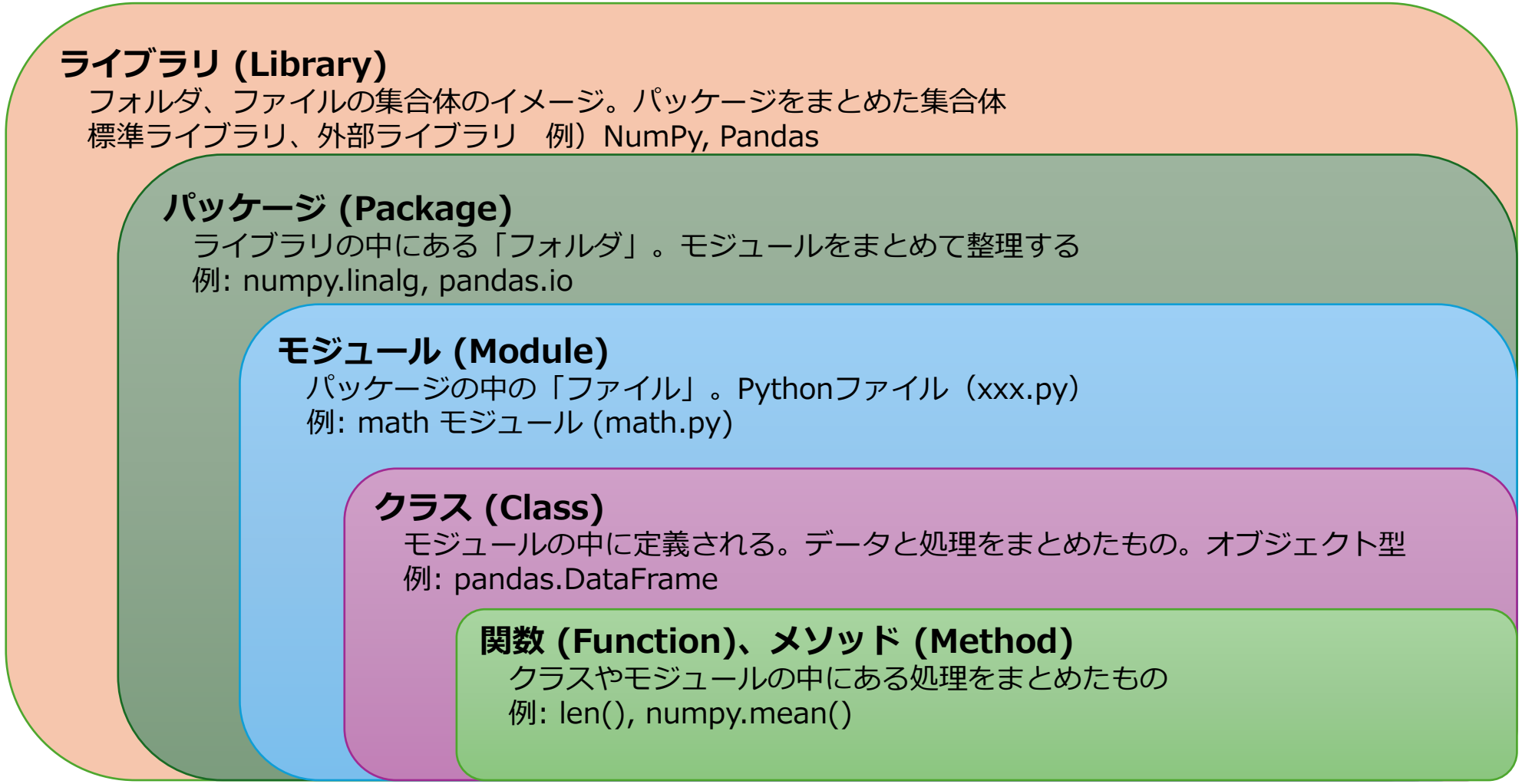
NumpyとPandas

ライブラリ、パッケージの関係

ライブラリ、パッケージ、モジュール、クラス、関数、メソッドの関係



- 関数 → 単独でも呼び出せる処理
- メソッド → クラスの中で定義された関数．クラスに属している関数

命名規則

対象	ルール (PEP8)	例
パッケージ	全小文字 (アンダースコア非推奨)	tqdm, requests ...
モジュール	全小文字 (アンダースコア可)	sys, os,...
クラス	最初大文字 + 大文字区切り	MyFavoriteClass
例外	最初大文字 + 大文字区切り	MyFuckingError
メソッド	全小文字 + アンダースコア区切り	my_favorite_method
関数	全小文字 + アンダースコア区切り	my_favorite_variable
変数	全小文字 + アンダースコア区切り	my_favorite_instance
定数	全大文字 + アンダースコア区切り	MY_FAVORITE_CONST
ファイル名	モジュール名+.py（アンダースコア可)	myhappymodule.py
内部変数	アンダースコアで開始	_my_happy_variable
内部メソッド	アンダースコアで開始	_my_happy_method

* 内部変数と内部メソッド：クラス内でのみ使用する変数とメソッドのこと
* PEP8: https://pep8-ja.readthedocs.io/ja/latest/

Numpy

Numpy（Numerical Python）

- 基本的には多次元配列の数値型データ
- 格納される複数要素はすべて同じデータ型やサイズで構成
- リスト型のように見えまるが、データ構造はndarray (np.ndarray)
- 各次元の位置は整数値のほかリストやスライスなど様々な形式で指定でき、任意の部分配列を選択できる。

1次元配列 [1,2,3,]

2次元配列 [[1,2,3],
[4,5,6]]

```
import numpy as np        # NumPyモジュールをインポートする

#ndarrayインスタンスを生成する
a = np.array([1, 2, 3])                    # 1次元配列
b = np.array([[1, 2, 3], [4, 5, 6]])   # 2次元配列
```

スライス

- リスト、文字列、タプル、バイト列などの一部分を切り取る仕組みを、スライスと呼ぶ
- スライスは、[開始:終了:ステップ]という書式で指定する
  - 開始 (start): 開始するインデックス。 0 で始まる．省略すると、シーケンスの先頭からになる
  - 終了 (stop): 終了するインデックス。終了インデックス自身は含まれない（未満の値）
  - ステップ (step): どのくらいの間隔で要素を取り出すか。省略すると、1つずつとなる

```
# インデックス2から5までの要素を取得
numbers = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

# 2, 3, 4 の3つが取れる (インデックス5の要素は含まれない)
print(numbers[2:5])    # 出力： [2, 3, 4]

# インデックス0から3までの要素を取得
print( numbers[0:4]）  # 出力： [0, 1, 2, 3]
```

スライス (省略記法)

- 開始と終了のインデックスは、省略することができる

- [:終了] 先頭から指定したインデックスの直前まで
- [開始:] 指定したインデックスから最後まで
- [:] 全ての要素 (コピーを作成する際によく使われる)

```
# 先頭からインデックス4の直前まで (0, 1, 2, 3)
print(numbers[:4]) # 出力: [0, 1, 2, 3]

# インデックス5から最後まで (5, 6, 7, 8, 9)
print(numbers[5:] ) # 出力: [5, 6, 7, 8, 9]

# リスト全体
print(numbers[:] ) # 出力: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

スライス (マイナスのインデックス)

- マイナスのインデックスが使用できる t
- 末尾から数えることを意味する

- -1: 最後の要素
- -2: 最後から2番目の要素

```
# 最後から3つの要素を取得
print(numbers[-3:] )　# 出力: [7, 8, 9]

# 最初の5つの要素を除いた残り (終了は未満)
print(numbers[5:-1] )　# 出力: [5, 6, 7, 8]
```

スライス (ステップ)

- ステップを指定することで、要素を飛び飛びで取り出すことが可能

```
# 2つおきに要素を取得
print(numbers[::2])       # 出力: [0, 2, 4, 6, 8]

# インデックス1から8まで、2つおきに要素を取得
print(numbers[1:9:2])     # 出力: [1, 3, 5, 7]
```

- 応用
  - ステップに -1 を指定すると、シーケンスを逆順にすることができる

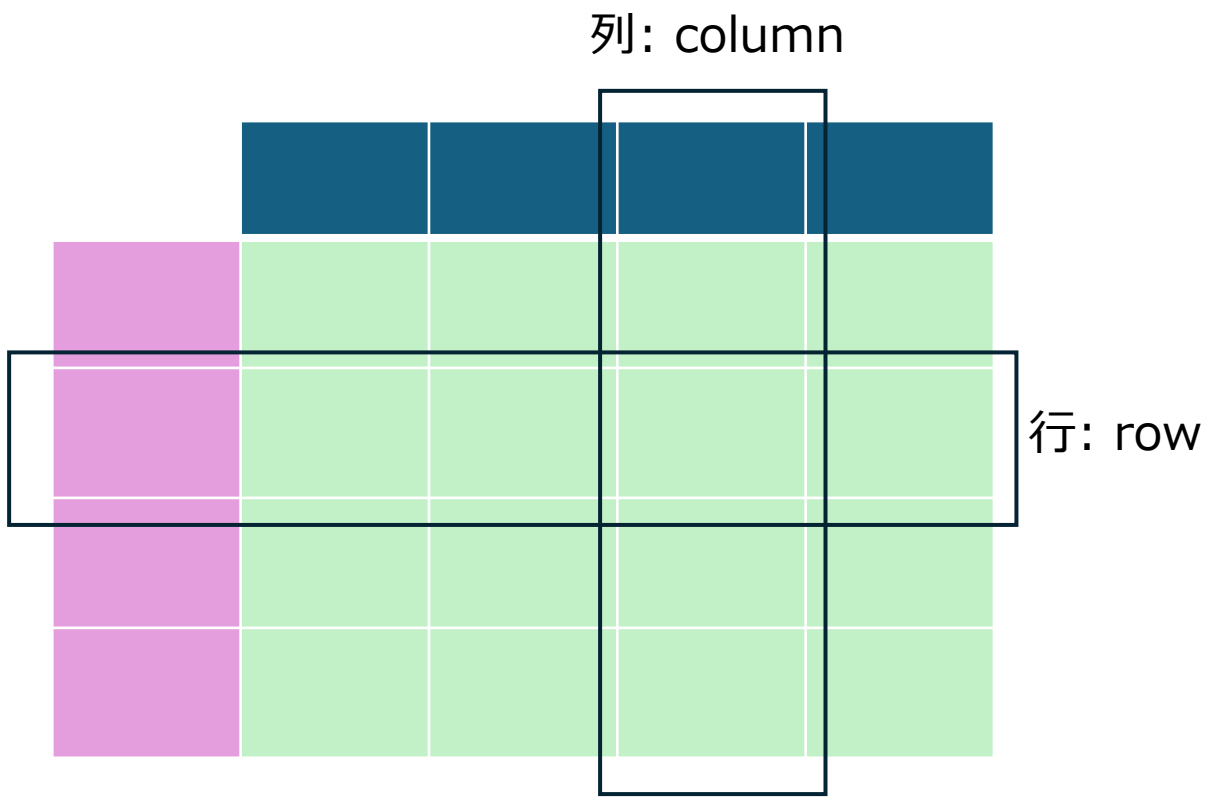
```
# リストを逆順にする
print(numbers[::-1]) # 出力: [9, 8, 7, 6, 5, 4, 3, 2, 1, 0]

# 文字列でも同様
text = "Python"
print(text[::-1] )  # 出力: "nohtyP"
```

Pandas

Pandas – DataFrame形式

- Excelのような二次元の表形式のデータ
- 列名一覧（columns）でデータを取り出したりできます。



	name	height	weight	memo
0	佐藤	170	60	あ
1	田中	160	50	い
2	鈴木	165	58	う
3	高橋	180	75	え

列名

値

インデックス

Pandas – DataFrameの作成

- pandas.DataFrameに2重リストを渡すとDataFrameオブジェクトを作成される

	0	1	2
0	佐藤	170	60
1	田中	160	50
2	鈴木	165	58

```
import pandas as pd

"""メインの処理"""
df = pd.DataFrame([['佐藤', 170, 60], ['田中', 160, 50], ['鈴木', 165, 58]])
```

Pandas – DataFrameの作成

- 列に名前を付ける

	name	height	weight
0	佐藤	170	60
1	田中	160	50
2	鈴木	165	58

```
import pandas as pd

df = pd.DataFrame([['佐藤', 170, 60], ['田中', 160, 50], ['鈴木', 165, 58]])
df.columns = ['name', 'height', 'weight']
```

Pandas – DataFrameの作成

- 列に名前を付ける
- 作成時に名前をつける

	name	height	weight
0	佐藤	170	60
1	田中	160	50
2	鈴木	165	58

```
import pandas as pd

df = pd.DataFrame([['佐藤', 170, 60], ['田中', 160, 50], ['鈴木', 165, 58]],
                  columns=['name', 'height', 'weight', ])
```

Pandas – Series形式

- 1次元のデータ

	0
0	佐藤
1	田中
2	鈴木
3	高橋

```
import pandas as pd

series = pd.Series(['佐藤', '田中', '鈴木', '田中'])
```

	0
name	佐藤
height	170
weight	60
memo	あ

```
import pandas as pd

series = pd. Series(['佐藤', 170, 60, 'あ'], index=['name', 'height', 'weight', 'memo'])
```

DataFrameやSeriesから値を取得・取得する

- “at” 、 “iat” 、 “loc” 、 “iloc”を利用してDataFrame および Series から値を取得・設定する
- プロパティのため () のような丸括弧ではなく、[] のような角括弧を利用する。
- [行(row)，列(columns)]で指定する。行番号・列番号は0で始まる
- スライス[start:stop:step]でも指定可能

プロパティ	位置の指定方法	取得できるデータ
at	- 行名（行ラベル） - 列名（列ラベル）	単一の値 例）df.at[“0", “height"]
lat	- 行番号 - 列番号	単一の値 例）df.iat[0, 0]
loc	- 行名（行ラベル） - 列名（列ラベル	単一の値や複数の値 （SeriesまたはDataFrame） 例）df.loc[: , “height"]
iloc	- 行番号 - 列番号	単一の値や複数の値 （SeriesまたはDataFrame） 例　df.iloc[0, :]