コンピュータリテラシ発展 ～Pythonを学ぶ～

第5回：Excel作業を自動化しよう

情報学部 情報学科 情報メディア専攻
清水 哲也 (shimizu@info.shonan-it.ac.jp)

今回の授業内容

今回の授業内容

- 前回の課題解説
- 準備：Google Driveをマウントする
- 準備：PythonでExcelを操作する
- Excelの値を表示する
- 課題

前回の課題解説

前回の課題解説

- 前回の課題の解答例を示します
- 解答例について質問があればご連絡ください

解答例

（解答例を作成後URL貼り付け）

準備：Google Driveをマウントする

Colab上でExcelファイルを扱うためには？

ColabでExcelファイルを扱う方法は複数あります

1. プログラムでExcelファイルを読み込む方法
2. ローカルにあるExcelファイルを読み込む方法
3. Google Driveをマウントして読み込む方法
4. Colabに直接アップロードして読み込む方法

Colab上でExcelファイルを扱うためには？

一番簡単な方法

- **4.「Colabに直接アップロードして読み込む方法」**
- 注意：ランタイムが終了するとファイルが削除されます

この授業では

- **3.「Google Driveをマウントして読み込む方法」**
- 作業しているファイルが削除されずに済むのでこの方法を採用します
- 基本的に今後この形式でファイル読み込みを行うので覚えておいてください

Google Driveをマウントする手順

Google Driveをマウントする手順

Google Driveをマウントする手順

Google Driveをマウントする手順

Drive上に作業用フォルダを作成

- MyDriveの直下にフォルダを作成します
- 作成方法
  - Colabから作成する方法
  - Driveから作成する方法
- フォルダ名は**アルファベットのみ**にしてください
- 例：「Hatten」「CLHatten」など

Colabからフォルダを作成する方法

Colabからフォルダを作成する方法

Colabからフォルダを作成する方法

準備：PythonでExcelを操作する

OpenPyXLについて

- PythonでExcelの読み書きをするためのライブラリです
- Pythonを使って次のような操作ができます
  - セルの値をPythonで読み書きする
  - 複数のExcelファイル間でデータをコピーする
  - フォント設定を変更する
  - Excelの行高や列幅を調整する
  - グラフを追加する

OpenPyXLについて

- OpenPyXLはサードパーティライブラリなので環境によってはインストールが必要になります
- Google Colabの場合はすでにインストールされているのでインストール作業は不要です

```
!pip install openpyxl
```

OpenPyXLについて

- OpenPyXLはライブラリなのでPython上で使うためにはインポートする必要があります
- インポートはNotebookのはじめに1回だけ実行すればそれ以降はインポートされた状態になります
- コーディングするうえで「`openpyxl`」を何度も書くのは面倒なので名前を「`op`」に変更してインポートします

```
import openpyxl as op
```

Excelの値を表示する

Excelファイルの準備

- Moodleから「Shopping.xlsx」をダウンロードしてください
- MyDrive直下に作成した作業用フォルダにアップロードします
- アップロード方法
  - Colabからアップロードする方法
  - Driveからアップロードする方法

Excelファイルの中身を確認

ExcelとOpenPyXLのオブジェクト

- **Excel**ファイル = **Workbook**オブジェクト

ExcelとOpenPyXLのオブジェクト

- シート = **Worksheet**オブジェクト

ExcelとOpenPyXLのオブジェクト

- セル = **Cell**オブジェクト

指定したセルの値を取得する

- shopping.xlsx内のB1セルの値を取得して表示するプログラムを作成します
- 手順は以下の通りです
    i. `load_workbook()` メソッドでExcelファイルを読み込みます
    ii. ワークシートとセルを読み込みます
    iii. 読み込んだ値を表示します

指定してセルの値を取得する

- `load_workbook()`メソッドでExcelファイルを読み込みます
  - セルを取得するためにExcelファイルを取得する必要があります
  - `load_workbook()`メソッドの引数にファイル名を指定します
  - ファイル名だけではなくファイルまでのパスも必要です
  - Colabの場合：対象ファイルを右クリックして「Copy path」でOK

指定したセルの値を取得する

- ワークシートとセルを読み込みます
  - ワークブックを読み込んだのでワークシートとセルを指定します
  - 対象のシートは「**Sheet1**」，セルは「**B1**」です
  - さらに値を取得するので `value` プロパティを指定します

指定したセルの値を取得する

1. `load_workbook()`メソッドでExcelファイルを読み込みます
2. ワークシートとセルを読み込みます
3. 読み込んだ値を表示します

```
wb = op.load_workbook('/content/drive/MyDrive/???/shopping.xlsx')
value = wb['Sheet1']['B1'].value
print(value)
```

「**???**」は各自作成したフォルダ名を入れてください

複数のセルをまとめて取得する

- shopping.xlsxのB列をすべて取得してみます
- データが入っているセルがすでにわかっている場合
  - 対象：セルB1～B5
  - 繰り返し処理を利用します
  - セルの行と列を個別に指定します

複数のセルをまとめて取得する

- セルの行と列を個別に指定する方法
    - `row`(行)と`colum`(列)を使って個別に指定します

```
Workbookオブジェクト.Cell(row=行数番号, colum=列数番号)
```

複数のセルをまとめて取得する

- shopping.xlsxのB列をすべて取得してみます
- データが入っているセルがすでにわかっている場合
  - 対象：セルB1～B5
  - 繰り返し処理を利用します
  - セルの行と列を個別に指定します

```
wb = op.load_workbook('/content/drive/MyDrive/???/shopping.xlsx')
sheet = wb[‘Sheet1’]

for i in range(1,6):
  print(sheet.cell(row=i, column=2).value)
```

list()関数を使ってセルをまとめて取得する

- `range()` 関数
  - 最終行を事前に調べる必要があります
  - 最終行を指定してCellオブジェクトを取得します
- `list()` 関数
- 指定した列番号や行番号に存在するExcelの `Cell` オブジェクトをまとめて取得します

list()関数を使ってセルをまとめて取得する

- `list()` 関数の使い方

```
list(Worksheetオブジェクト.columns)[列番号]
```

注意：listを使う場合列番号はPython上の数え方になりますのでB列の場合は「1」を指定します

- `list()` 関数を使ってB列のCellオブジェクトをまとめて取得します

```
wb = op.load_workbook('/content/drive/MyDrive/???/shopping.xlsx')
sheet = wb[‘Sheet1’]

for cell in list(sheet.columns)[1]:
  print(cell.value)
```

課題

課題

- Moodleにある「SCfCL-5th-prac.ipynb」ファイルをダウンロードしてColabにアップロードしてください
- 課題が完了したら「File」>「Download」>「Download .ipynb」で「.ipynb」形式でダウンロードしてください
- ダウンロードした **.ipynbファイル** をMoodleに提出してください
- 提出期限は **5月23日(木) 20時まで** です