# Python で学ぶ WAV プログラミング

北山洋幸◉著

CUTT
カットシステム

■サンプルファイルのダウンロードについて

本書掲載のサンプルファイルは、一部を除いてインターネット上のダウンロードサービスからダウンロードすることができます。詳しい手順については、本書の巻末にある袋とじの内容をご覧ください。

なお、ダウンロードサービスのご利用にはユーザー登録と袋とじ内に記されている番号が必要です。そのため、本書を中古書店から購入されたり、他者から貸与、譲渡された場合にはサービスをご利用いただけないことがあります。あらかじめご承知おきください。

・本書の内容についてのご意見、ご質問は、お名前、ご連絡先を明記のうえ、小社出版部宛文書（郵送またはE-mail）でお送りください。
・電話によるお問い合わせはお受けできません。
・本書の解説範囲を越える内容のご質問や、本書の内容と無関係なご質問にはお答えできません。
・匿名のフリーメールアドレスからのお問い合わせには返信しかねます。

本書で取り上げられているシステム名／製品名は、一般に開発各社の登録商標／商品名です。本書では、™ および ® マークは明記していません。本書に掲載されている団体／商品に対して、その商標権を侵害する意図は一切ありません。本書で紹介しているURLや各サイトの内容は変更される場合があります。

# はじめに

本書はPythonで音響（WAVファイル）を扱う入門書です。プラットフォームや開発環境へ依存しないように心がけました。

Pythonとは、プログラミング言語の一種です。Pythonはプログラミングに不慣れな人であっても習得しやすい言語です。習得しやすいから非力かというと、そうではなく初級者なら初級者的な使い方が、そして上級者には上級者らしい使い方のできる万能なプログラミング言語といってよいでしょう。Pythonは、Windows、Linux、macOSなどマルチプラットフォームへ対応しています。このため、環境への依存度は低く、さまざまな環境で使用できます。プログラミングというと、多少敷居が高く理系の素養が必要と思う人もいるでしょうが、そのようなことはありません。Pythonはプログラミング未経験者にも優しい言語です。

すべてのプラットフォームの解説を行うと、説明が冗長になりページ数も多くなってしまうので、主にWindows上で解説することを選択します。それでもPythonを使用する場合、プラットフォームの違いを実感することは多くないので特に支障はないでしょう。いくつかUbuntu上での例も示します。

Pythonには音を扱う多数のモジュールが用意されています。このため、音源データを直接操作する必要は少なく、モジュールを利用して高度な処理を行うことができます。とはいえ、モジュールが機能を提供していないときや、モジュールの関数が自身の目的に合致しない場合は、自身で音源データを操作する必要があります。本書では、前者はもちろん、後者に対するプログラミングについても言及します。また、それほど高度なことは行っていませんが、Pythonで音響処理やWAVファイルを扱いたいと思う人には、十分な基礎情報を提供できたと考えています。基本を身に付ければ、Python、および多数のモジュールを使って高度な応用プログラムを開発することが期待できます。

本書の対象読者は、

- Python初級者
- 音響プログラミング初級者
- 手軽にWAVファイルを直接操作したい人

です。

微力ながら、本書が音響プログラミングのきっかけになることを期待します。

2020年5月　新型コロナ対応のため自粛中の自宅にて

北山 洋幸

## 謝辞

出版にあたり、お世話になった株式会社カットシステムの石塚勝敏氏に深く感謝いたします。

## 用語

用語の使用に関して説明を行います。

### ■引数

引数は、一般的にメソッドや関数の引数を指しますが、コマンドラインに与えるパラメタも引数と表現します。どちらを指すかは、文脈から何を指しているか判断してください。

### ■関数とメソッド

メソッドと表現すべきところで関数と記述している場合があります。

### ■フレームレートとサンプリング周波数

同じものを指しますが、関数によって使い分けているため、なるべく使用した関数に合わせるようにします。混在して使用しますが、同じものを指します。

### ■カタカナ語の長音表記

「メモリー」や「フォルダー」など、最近は語尾の「ー」を付けるのが一般的になっていますので、なるべく「ー」を付けるようにします。ただ、開発環境やドキュメントなどに従来の用語を使用している場合も多いため、本書では、語尾の「ー」は統一していません。

### ■「/」と「¥」

フォントの関係で、「/」と「¥」は混在します。どちらも同じものです。

### ■クラスとオブジェクト

クラスとオブジェクトはなるべく使い分けています。クラス、オブジェクト両方に適用できる内容については、クラス、オブジェクトを省いている場合もあります。

### ■クラスとインスタンス

クラスとインスタンスもなるべく使い分けています。クラス、インスタンス両方に適用できる内容については、クラス、インスタンスを省いている場合もあります。

## 開発環境

いくつかの環境でテストしましたが、動作チェックしたのは下記の構成です。

**オペレーティングシステム：Windows 10 Pro**

| | | |
|---|---|---|
| Anaconda | 2020.02 版 | |
| Python | 3.7.4 | |
| Spyder | 3.3.6 | |
| Numpy | 1.16.5 | （conda list で確認） |
| Scipy | 1.3.1 | （conda list で確認） |
| PyAudio | 0.2.11 | （conda list で確認） |
| soundfile | 0.9.0.post1 | （conda list で確認） |
| wavio | 0.0.4 | （conda list で確認） |

以下のシステムでもチェックしましたが、この環境では一部のプログラムしか確認していません。

**オペレーティングシステム：Windows 10 Pro**

| | | |
|---|---|---|
| Anaconda | 2020.02 版 | |
| Python | 3.7.6 | |
| Spyder | 4.0.1 | |
| Numpy | 1.18.1 | （conda list で確認） |
| Scipy | 1.4.1 | （conda list で確認） |
| PyAudio | 0.2.11 | （conda list で確認） |
| soundfile | 0.9.0.post1 | （conda list で確認） |
| wavio | 0.0.4 | （conda list で確認） |

**オペレーティングシステム：Ubuntu 18.04.3**

| | | |
|---|---|---|
| Anaconda | 2020.02 版 | |
| Python | 3.7.6 | |
| Spyder | 4.0.1 | |
| Numpy | 1.18.1 | （conda list で確認） |
| Scipy | 1.4.1 | （conda list で確認） |
| PyAudio | 0.2.11 | （conda list で確認） |
| soundfile | 0.10.3.post1 | （conda list で確認） |
| wavio | 0.0.4 | （conda list で確認） |

## URL

書籍中に記述されているURLは原稿執筆時点のものです。URLの変更やWebサイトの構造は頻繁に変更されますので、記述したURLが存在するとは限りません。もし、ページなどが見つからない場合は、トップページへ移動して探すか、インターネットでキーワードを検索してください。記述したURLに必ず紹介した内容が記載されていることを保証するものではありません。

# 目次

はじめに ..... iii

**第 1 章 開発環境とインストール ..... 1**

1.1 Anaconda を Windows にインストール ..... 2
1.2 Anaconda Navigator ..... 8
1.3 Anaconda Prompt ..... 13
1.4 Anaconda を Ubuntu へインストール ..... 15

**第 2 章 Spyder 入門 ..... 21**

2.1 Spyder 使用法の基礎 ..... 21
2.2 デバッグ機能 ..... 26
2.3 ダークモードを使わない ..... 29

**第 3 章 はじめてのプログラム ..... 33**

3.1 WAV ファイルを読み込む手法 ..... 33
3.2 matplotlib モジュールで波形を表示 ..... 38
3.3 soundfile のインストール ..... 39
3.4 PyAudio のインストール ..... 40

**第 4 章 WAV 情報表示と波形ダンプ ..... 43**

4.1 WAV ファイルの情報を表示 ..... 43
4.2 WAV ファイルをダンプ ..... 47
4.3 soundfile モジュールを使用したプログラム ..... 50

**第 5 章 チャンネルを操作 ..... 53**

5.1 WAV ファイルの左チャンネルの音を消す ..... 53
5.2 WAV ファイルの右チャンネルの音を消す ..... 56
5.3 Ubuntu で実行 ..... 57

**第 6 章　モノラル変換 ..... 59**

6.1　左チャンネルからモノラル WAV ファイルを生成 ..... 59
6.2　右チャンネルからモノラル WAV ファイルを生成 ..... 62
6.3　左右のチャンネルからモノラル WAV ファイルを生成 ..... 64

**第 7 章　サウンドの抜き出し・分割・連結 ..... 67**

7.1　サウンドファイルから一部を抜き出す ..... 67
7.2　サウンドファイルから任意の一部を抜き出す ..... 69
7.3　サウンドファイルから任意の一部を抜き出す・その 2 ..... 74
7.4　サウンドファイルを分割する ..... 77
7.5　サウンドファイルを任意の数に分割する ..... 79
7.6　サウンドファイルを任意の数に分割する・その 2 ..... 82
7.7　サウンドファイルを連結する ..... 85

**第 8 章　テキストから WAV ファイルを生成 ..... 89**

8.1　入力テキストファイルの形式 ..... 89

**第 9 章　いろいろな波形を生成 ..... 95**

9.1　方形波の生成 ..... 95
9.2　三角波の生成 ..... 101
9.3　ノコギリ波の生成 ..... 104
9.4　正弦波の生成 ..... 107

**第 10 章　ボリューム変更・バランス変更・カラオケ化 ..... 109**

10.1　ボリューム変更 ..... 109
10.2　バランス変更 ..... 113
10.2.1　サンプル WAV ファイル生成 ..... 113
10.2.2　バランスを左右に移動 ..... 114
10.3　カラオケ化 ..... 119

**第 11 章 ディレイやエコー ..... 123**

11.1 二重唱 ..... 124
11.2 リバーブとエコー ..... 127
11.2.1 リバーブ ..... 127
11.2.2 エコー ..... 132
11.2.3 ループで同じプログラムを ..... 133

**第 12 章 matplotlib 入門 ..... 135**

12.1 matplotlib で波形表示 ..... 135

**第 13 章 WAV ファイルをグラフ化 ..... 147**

13.1 簡単な表示 ..... 147
13.2 タイトルなど各種情報も表示 ..... 149
13.3 ステレオ／モノラルに対応 ..... 150
13.4 ステレオ／モノラルに対応・その２ ..... 152

**第 14 章 FFT と IFFT ..... 157**

14.1 FFT 概論 ..... 157
14.2 単純な FFT と IFFT ..... 161
14.3 ナイキスト周波数 ..... 163

**第 15 章 フィルタ ..... 167**

15.1 FFT でフィルタ ..... 167
15.2 単純な FIR フィルタ ..... 173
15.2.1 ハイパスフィルタ ..... 179
15.3 WAV ファイルへ適用 ..... 185
15.4 scipy でフィルタ ..... 188

**第 16 章 スペクトログラム ..... 195**

16.1 スペクトログラム ..... 195

第 17 章 ハイレゾ対応 ..... 201

17.1 wavio モジュール ..... 201

17.1.1 wavio モジュール概要とインストール ..... 201
17.1.2 情報表示（wavio モジュール） ..... 202
17.1.3 ダンプ（wavio モジュール） ..... 205
17.1.4 24 ビットファイルを生成（wavio モジュール） ..... 207
17.1.5 24 ビットファイルを読み書き（wavio モジュール） ..... 209
17.1.6 ボリューム変更（wavio モジュール） ..... 217

17.2 soundfile モジュール ..... 220

17.2.1 情報表示（soundfile モジュール） ..... 220
17.2.2 ダンプ（soundfile モジュール） ..... 222
17.2.3 ダンプ・その 2（soundfile モジュール） ..... 224
17.2.4 24 ビットファイルを読み書き（soundfile モジュール） ..... 224
17.2.5 ボリューム変更（soundfile モジュール） ..... 225

17.3 wave モジュール ..... 228

17.3.1 ダンプ（wave モジュール） ..... 228
17.3.2 ダンプ・その 2（wave モジュール） ..... 230
17.3.3 ダンプ・その 3（wave モジュール） ..... 231
17.3.4 24 ビットファイルを読み書き（wave モジュール） ..... 232
17.3.5 ボリューム変更（wave モジュール） ..... 234
17.3.6 ハイレゾ対応ローパスフィルタ ..... 238

第 18 章 ハイレゾ 32 ビット対応 ..... 243

18.1 ダンプ ..... 243
18.2 32 ビットファイルを読み書き ..... 245
18.3 ボリューム変更 ..... 246

第 19 章 サンプリング周波数変換 ..... 249

19.1 サンプリング周波数変換概論 ..... 249
19.2 単純なリサンプリング ..... 251
19.3 リサンプリング ..... 258

第 20 章 直接再生 ..... 267

20.1 加工せず直接再生 ..... 267
20.2 左右のチャンネルを交互にミュート ..... 270

20.3　バランス変更 ..... 272
20.4　カラオケ化 ..... 273
20.5　フィルタ ..... 274
20.6　非ブロッキング ..... 276

**付　録** ..... **281**

付録 A　wave モジュール ..... 281
付録 B　Soundfile モジュール ..... 284
付録 C　PyAudio モジュール ..... 290
付録 D　WAV ファイルを扱うための numpy 超入門 ..... 308

参考資料／参考文献 ..... 318

索　引 ..... 319

# 第1章

## 開発環境とインストール

本章では、環境の設定、簡単な開発環境の使用法、そして環境が正しいことを確かめるプログラムの作成について解説します。Python はマルチプラットフォームで使用可能です。Python で開発したプログラム自体はポータビリティがありますが、Python 対応のプログラムを開発・実行させる環境は、それぞれが自身でセットアップしなければなりません。

Python のインストールは簡単になりました。あるいはプラットフォームによっては、Python は最初からインストールされています。ただ、各種モジュールなどは別途インストールが必要なものもあります。開発環境のバージョンは、日々変化するためインストールの詳細を説明しても価値があるとは思えません。しかし、何も解説しないのも初心者には不親切です。そこで、本書では Windows に環境を構築する方法を中心に解説します。Ubuntu についても少し解説します。

なお、Python、各モジュール、オペレーティングシステムの組み合わせやバージョンの違いなどで、正常に環境をインストールできない、あるいは正常にプログラムが動作しないこともあります。そのような場合は、動作を確認できているバージョンに合わせる、あるいは公式サイトなどに掲載されている情報や、フォーラムなどを覗き解決法がないか調べるとよいでしょう。利用したモジュールには、比較的小規模でソースコードを簡単に入手できるものもありますので、正常に動作しない場合は自身で修正を行ってみるのもよいでしょう。筆者もいくつかのモジュールはドキュメントが不十分で理解が困難であったため、ソースコードを参照しました。基本的に、オープンソースで提供されているものは、自身で解決する努力は必要です。

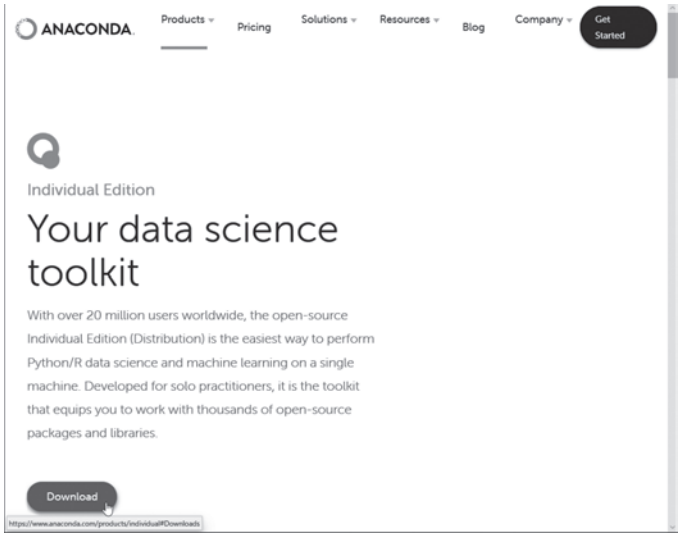
# 1.1 Anaconda を Windows にインストール

Python 本体のみを使うには、いろいろな環境が提供されているため、それほど難しくありません。例えば Python の本家（https://www.python.org/）から Python のみをインストールするだけです。ここでは、IDE やコンソールなどもパッケージされて広く使われている Anaconda を、Windows にインストールする手順について紹介します。

## Anaconda のダウンロード

Anaconda の公式ホームページからダウンロードします。まず、Anaconda のダウンロードページにアクセスします（https://www.anaconda.com/download/）。表示されたページの「Download」ボタンをクリックします。ブラウザの表示範囲が狭い場合は、「Download」ボタンが見えませんのでスクロールしてください。

**図 1.1 ● Anaconda のダウンロードページ**




「Anaconda Installers」が現れますので、「Windows - Python 3.7」の「64-Bit Graphical Installer」をクリックします。Python 2.7 系のための Anaconda も提供されていますので、2.x 系を使用したい人は、そちらを選択するとよいでしょう。本書は、Python 3 系をダウンロードします。

**図 1.2 ● Python 3 系を選択**

ブラウザの下部に、インストール方法のボタンが現れます。ここでは、直接インストールを行いたいので「実行」を選びます。

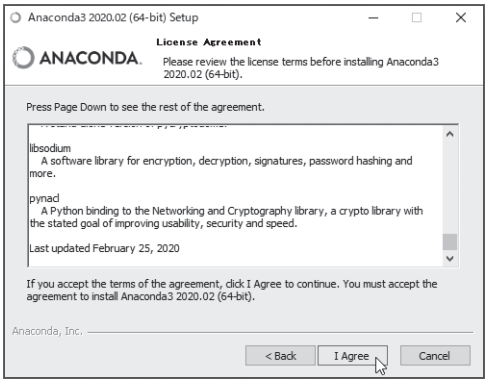
**図 1.3 ●「実行」を選ぶ**

しばらくすると、インストーラが起動します。Anaconda のビット数表示や、ほかのアプリケーションを閉じるように案内されますので、間違いがないことを確認したら「Next >」ボタンをクリックして次の画面に進みましょう。

**図 1.4 ●インストーラの起動画面**

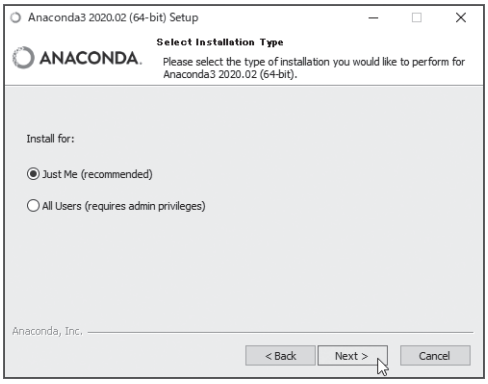
ライセンスの確認画面が表示されますので、最後までスクロールしながら、ライセンスを一通り確認し、「I Agree」ボタンをクリックして次の画面に進みます。

**図 1.5 ●ライセンスの確認画面**



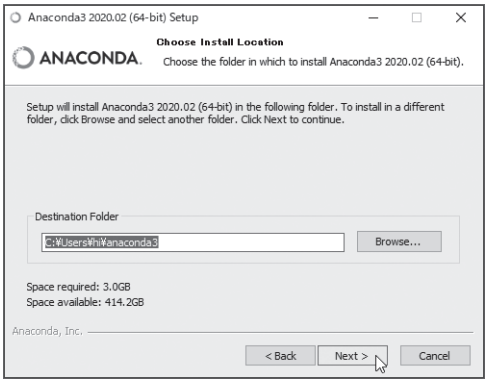

現在の単一ユーザーとしてインストールするか、全ユーザーへインストールするかを選択する画面が表示されます。単一ユーザーとして（Just Me）インストールするのが推奨されています。複数の人でパソコンを共用し、全員が同じバージョンの Anaconda を使用するような特殊な環境でない場合、「Just Me」を選択したまま「Next >」ボタンをクリックして次の画面に進みましょう。

**図 1.6 ●インストールの種類を選択**



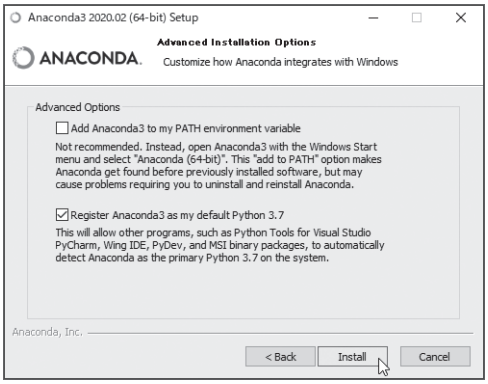
インストール先を選択する画面が表示されます。特別な理由がないかぎり変更する必要はないでしょう。インストール先を確認・選択して「Next >」ボタンをクリックして次の画面に進みましょう。「¥User」の配下にインストールしたくない場合は、「Browse...」ボタンをクリックし、適切な場所へインストールしましょう。

**図 1.7 ●インストール先を選択**



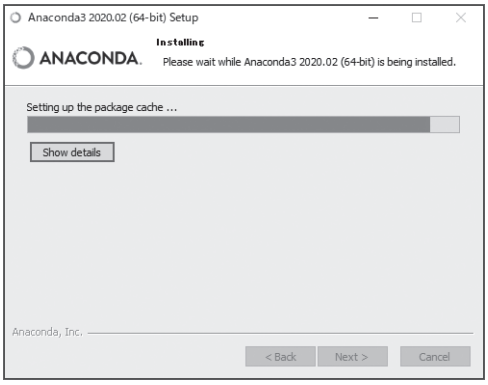

インストールのオプションを選択する画面が表示されます。特に変更の必要はありませんので、このまま「Install」ボタンをクリックしインストールを開始します。「Add Anaconda to my PATH environment variable」は、環境変数 PATH に Anaconda のフォルダを追加するか否かを決める選択肢です。最初の方に「Not recommended ...」と記述されており推奨されていません。これをチェックしなくても、コンソールを使用したいときは「Anaconda Prompt」を使用すると Python へのパスは通っています。また「Register Anaconda as my default Python 3.7」は「Anaconda をデフォルトの Python 3.7 として登録するか」の選択肢です。これにチェックを付けておくと、インストールした Python がシステム上のプライマリとして扱われますので、ほかの開発ツールは、この Python（Anaconda）を自動で認識します。詳細は表示されているメッセージを参照してください。

**図 1.8 ●インストールのオプションを選択**



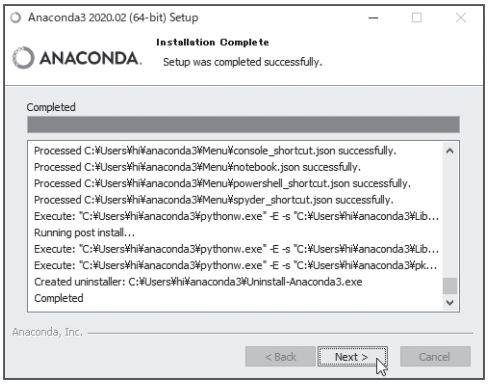

「Install」をクリックすると、しばらくインストール作業が続きますので、完了するのを待ちます。

**図 1.9 ●インストール中**



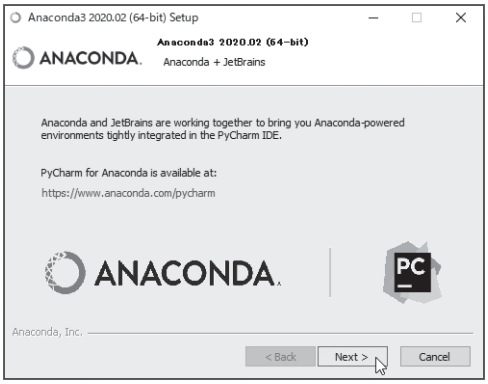

インストールが完了すると、図 1.10 のように「Completed」という表示に変わります。「Next >」をクリックして次の画面に進みます。示した画面は「Show details」ボタンをクリックした状態です。「Show details」ボタンをクリックしないと、インストール状況は表示されません。

**図 1.10 ●インストール完了**



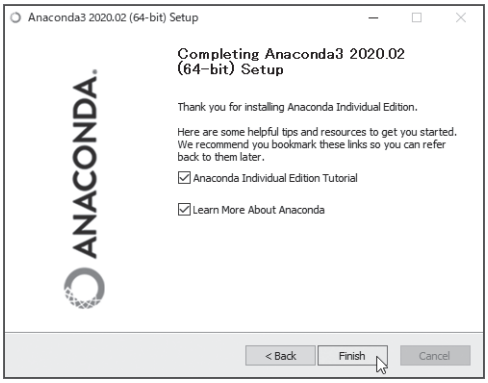

PyCharmに関する案内メッセージが現れます。「PyCharm for Anaconda」がavairableになっているサイトのURLも表示されます。今回は、PyCharmなどは使用しません。内容を読んで「Next >」をクリックして次の画面に進みます。

**図 1.11 ●インストーラを終了する**



インストール作業が終わったことを表す画面が表示されます。「Finish」ボタンをクリックし、インストーラを終了させます。チェックボックスにチェックされたままだと関連情報が表示されますが、必要なければチェックを外してください。いずれにしても Anaconda 3 はインストールされています。

**図 1.12 ●インストール完了**

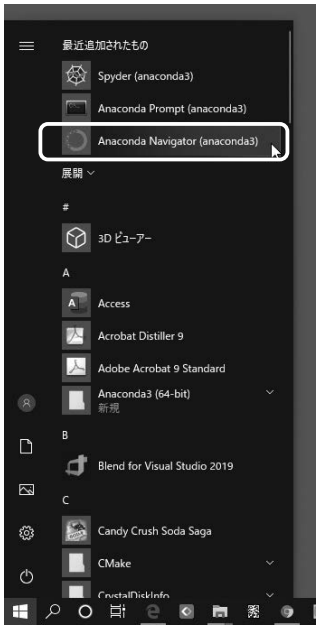



これで Anaconda のインストールは完了です。

## 1.2 Anaconda Navigator

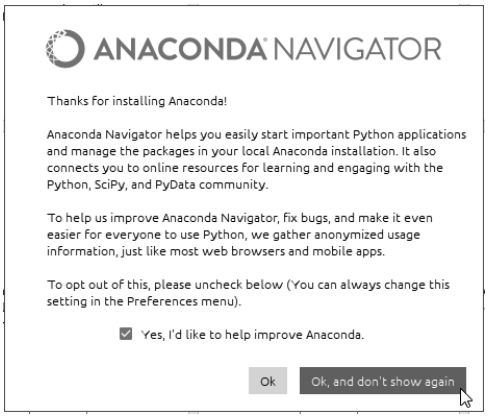
Anaconda のインストールが完了すると、スタートメニューに Anaconda のフォルダが追加されます。この中に「Anaconda Navigator」という項目があります。これは Anaconda を管理するためのツールです。まず、Anaconda Navigator を起動してみましょう。Anaconda Navigator は、Anaconda のフォルダだけでなく、スタートメニューの「最近追加されたもの」にも表示されますので、そちらから起動します。

**図 1.13 ● Anaconda Navigator の起動**




最初の実行で、Anaconda の改善に協力するか否かを求められます。ここでは、何も変更せず「OK, and don't show again」ボタンをクリックします。つまり、改善に協力し、この画面を今後表示させないようにします。メッセージにあるように、この設定はいつでも変更可能です。

**図 1.14 ● Anaconda Navigator の初回起動時の画面**

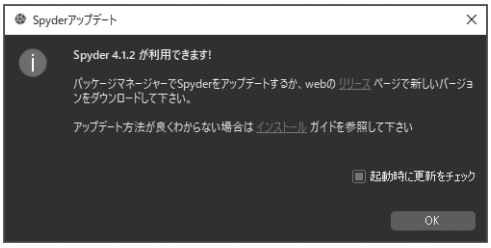


Anaconda Navigator が立ち上がると、図 1.15 に示す画面が現れます。左側に Environments、Learning、Community や Documentation などが存在します。これらはとても有益ですので、覗いてみるとよいでしょう。ここでは、右側に表示されている Spyder を起動します。「Launch」ボタンをクリックして起動します。

**図 1.15 ● Spyder の起動**

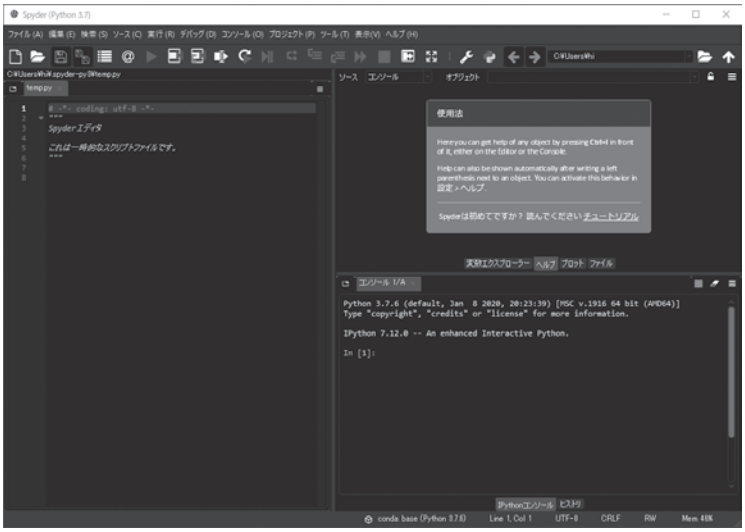
起動前に、ファイヤーウォールが警告を発する場合や、Spyder の最新バージョンが存在する案内が表示される場合があります。ファイヤーウォールの警告が出たら、通信を許可してください。Spyder の更新は行う必要はありませんので、最新バージョンが案内されても構わず「OK」ボタンをクリックします。もちろん、Spyder の最新バージョンを使用したいなら、案内メッセージに従って最新に更新してもよいでしょう。

**図 1.16 ● Spyder のアップデート案内**



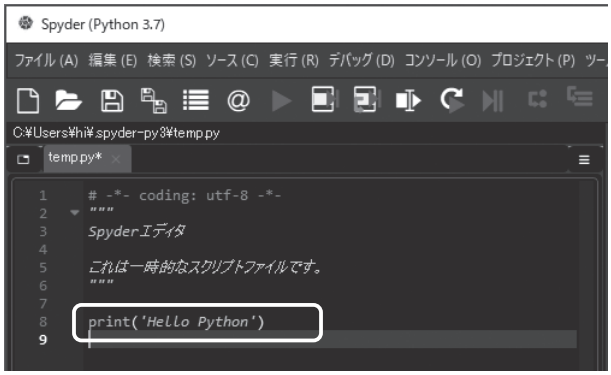
しばらくすると Spyder が起動します。

**図 1.17 ● Spyder の画面**



Python や Spyder のインストールに問題がないか確認したいので、「Hello Python」を表示する簡単なプログラムを作ってみましょう。

**図 1.18 ●「Hello Python」プログラムの作成**



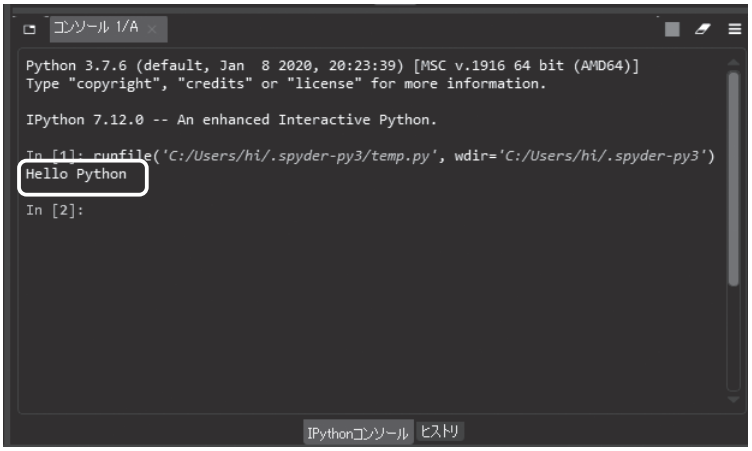

左側のエディタペインに print 文を入力します。実行するには ▶ をクリックします。すると、最初の実行で以降の画面が現れます。コンソールの選択、コマンドラインや作業ディレ

クトリなどを指定できます。ここでは何も変更せず「実行」ボタンをクリックします。

**図 1.19 ●実行設定の画面**

すると、右下の IPython コンソールに実行結果が表示されます。

**図 1.20 ●実行結果の表示**

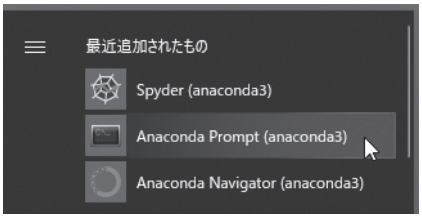


今回はSpyderをAnaconda Navigator経由で起動しましたが、Anaconda Navigatorを使用する必要がないときは、直接Spyderを起動するのもよいでしょう。

## 1.3 Anaconda Prompt

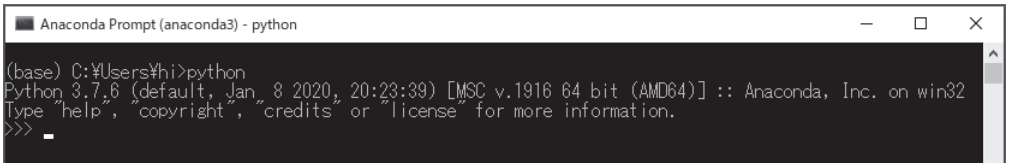
軽い作業や、既にプログラムの動作に大きな問題がない場合、手軽にPythonを利用したいときがあります。そのようなときは、コンソールベースで利用するのがよいでしょう。Anacondaには、そのような環境も含まれています。Anacondaのインストールが完了すると、スタートメニューにAnaconda3フォルダが追加されます。この中に「Anaconda Prompt (anaconda3)」というメニューが追加されます（表示は異なる場合があります）。これを起動しましょう。Anacondaのインストール直後であれば、スタートメニューの「最近追加されたもの」にも、「Anaconda Prompt (anaconda3)」が表示されますので、そちらをクリックしてもよいでしょう。

**図 1.21 ●スタートメニューの様子**




すると、Anaconda Promptが起動します。

**図 1.22 ● Anaconda Promptが起動**



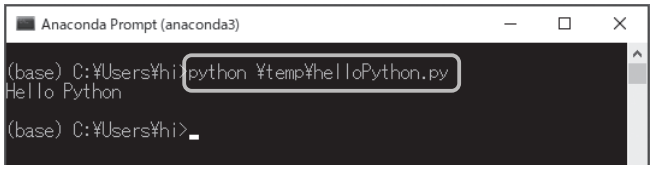

問題がないか「Hello Python」をインタラクティブに実行し、動作を確認します。問題なく

動作すれば、Python が使用できる状態にあることが分かります。

**図 1.23 ●動作確認**

インタラクティブではなく Python プログラムを記述したファイルを実行することもできます。一旦 Python を `quit()` などで抜けて、プロンプトから Python に続きプログラムファイルを指定します。

**図 1.24 ●プログラムファイルを指定**




問題ないことを確認できました。以降に、`hellpPython.py` ファイルの内容を示します。

```
# -*- coding: utf-8 -*-
print('Hello Python')
```

Anaconda のインストール時に Path を設定にチェックを入れておくと、通常のコマンドプロンプトから Python を使用できます。今回は、チェックを入れていませんので、Anaconda Prompt を使用します。

これで、Anaconda Prompt で Python を使う方法の解説を終了します。

# 1.4 Anaconda を Ubuntu へインストール

本書では、Windows を中心に解説しますが、Ubuntu へ Anaconda をインストールする方法も簡単に解説します。

Ubuntu には、一般的に Python は最初からインストールされています。その環境で構わないなら、そのまま使用することが可能です。初期の状態で使用する場合は、追加のモジュールや Spyder などは、自身で別途それらをインストールしなければなりません。もっとも Anaconda をインストールしたとしても、いくつかのモジュールは自身でインストールが必要です。

Anadonda をインストールする前の、Python の状態を示します。

```
$ python -V
Python 2.7.17
$ python3 -V
Python 3.6.8
$
```

使用した環境では、Python のバージョン 2.7.17 と 3.6.8 が使用できる状態でした。

## Anaconda のダウンロード

Anaconda の公式ホームページからダウンロードします。まず、Anaconda のダウンロードページにアクセスします（https://www.anaconda.com/download/）。表示されたページの「Download」ボタンをクリックします。