ステップ30

Python［基礎編］

ワークブック

CUTT
カットシステム

はじめに

本書はPythonの基礎を習得することが目的の書籍です。

また、すでに他のPython入門書籍・サイト・ビデオなどで学習していて、それら学習内容の定着具合を確かめたい方にも活用いただける教材です。自身の習熟度合い・理解度を客観的に見ることは難しく、プログラムを書いたり読んだり、レビューをもらったり、それらを繰り返して自分に足りない部分を学んでいくのが一般的ですが、本書は演習問題が付属したワークブック形式となっており、習熟度・理解度の確認が容易になっています。

最近では機械学習が注目されたこともあり、Python入門書籍で簡単な画像認識などを扱うことも多いのですが、本書ではPythonの基礎的な内容を中心に学びます。どのような分野に進んでも本書で学んだ内容が土台となるはずです。

■ 読者対象

プログラミングをはじめて学ぶ工業高校生を対象読者に想定しています。工業高校生に限らず、はじめてプログラミングを学ぼうとする大学生や社会人の皆さんにも役に立つと思います。

■ 本書の構成

本書では次の構成をとります。

- 要点として、記憶してほしい語彙や概念を示します。
- ワーク［基礎］として、簡単な問題を出題します。
- ワーク［応用］として、やや高度な問題を出題します。

ワークの中には、答えが複数出てくる問題もあります。学校などの多くの方が同時に学習する環境ならば、ぜひ皆さんに様々な解答を出していただき、他人の発想を学んだり、その解答が適切かを議論していただければと思います。

目次

※ワークの解答は、以下の Web ページからダウンロードできます。

http://----------

Step 01

Pythonとは

学習のはじめに、Pythonというプログラミング言語について説明します。Pythonとは何か？　Pythonの何がいいのか？　そのような質問をされても答えられるようにしましょう。

要点

1.1　Pythonの歴史

1989年の12月、オランダ人のグイド・ヴァンロッサム（Guido van Rossum）はクリスマス休暇の暇つぶしとしてプログラミング言語の開発を始めました。これがPythonです。その後1991年2月にalt.sourcesニュースグループ上でバージョン0.9.0が一般公開され、今ではバージョン3も後半です。

4大P言語（Perl, Python, PHP, Ruby）の中では2番目に古く、意外に感じるかもしれませんがJavaやC#よりも年上となります。1957年のFORTRAN（フォートラン）から始まるプログラミング言語の歴史から見ると90年前後というのは最近に感じますが、最近よく使われているメジャーな言語の中ではそれなりに古く、歴史のある言語といえます。

余談ですが、「Python」という名前の由来はイギリスのコメディ番組「空飛ぶモンティ・パイソン」から取っています。

1.2　Pythonの特徴

Pythonの特徴は多くありますが、代表的なものは可読性・生産性・汎用性の3つです。

■ 可読性

可読性とはプログラムの読みやすさのことです。Pythonは「実行可能な疑似コード」と表現されるほど、自然言語に近い言語です。他言語では何かと出てくる丸括弧などの記号も少なくなるように設計されています。コードが読みやすいとそれだけ学びやすく、覚えやすく、上達が早くなりますし、他人との共同作業もはかどります。

■ 生産性

ある時間内や行数で、どれだけのプログラムや処理が作れるかを生産性といいます。Pythonはスクリプト言語と呼ばれるグループの仲間です。スクリプト言語の多くはプログラムの実行が簡単で、少ないコードでたくさんのことができます。CやJavaといった言語に比べるとコードの量が半分以下になることも珍しくありません。

■ 汎用性

Pythonは幅広い用途に使える言語なため、汎用的な言語といわれます。Webアプリケーショ

ンや科学分野をはじめ、多くの分野で実際に利用されています。汎用目的に作られた言語は数多くありますが、ある分野での開発が現実的ではない言語もあります。例えば、Web が主戦場の PHP でスマホアプリの開発は（少なくとも今は）現実的ではないでしょう。Python にも不得意なこと、または他言語の方が得意な分野はもちろんあるのですが、総合的にはとても汎用的な言語です。

プログラミング言語に最も必要なものは、コミュニティです。どんなに素晴らしい言語であっても、利用者が少なければ今後の発展は見込めず、情報の取得に苦労するでしょう。Python はどうでしょうか？　ご安心ください、とてもホットな言語です。ある言語がどれだけ人気があるかの指標として、以下の 3 サイトはよく参考にされます。どのサイトでも Python は高い位置に存在することが分かります。

http://pypl.github.io/PYPL.html
https://www.tiobe.com/tiobe-index/
https://spectrum.ieee.org/static/interactive-the-top-programming-languages-2017

ワーク

基礎

次のうち、Python の説明として正しいものを 1 つ選びましょう。分からない語句は、Google 検索を活用しても構いません。

1. Python は生産性が低く、読みづらい言語だ。
2. Python はオブジェクト指向をサポートしていない。
3. Python は可読性、生産性、汎用性に優れた静的言語だ。
4. Python は可読性、生産性、汎用性に優れたスクリプト言語だ。

Step 02

四則演算

Pythonに付属している対話モードはちょっとした電卓にも使えます。対話モードで四則演算 + α を試してみましょう。安物の電卓より、Python での数値計算のほうが便利です。Python のインストールは付録 A を、対話モードの使い方は付録 B を見ましょう。

要点

2.1 Python での四則演算 + α

足し算、引き算、掛け算、割り算のことを四則演算といいます（すでに知ってますよね？）。Python では四則演算はもちろん、他にも様々な計算ができます。実行結果 2.1.1 で確認しましょう。

実行結果 2.1.1　対話モードでの実行例

```
>>> 1 + 2
3
>>> 3 - 2
1
>>> 5 * 2
10
>>> 5 ** 2
25
>>> 4 / 2
2.0
>>> 4 // 2
2
>>> 10 % 3
1
```

掛け算が×ではなく *（アスタリスク）に、割り算が÷ではなく /（スラッシュ）になります。/ による割り算は小数点が表示されますが、小数点部分を切り捨てたいならば // とします。** で指数の計算ができ、%（パーセント）記号で剰余（割った余り）を求めることもできます。

2.2 ZeroDivisionError

数学では、0 で割ることは許されない行為です。Python では 0 で割ったときに実行結果 2.2.1 のように表示されます。

実行結果 2.2.1

```
>>> 1 / 0
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: division by zero
```

これは例外と呼ばれるもので、Python で実行中に検出されたエラーのことです。この例外を扱う方法は後に説明しますが、今はエラーがこのように表示されることを覚えてください。

2.3 演算の優先順位

演算には優先順位があり、それは数学での優先順位と同じです。

実行結果 2.3.1

```
>>> 10 + 20 - 10 / 5 * 100 ** 0
28.0
```

実行結果 2.3.1 では、まず `100**0` の指数が計算され、掛け算・割り算の後に足し算・引き算が計算されます。通常の数学における優先順位と同じです。

実行結果 2.4.1

```
>>> 10 + (20 - 10) / 5 * 100 ** 0
12.0
```

実行結果 2.4.1 を見てください。丸括弧をつけてみました。丸括弧内の演算が優先されるのも同様です。

ワーク

基礎①

次のうち、ZeroDivisionError となるものを 1 つ選びましょう。

1.

```
>>> 5 - 0
```

2.

```
>>> 5 * 0
```

3.

```
>>> 5 / 0
```

4.

```
>>> 6 ** 0
```

基礎②

「+」「-」「*」「/」記号を 1 つずつ使い、括弧の中を埋めましょう。

```
>>> 10 (    ) 5
15
>>> -10 (    ) 10
-20
>>> 10 (    ) 3 (    ) 5
6.0
```

応用

「%」「//」「**」記号を 1 つずつ使い、括弧の中を埋めましょう。

```
>>> 10 (  ) 3
3
>>> 10 (  ) 3
1
>>> 5 (  ) 2
25
```

Step 03

変数

変数を使うことで、計算結果や各種データを保存することができます。Pythonに限らずプログラミングにおいては重要な概念となるので、しっかり覚えましょう。

要点

3.1　変数の使い方

実行結果 3.1.1 は、price という変数に整数の 100 を保存し、その変数を使うサンプルコードです。

実行結果 3.1.1

```
>>> price = 100
>>> price
100
>>> price + 10
110
```

変数は、計算結果やデータを保存するのに使われます。price = 100 のように左辺に名前を、右辺に値や式を書きます。これを「変数 price に 100 を代入する」と表現します。代入が終われば、次からは price という名前で自動的に 100 の値が使われます。2 行目の price は 100 と表示され、4 行目の「price + 10」は「100 + 10」となり 110 と表示されます。= は数学では等しいという意味ですが、Python では代入を意味する記号ですので注意してください。また、他の言語でよくある変数名だけを宣言する機能はありません。

3.2　変数名に使える名前

変数名は自由につけることができますが、いくつかのルールがあります。

- 変数名の先頭には _（アンダースコア）、もしくはアルファベットしか使えない
- 変数名の 2 番目以降には、数字、アルファベット、アンダースコアが使える
- 予約語は使えない

予約語とは、Python プログラムの構文として使われているキーワードです。予約語を確認するには、実行結果 3.2.1 のように入力してください。