定するときれいに表示される、保存ファイルは jupyterの独自形式になってしまうので、該当のプログラム部分を選んで、.pyの形で保存しないと、他の環境からは実行できない
 - ▶ 同じような環境としてGoogle Collaboratoryがある(情報基礎で今年度から使用している)Jupyterと同様にWebブラウザ 上で稼働する

VSCode

- ▶ フリーで開発されたものをMicrosoftが買い上げた?もの、Pythonだけでなくて、ほとんどのプログラミング言語を開発 できる
- ▶ Python用には、専用のプラグインをダウンロードする必要があるが、たぶん、Anacondaのものは、設定されているに違いない

PyCharm

▶ プロジェクトを作って、その中でいくつかの.py形式のテキストファイルを作って実行するもの、フリー版は永く使われてきている

Anacondaのモジュールのインストール・アップデート

- Anaconda Navigator
 Ø Environments
 - メニューのInstalledにリストアップされているものが、インストールされているモジュール
 - メニューのUpdatableにリストアップされているものが、最新版があるもの>>> 選択して下の方に表示される「Apply」ボタンで更新版がダウンロードされ、インストールされる
- Mac OS Xの場合、新しいAnaconda自体を再インストールすると/opt(通常は見えないフォルダ)の下にanaconda3のフォルダが作られてそこにインストールされるので注意すること。
- Anacondaは、通常のPythonのバージョンより、バージョンアップが遅れることが多い。また、Anacondaに登録されているライブラリ以外の、独自のライブラリなどを追加できないという制約があるので、あまりお薦めしない。

その他の開発環境(Spyder)

- Spyderは、エディタ中心の統合的開発環境ではあるが、見た目が良くない(エディタで等幅フォントしか使えない)のであまりお薦めしない。
- Spyderには、IPythonという、少し進んだインタープリタが付属している。
- SpyderのWebページからダウンロード・パッケージファイルをインストールする
 - https://www.spyder-ide.org
- 起動の仕方
 - ▶ Windowsの場合、スタートアップメニューから、Spyderを立ち上げる
 - ▶ MacOSの場合、アプリケーションのフォルダ>>> Spyder.appを起動する
- 使い方は、Python IDLEと似ているが、独自のPythonインタープリタを有しており、Anaconda と同様にバージョンが、標準のバージョンに比べて、遅れがちになっている。また、独自のライブラリを追加するのも、Anacondaと同様に面倒になっている。

その他の開発環境(PyCharm)

- PyCharmは、昔良く使われていたが、現在は、コアなファンしか使っていないのではないだろうか?
 - https://www.jetbrains.com/pycharm/
- ダウンロードのページに行き、Community版をダウンロードする
- プロジェクトを作成する
- プロジェクトにファイルを追加する
- ファイルを保存して、上部パネルの▶ (実行) 記号をクリックする(あるいは、Runメニューの最初の項目Run 'モジュール名'を選ぶ)と、ターミナルパネルが下に割れて開いて、実行結果が表示される

Web上のインタープリタ(Jupyter/Colaboratory)

- Jupyter Lab/Note
 - https://jupyter.org/try-jupyter/
 - ▶ pyodideのインタープリタが動きます(2025年4月現在は、3.12.7版のインター プリタ)
- Google Colaboratory
 - https://colab.research.google.com/notebooks/welcome.ipynb?hl=ja
 - ▶ 2025年4月の段階でも、Python 3.11.11までしかサポートされていない
 - ▶ サーバー側を3.12や3.13にアップデートできるが、いちいち.pyファイルをドライブに転送する必要がある
 - ▶ Jupyterと同様なWebベースの開発環境(Googleアカウント必要)

Web上のインタープリタ(Replit/Python Tutor)

- replit Python
 - https://replit.com/languages/python3
 - ▶ Googleなどのアカウントを用いてログインします。
 - ▶ create Replボタンで、Pythonのreplを選びます。(2025年4月現在は、最新版は、3.13.1版のインタープリタ)→DependenciesのPython Toolsで、3.13版を選んだ場合
 - ▶ numpy, scipy, sympy, matplotlibなどのライブラリも使えます(インストールする必要あり)
- Python Tutor
 - https://pythontutor.com
 - ▶ Pythonを選んで、Write code inのメニューでPython3.11のインタープリタを選びます。
 - ▶ importでライブラリなどを呼べないので注意してください。