1. 序論: プログラムで何ができる

プログラミング・データサイエンス I

2022/4/14

1　序論

1.1　プログラムとは? 手続き的プログラムの場合

プログラムとは? 手続き的プログラムの場合

- コンピュータの処理を並べたもの
  - 書いた順番に実行
  - 条件分岐
    ある条件の時は A を、それ以外は B を実行
  - 繰り返し
    各○○に処理 A を行う

プログラムというと何をイメージしますか。呪文でもありませんし、秘密の暗号でもありません。

プログラムと呼ばれるものには、いくつかの種類があります。通常は、「手続き的プログラム」を差しています。手続き的プログラムとは、コンピュータに行わせる処理を順番に並べたものです。例えば、皆さんが学校の先生であるとしましょう。教室で出席をとる手順を考えましょう。名簿を持って、名前を呼び、返事を確認してチェックマークを付けます。返事が無い場合には、欠席者としてマークします。これを全員分繰り返す。最後にチェックマークの数を数える。これが手続きです。

「名前を呼ぶ→チェックマークを付ける」のように、予め定めた通りに実行する部分があります。「返事が無い場合→欠席マークを付ける」のように、条件に応じて違う動きをする部分があります。さらに、「全員分繰り返す」のように、繰り返しがあります。

これをプログラミング用の言語を使って書き、実行するのがプログラミングです。とくに難しくありません。

1.2 プログラム（コンピュータ）が得意なのは?

プログラム（コンピュータ）が得意なのは?

- ある処理を繰り返し（非常に多数回）繰り返す
- 大量のデータから、条件を満たすものを探す
- 毎日、決まった時刻にある処理を実行する
- ネットワークも使える

コンピュータが得意なことは何でしょう。

今度は、試験結果の集計のことを考えましょう。採点結果が表になっていて、ここから平均点を計算しましょう。クラスの人数が少ないうちは、手で計算しても、電卓を使ってもよいでしょう。しかし、すごく多い、例えば大学入試センター試験は 50 万人が受験しますが、その人数になるとコンピュータが必要です。コンピュータは、同じ処理を大量に繰り返すのが得意です。

大学入試センター試験の次に各大学の個別試験があります。各大学は、大学入試センターから、受験者の成績をもらいます。50 万人の中から、ある大学の志願者のデータだけを取り出す必要があります。これも、コンピュータが得意とする処理です。

コンビニエンスストアでは、各商品の売り上げを POS というシステムで集めています。決まった時刻に集計し、必要な商品を店舗に届けます。このように定時に何かの処理を実行することもコンピュータならば簡単です。さらに、この情報は、ネットワークで集計されていることはいうまでもありません。

1.3 データサイエンス

データサイエンス

- データ駆動型社会
  - データが利益を生む
  - データに基づく経営判断や政策立案
- データサイエンス
  - データを分析する
  - データから新たな知識を生み出す

現代はデータ駆動型社会と言われています。例えば、オンラインショッピングサイトは

単に商品をオンラインで売っているだけではありません。商品購入のデータから、サイト来訪者に商品を推薦することがありますね。多数の顧客の購入情報は、売れ筋商品の情報にもなります。

SNS (Social Networking Services) も、ユーザに情報交換の場所を提供しているだけではありません。ユーザが交わすメッセージを分析し、何が話題になっているかを抽出します。

GAFA と呼ばれる大規模情報企業は、オンラインショッピングサイトや SNS、その他のサービスを提供していますが、それと同時に大量のデータを収集・分析し、そこから利益を得ています。大量のデータは利益を生むのです。

また、企業経営や、行政における政策立案では、根拠、特にデータに基づく判断が重要となってきています。企業経営では、自社の製品やサービスの売上や素材の仕入れ状況、競合他社の状況、消費や景気の動向、自社の人員や施設設備の状況など、具体的な根拠に基づく判断が必要です。さらに、新たな商品やサービスのインパクト、自社の資源の再編成による改善など、データとモデルに基づくシミュレーションを行い、判断をしなければなりません。行政においても、対象となる地域の状況を適切に把握し、何が必要なのかを判断するとともに、施策の影響をシミュレーションしなければなりません。つまり、データが重要になっています。

データの収集と分析は、従来は「統計」という名で呼ばれてきました。しかし、今必要なのは、もっと広い切り口でのデータ分析です。例えば、大量のデータから特徴を取り出す際に、伝統的な数学的手法だけでなく、機械学習という方法を使うなどです。機械学習は、人工知能の一種であり、コンピュータによるデータ分析を行います。

このような新しい分野を、「データサイエンス」と呼びます。

1.4 様々なプログラミング言語

様々なプログラミング言語

- 高級言語
  - 英語のような単語で記述
  - スクリプト型言語
  - コンパイラ型言語
- 低級言語
  - 機械に近い言語：読みにくい

プログラミング言語には、「高級言語」と「低級言語」があります。べつに、価値が「高級」「低級」という意味ではありません。「低級言語」は、よりコンピュータという機械に近い言語です。従って、読み書きは難しいものです。一方、ほとんどのプログラマーは、「高級言語」を使っています。「高級言語」では、プログラムの全体構造や処理がわかりやすく記述することができます。世界中で使われるために、英語のような単語で記述するものがほとんどです。

また、プログラミング言語の別の分類方法があります。プログラムを書くとただちに実行できる「スクリプト型」と、実行前にコンピュータが高速に実行できるように翻訳する「コンパイラ型」です。

Python は「高級言語」であり、「スクリプト型」の言語です。

1.5 Python とは

Python とは

- 1989 年ごろから、Guido van Rossum が開発開始した言語
- 読みやすく、書きやすいと言われている
- 無償で利用できる
- データ処理や Web アプリなど、非常に広汎に利用されている

この講義で使う Python (パイソン) について説明しましょう。

Python は、無償、つまりタダで使うことができるプログラミング言語で、読みやすく、書きやすいと言われています。なぜ、「言われています」という言い方をするかというと、プログラムを書いてきた人々のなかには、「そうかなぁ」という声も少なくないからです。

しかし、これまでプログラムを書いてこなかった人々が使い始めているので、読みやすく、書きやすいのかもしれません。実際に、データ処理や Web アプリ開発など、広汎な分野で使われています。書籍も多数出版されています。

1.6　この講義の目標

この講義の目標

- 簡単な Python プログラムを書ける
  マニュアル、本、Web を見ながらでも可
- プログラムを書いて、何か処理を実行しようと思うことがある

この講義では、皆さんが、簡単な Python のプログラムを書けるようになるのが目標です。仕事で Python を書けるレベルを求めるわけではありませんから、教科書、マニュアル、そして Web 上の記事を見てかまいません。仕事でプログラムを書く人だって、教科書やマニュアルなどを参照します。

そして、この後の学習や就職後に、「これって、ちょっとプログラム書けば、サクッとできる」と考えることができるようになれば、大成功です。プログラムを書くのは理系の子、と考えず、自分もできると思ってほしい。実際に、情報系の技術者の中には、文系の学部出身者が少なくありません。

1.7　教科書と講義の進め方

教科書と講義の進め方

- 大重美幸「詳細! Python 3 入門ノート」（ソーテック、2017）
- 必ず、教科書を読んでくる
  次回は、2 章から 4 章の部分

プログラミングを学ぶということは、言語を習得することですから、自習が重要です。授業で、全てを教えることはできません。また、英語の学習と同じで、文法を学んだだけでは使えるようにはなりません。他の人の書いた良いプログラムを読むことと、自分でプログラムを書くことでしか、習得できません。

この講義では、大重さんの教科書を使います。教科書を入手し、必ず読んで来てください。もちろん、どんどん先に進んでくれても何も問題ありません。

2　環境構築

環境構築

- 拡張子を表示
- Python のインストール
- 講義で必要なライブラリのインストール
- テキストエディタの導入
- GitHub との連携

最初に行うことは、プログラミングの環境を整備することです。最初に全体の流れを説明しておきます。

コンピュータで扱うファイルには、拡張子がついているものが多数あります。拡張子の情報は、そのファイルを扱うアプリケーションを指定するととても重要なものです。拡張子を表示する設定から始めます。

次に、Python をインターネットからダウンロードしてインストールします。更に、講義で必要なライブラリをインストールします。

プログラムを書き、実行する環境も大切です。今回は、Visual Studio Code というテキストエディタを使います。

最後に、サンプルプログラムを取得する環境を作ります。

2.1　拡張子を表示する

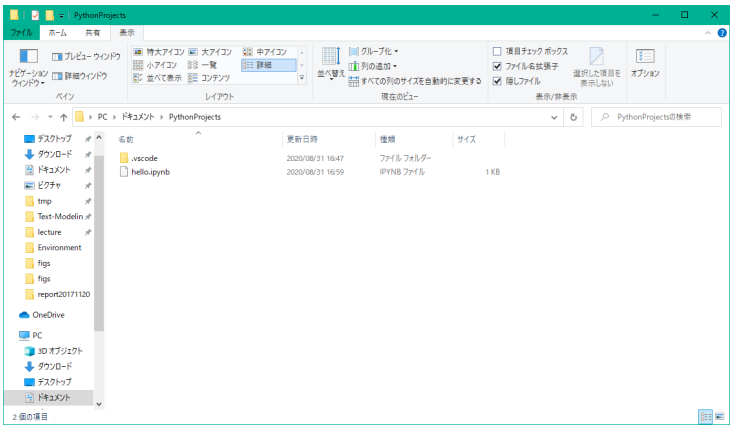
拡張子を表示する

- ファイルの拡張子は、アプリケーションとの対応を示す重要な情報
- 常に、拡張子を表示するように設定変更
- エクスプローラーで設定

あなたのコンピュータでは、ファイルの拡張子を表示していますか。拡張子は、ファイルの種類を表すと同時に、そのファイルを扱うアプリケーションとの対応付けをする大切なものです。エクスプローラーで拡張子を表示するように変更しましょう。既に変更している人は、この節を飛ばしてください。

エクスプローラーを開いて、「表示」メニューを開けましょう。メニューの右の方に、

「ファイル名拡張子」というチェックボックスがあります。これをチェックしておきましょう (図 2.1)。



2.2 Python インストール

Python インストール

- インターネットから Python をダウンロード
  - Miniforge3 を利用
- インストーラを使ってインストール

Python という言語に対して、それを実行する環境には、いくつかのものがあります。必要となるライブラリが最初からパッケージされたものがある一方で、最初は最小限だけをインストールし、あとから必要なものをインストールものもあります。前者は、様々なライブラリを一気にインストールすることができるというメリットがある一方で、そのために大きなディスク容量を必要とするという課題があります。今回の目的は、学習のための環境構築ですから、軽量なものを選択します。

今回は `Miniforge3` を利用します。以下の URL から自分の OS に併せてダウンロードします。Windows の場合には、自分だけにインストールすれば十分です。

`https://github.com/conda-forge/miniforge`

インストールが完了したら、`Miniforge Prompt` を起動し、以下のコマンドを実行します。これにより、作業用の環境 `myenv` を作成します。以降は、この作業用環境にライブラリを追加します。

```
conda create -n myenv python=3.9.7
```

2.3 ライブラリのインストール

ライブラリのインストール

- 講義で必要な Python ライブラリをインストール
- Miniforge Prompt を起動
  - conda activate myenv で作業環境に移動
  - conda install コマンドを利用

次に、当面の学習のために必要なライブラリを導入します。Miniforge Prompt を起動します。起動後、作業用環境に移動します。

```
conda activate myenv
```

当面の学習に必要なライブラリを以下に示します。

```
jupyter
xlrd
lxml
pandas
matplotlib
requests
openpyxl
html5lib
bs4
```

これらをインストールしてきます。例として jupyter をインストールする例です。複数パッケージを書くこともできます。

```
conda install jupyter
```

最後に、作図に日本語を使うことができるライブラリを導入します。残念ながら上記のコマンドで実行できません。以下のコマンドで導入します。

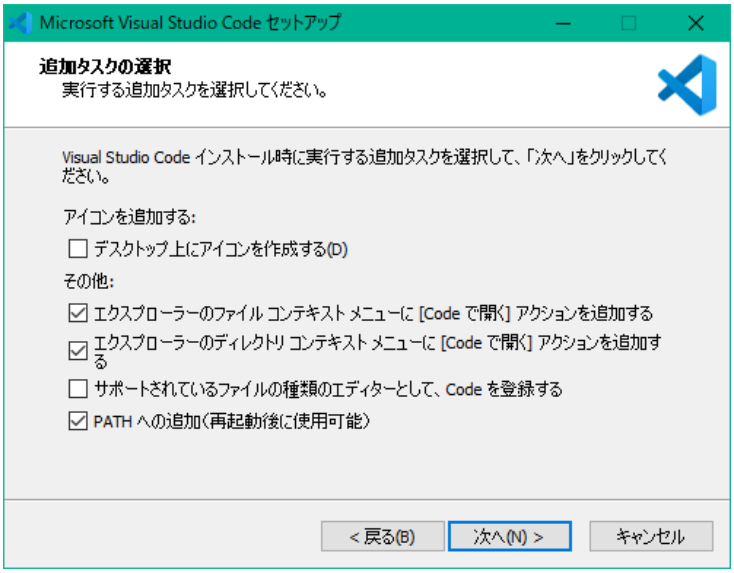
```
pip install japanize-matplotlib
```

2.4 VSCode のインストール

VSCode のインストール

- プログラムを書き、実行するための環境
- Visual Studio Code (VSCode) の導入
- インストール途中の「追加タスクの選択」に注意

次は、プログラムを書いて、実行するための環境を作りましょう。Microsoft が無償で提供する Visual Studio Code (VSCode) を使いましょう。このエディターは、様々なプログラミング言語に対応した拡張機能があり、通常のテキストの編集にも便利です。以下



の URL からダウンロードしてインストールしてください。

`https://azure.microsoft.com/ja-jp/products/visual-studio-code/`

インストールの途中で現れる「追加タスクの選択」で、

- エクスプローラーのファイルコンテキストメニューに [Code で開く] アクションを追加する
- エクスプローラーのディレクトリコンテキストメニューに [Code で開く] アクションを追加する

の二つにチェックをしてください (図 2.4)。

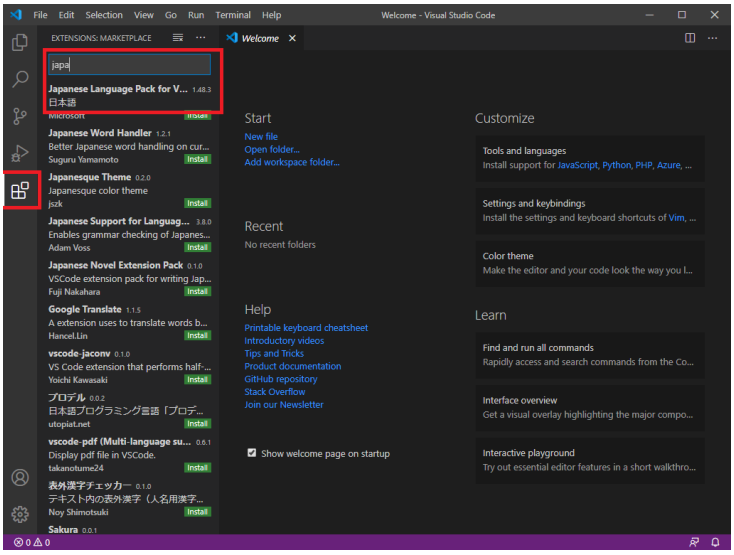
2.5 VSCode の日本語化

VSCode の日本語化

- VSCode の日本語化
- 拡張機能を利用

新しいバージョンの VSCode では、インストール直後に、日本語化パッケージのインストールが始まります。もしも、日本語化が自動で始まらない場合には、以下の手順で日本語化します。

VSCode を起動しましょう。図 2.5 の左に、小さな正方形 3 つと 1 つからなるアイコンがあります。これは、VSCode に機能を追加する際に使うアイコンです。このアイコンを押し、上のほうの検索窓に `japan` と打つと、名前に `japan` を含む拡張機能を表示します。その中から、`Japanese Language Pack` を選んでインストールします。一旦 VSCode を終了し、再度、起動すると日本語表示となります。



2.6 Python 用の拡張機能

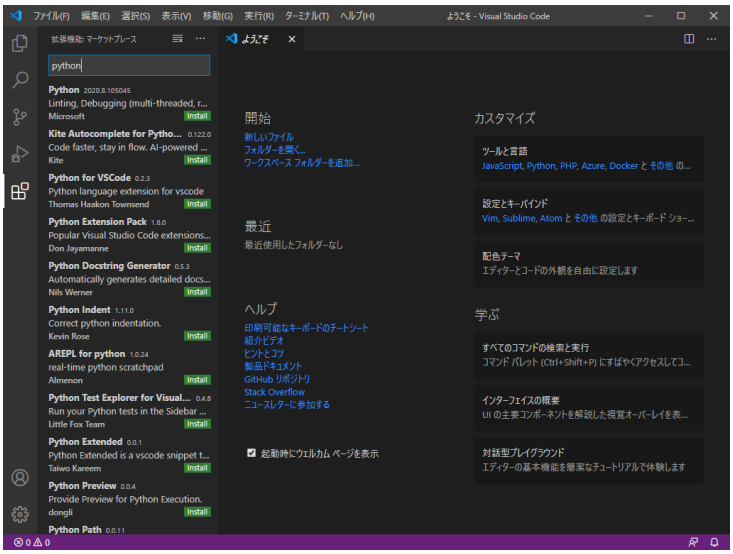
Python 用の拡張機能

- VSCode に Python 環境を導入
- 拡張機能を利用
- Python インタープリターとの対応付け

次に、Python の環境を VSCode に作りましょう。日本語化と同様に、機能を追加します。検索窓に `python` と入力し、Microsoft が提供する

`Python`

を選んでインストールします (図 2.6)。



また、デフォルトのインタープリターを設定しておきましょう。左下の歯車のマークを押すと、設定メニューが現れます。その中の「設定」を選び、「設定の検索」窓に `Python:Default` と打ちましょう。`Python:Derault Interpreter Path` に、`myenv` 環境へのパス

`~\\miniforge3\\envs\\myenv\\phtyon.exe`

を入力しておきます。

もう一つ、`Python Terminal` と打って検索すると、

python>Terminal:Activate Environment

という設定項目があります。これのチェックを外します。

もう一つ、Visual Studio IntelliCode という Microsoft が提供する拡張機能を入れておきましょう。コマンド等を途中まで入力すると、候補となるモノを提案してくれるものです。プログラム作成が効率的になります。

3 簡単な例を作ろう

簡単な例を作ろう

- 作業用フォルダの作成
- ファイルを作成
- ファイルの編集
- 実行

課題 1 では、適当な作業をフォルダを作って、環境が動作することを確かめよう。例えば、「ドキュメント」の下に PythonProjects というフォルダを作成しましょう。このフォルダにマウスを置き、右ボタンを押し、「Code で開く」を選びます。作成者を信頼するかを聞かれますから、「はい、作成者を信頼します」を選びます。

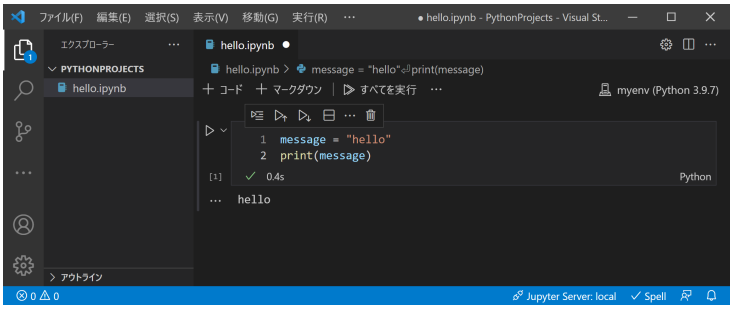
左の一番上にある文書のアイコンを押すとファイル操作を行うことができます。その右にエクスプローラーのメニューが表されます。文書にプラスのついたアイコンを押すとファイルを生成します。フォルダにプラスのついたアイコンは、フォルダ生成です。

それでは、hello.ipynb というファイルを作りましょう。図 1 のように、ソースコード 3.1 を入力します。入力している部分のことをセルと呼びます。セルの左にある”▷”を押すと、そのセルの部分を実行し、セルの下に結果を表示します。実行しようとすると実行するためのカーネルを聞かれることがあります。その際には、myenv を選んでください。図 1 では、結果として hello という文字列を表示しています。

ソースコード 3.1 簡単なプログラム

```
1 message = "hello"
2 print(message)
```

なお、右下にファイルを信用するかというメッセージが出て、実行できないことがあり



ます。その場合には、trust ボタンを押してください。

左上にあるファイル名が着いたタブに注目してください。ファイル名に白い○がついている場合には、ファイルが保存されていないことを表しています。コントロールキーを押しながら”S”を押すと、ファイルを保存することができます。

4 追加作業

VSCode の設定

- いろいろ設定できます
- 例えば、配色やフォントを設定してみよう

設定アイコンの中に、「配色テーマ」というメニューがあり、背景色と文字色の組合せを変えることができます。また、「設定」というメニューから、エディターとしての様々な設定、例えばフォントの大きさなどを変更することができます。設定は、

```
AppData\Roaming\Code\User\settings.json
```

に保存されています。

Github との連携

- 講義のサンプルプログラムを Github から配布
- Git をインストール

Github は、ソフトウェア開発のプラットフォームです。授業では、サンプルプログラムの配布に使用しました。Github からプログラムをダウンロードできるようにもしましょう。

最初に行うのは、Git というプログラムのインストールです。以下の URL から、OS

に合わせてダウンロードし、デフォルトのまま、インストールします。

`https://git-scm.com/downloads`

Github からのサンプルプログラムの取得は、次回行います。

5 次回

次回は、2 章から 4 章の部分を扱います。必ず読んでおいてください。もちろん、実際にプログラムを動かしてみても構いません。