プログラミングA2
第4回

練習問題解答例

練習問題：html.py

HTMLの
…
タグと
…
タグで文字列を囲むデコレータを実装せよ.

[要件]

1. get_title関数
   - Monsterクラスのインスタンスを受け取り,
   - インスタンスからtitle属性を取得し,
   - そのtitle文字列を返す
2. concat_strs関数　　stringsをconcatenateするの意味
   - Monsterクラスのインスタンスがならぶリストを受け取り,
   - 各要素に対してget_title関数を用いて文字列を取得し,
   - それらを結合した文字列を返す
3. li_decorator関数
   - 受け取った関数を実行する前後に
, を付与するデコレータ
4. ul_decorator関数
   - 受け取った関数を実行する前後に
, を付与するデコレータ関数
5. 3のデコレータで1の関数をデコレートする
6. 4のデコレータで2の関数をデコレートする

解答例：html.py

デコレータ
デコレート対象
デコレート中
デコレート後

html.py：デコレート対象の関数

```
@li_decorator
def get_title(mon):
    return mon.title


@ul_decorator
def concat_strs(mon_lst):
    s = []
    for mon in mon_lst:
        s.append(get_title(mon))
    return "".join(s)
```

html.py：デコレータ関数

```
def li_decorator(func):
    def _func(t):
        s1 = "<li>"
        s2 = func(t)
        s3 = "</li>"
        return s1+s2+s3+"¥n"
    return _func


def ul_decorator(func):
    def _func(t):
        s1 = "<ul>"
        s2 = func(t)
        s3 = "</ul>"
        return s1+s2+s3+"¥n"
    return _func
```

実行例：html.py

html.py

```
if __name__ == "__main__":
    monsters = [
        Monster("イーブイ"),
        Monster("シャワーズ"),
        Monster("サンダース"),
        Monster("ブースター"),
        Monster("エーフィ"),
        Monster("ブラッキー"),
        Monster("リーフィア"),
        Monster("グレイシア"),
        Monster("ニンフィア"),
    ]


print(concat_strs(monsters))
```

実行例

```
<ul><li>イーブイ</li>
<li>シャワーズ</li>
<li>サンダース</li>
<li>ブースター</li>
<li>エーフィ</li>
<li>ブラッキー</li>
<li>リーフィア</li>
<li>グレイシア</li>
<li>ニンフィア</li>
</ul>
```

デコレートしなかった場合も確認してみよう.

練習問題：zukan.py

Zukanクラスをイテレータとして実装せよ

[要件]

- `__init__`(ファイルパス)メソッドを実装する
  - ファイルパスを引数として静的メソッドを呼び出し，ポケモン名文字列のリストを受け取り，インスタンス変数に設定する【済】
  - どの要素まで取り出したかを記憶するインデックス`idx`をインスタンス変数として定義し，`0`で初期化する【未】
- `__iter__`()メソッドを実装する
  - Zukanクラスのインスタンス（＝イテレータ）を返す【未】
- `__next__`()メソッドを実装する
  - インデックスがリストの長さと等しかったら，`StopIteration`例外を`raise`する【済】
  - 該当するリストの**要素**を抽出する【未】
  - インデックスをインクリメントしたあと，**要素**を返す【未】

解答例：zukan.py

zukan.py

```
class Zukan:
    def __init__(self, file_path):
        self.titles = __class__.read_file(file_path)
        self.idx = 0

    def __iter__(self):
        return self

    def __next__(self):
        if self.idx == len(self.titles):
            raise StopIteration()
        title = self.titles[self.idx]
        self.idx += 1
        return title

    @staticmethod 省略
```

← 例外の送出

実行例：zukan.py

zukan.py

```
if __name__ == "__main__":
    zukan = Zukan(sys.argv[1])
    print(next(zukan))
    print(next(zukan))

    for i, z in enumerate(zukan, 1):
        print(f"{i:03d}¥t{z}")
        if i == 5:
            break


    for i, z in enumerate(zukan, 1):
        print(f"{i:03d}¥t{z}")
        if i == 5:
            break
```

← イテレータの生成

実行例

```
フシギダネ
フシギソウ


001     フシギバナ
002     ヒトカゲ
003     リザード
004     リザードン
005     ゼニガメ

001     カメール
002     カメックス
003     キャタピー
004     トランセル
005     バタフリー
```

zukan自体がイテレータであるため，
for-in文の中で自動的に呼ばれるiter()関数により
自分自身が返され，新たなイテレータが生成されない

練習問題：name_generator.py

ポケモンの名前のリストと部分文字列を受け取り，リストから部分文字列を含む名前を1つずつ返すジェネレータ関数を定義せよ．

name_generator.py

```
def name_generator(lst, char):
    for title in lst:
        if char in title:
            yield title

titles = read_names(sys.argv[1])
char = sys.argv[2]
name_generator = name_generator(titles, char)
for res in name_generator:
    print(res)
```

実行例

```
python lec04/name_generator.py lec04/data/poke_names.txt リー
バタフリー
ニドリーナ
：
メリープ
ハリーセン
```