

付 録

記述の手間を減らし、
コードを読みやすくする

Tips



ここでは付録として、コードの記述の手間を減らし、コードを読みやすくするための方法を3つ紹介します。本書の作例でプログラミングの練習をするなどして、徐々にPythonに慣れてきたら、ここで紹介する方法で書き換えられそうな箇所を探し、実際に書き換えてみるとよいでしょう。

付録1 モジュール名の記述を省き、関数名だけ記述する方法

モジュールの中には、いくつかの関数が用意されているケースが多いです。そのため、関数のコードを記述する際は「モジュール名.関数名」といった具合に「モジュール名.」から記述するのが基本です。この「モジュール名.」の部分の記述を省き、関数名だけ記述すれば済むようにする方法を紹介します。

import文にキーワード「from」を一緒に使うと、指定したモジュールから指定した関数だけを読み込むことができます。すると、コードには関数名だけ記述すれば済むようになります。書式は次のとおりです。

書 式

```
from モジュール名 import 関数名
```

たとえば、作例2の**11.2節**（315ページ）で使ったbs4モジュール（BeautifulSoupモジュール）からBeautiful

Soup 関数だけを読み込むには次のように記述します。

コード

```
from bs4 import BeautifulSoup
```

これで、「bs4.BeautifulSoup」のうち「bs4.」を省き、「BeautifulSoup」と記述するだけで済みます。

さらに、**12.4節**で紹介したasによる簡略化を、関数名の記述で使えば、さらに記述の手間を減らすことになるでしょう。

ただし、fromの多用には注意が必要です。同じ名前の関数／メソッドでも、異なるモジュールのものであり、機能が異なる場合があります。そのような関数やメソッドをfromで省略してしまうと、どのモジュールのものかわからなくなり、コードの可読性は逆に下がってしまいます。

付録2 オリジナルの関数を定義して使うには

本書では、これまでprintなどの各種関数を利用してきました。Pythonでは、組み込み関数やモジュールの関数に加え、プログラマーが自分でオリジナルの関数を定義して使うこともできます。そのような関数は専門用語で「ユーザー定義関数」と呼ばれます。

ユーザー定義関数を利用すると、重複する処理のコードをまとめてくり出すことで、コード全体の記述量を減らせるとともに、メンテナンス性も向上できます。

ユーザー定義関数を定義するコードの書式は次のとおりです。

書 式

```
def 関数名(引数1, 引数2, ……):  
    処理  
    return 戻り値
```

定義にはdef文を用います。関数名に続けて、カッコと「:」を記述します。カッコ内には、引数を指定できます。引数が複数の場合は、「,」で区切って並べます。関数の中身である処理のコード、および戻り値を返すreturn文のコードは、インデントしてブロックとして記述します。

たとえば、引数に指定した価格から、消費税（8%）込みの価格を返すユーザー定義関数なら、次のように記述します。ここでは関数名を「get_ctax」、引数名を「price」とします。

コード

```
def get_ctax(price):  
    return price * 1.08
```

ユーザー定義関数を呼び出して実行する方法は、組み込み関数などと同じです。「関数名(引数)」の形式で記述すれば、その関数に指定した引数を渡して実行できます。たとえば、例にあげたget_ctax関数を使い、1000円の商品

の消費税込み価格を求めるには、次のように記述します。

コード

```
get_ctax(1000)
```

ユーザー定義関数はPythonのみならず、他の言語にも共通する仕組みです。

付録3 文字によって要素を わかりやすく取り出せる「辞書」

9章で紹介したリストは、個々の要素を取り出す際にインデックス番号を使いました。Pythonにはインデックス番号以外に、指定した文字列によって要素を取り出すリストも用意されています。そのようなリストは「辞書」（または「ディクショナリ」）と呼ばれます。

「辞書」を作成するコードの基本的な書式は以下です。

書 式

```
辞書名 = {キー1:値1, キー2:値2, ……}
```

要素を取り出すための文字列は「キー」と呼ばれます。キーとペアになる値を「:」で結んだ記述が1つの要素となります。要素は「,」で並べて記述します。

たとえば、次のような「辞書」を作成したい場合で説明します。

名前	キー	値
cstmr	氏名	立山秀利
	性別	男性
	年齢	48

そのコードは以下になります。キーは文字列なので「」で囲います。

コード

```
cstmr = {'氏名': '立山秀利', '性別': '男性', '年齢': 48}
```

この辞書「cstmr」から、たとえばキー「氏名」の要素の値を取得するには、次のように記述します。

コード

```
cstmr['氏名']
```

「辞書」はリストに比べ、文字列であるキーを使うことから、目的の要素をよりわかりやすく取り出せるのがメリットです。実際、Webの申し込みフォームなどでは、このような「辞書」の形式でデータを扱うケースが多々あります。

なお、「辞書」はPythonのみならず、他の言語にも共通する仕組みです。