

4章 変数を理解しよう

| Pythonの変数について学びます

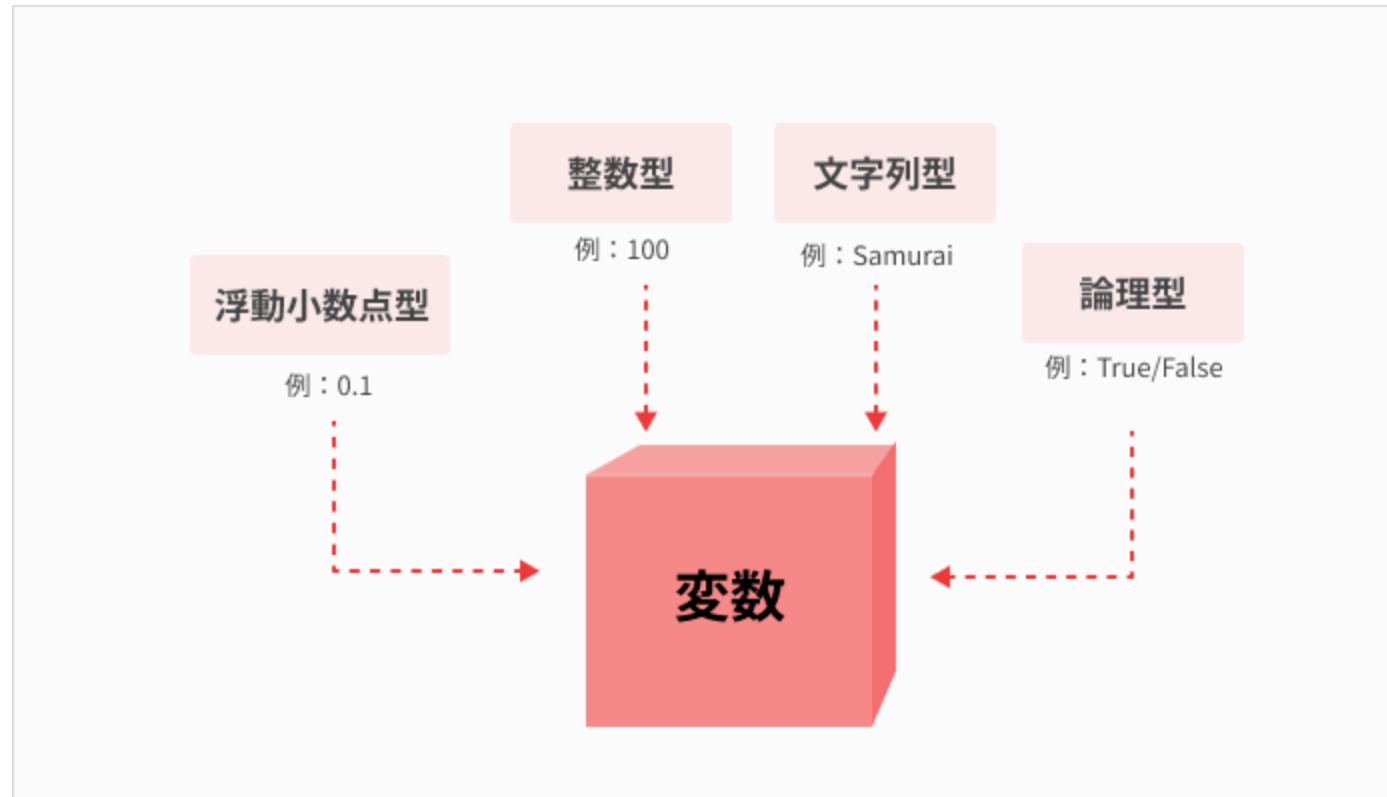
本章の目標

- 変数とは何か概要をつかむこと
- 変数の使い方を知ること
- 変数を実際に使ってみること
- 変数名のつけ方ルール（命名規則）を知ること

| 変数とは簡単にいえば、文字列や数値などのデータを入れる箱のようなものです

- 変数の中身はいつでも入れ替えられる
- Pythonをはじめとするプログラミング言語では、変数を当たり前のように使う

4章 変数とは



4章 なぜ変数が必要なのか

実際のプログラムやサービスでは、データの中身が変わることが当然のようにあります

- 前章で扱ったデータは「45」「晴れ」のように中身が決まったものだけ
- 実際のサービスでは、ユーザーごとにデータを変える必要がある

4章 変数が必要な例

画面に「おへりなさい、○○さん」と表示するWebアプリを考えましょう

The screenshot shows the homepage of the SAMURAI ENGINEER Plus application. On the left sidebar, there are navigation links: ホーム (Home), タイムライン (Timeline), カリキュラム (Curriculum), 教材 (Materials), Q&A (Q&A) with sub-links for お悩みQ&A and 技術Q&A, and 学習ログ (Learning Log). The main content area is titled "ホーム" (Home) and displays a welcome message "おかえりなさい、SAMURAIさん". Below this, there is a summary of learning statistics: SAMURAI (user icon), 総学習回数 (Total study sessions: 2), 完了教材数 (Completed materials: 2), 総レッスン数 (Total lessons: 0), and レッスンチケット (Lesson ticket: 1). A red box highlights the name "SAMURAIさん" in the welcome message. Below these stats, there are two cards: one for "次回レッスン未定" (Next lesson scheduled) with a link to "レッスンを行う >" and another for "学習中の教材" (Currently learning materials) showing "HTML/CSSの基礎を学ぼう" with a progress bar at 3/23 (13%).

| 文字列をそのまま記述する場合、ユーザーごとに変えなければならず大変です

- ○○の部分を「侍太郎」「侍花子」などユーザーごとに変える必要がある
- このとき、変数を使ってデータを変えていく

| Pythonで変数を使うには「変数名 = 変数値」と記述するだけでOKです

```
user_name = "侍太郎"
```

- このように記述することを「変数を定義する」という
- Pythonでは変数定義時にデータ型を指定する必要がない

4章 変数の使い方

変数の使い方



`user_name = ”侍太郎”`
で値を代入するだけで変数が使える

| データを代入する時点で、データ型が決まります

- 文字列を代入すれば文字列型
- 整数を代入すれば整数型として扱われる
- ほかの多くの言語では変数定義時にデータ型の指定が必要なため、Pythonのほうが楽

4章 変数名だけ書くとエラーになる

| 変数定義時にデータを指定しないとエラーになります

The screenshot shows a Jupyter Notebook cell. On the left, there is a toolbar with a play button and a timer showing "0 秒". The code cell contains the following Python code:

```
user_name
user_name = "侍太郎"
```

Below the code cell, an error message is displayed:

```
NameError                                 Traceback (most recent call last)
<ipython-input-1-7482852b6568> in <module>
----> 1 user_name
      2 user_name = "侍太郎"

NameError: name 'user_name' is not defined
```

At the bottom of the error message, there is a button labeled "SEARCH STACK OVERFLOW".

4章 変数を使ってみよう

| 変数に値を代入し、その変数の中身を出力してみます

```
user_name = "侍太郎"  
print(user_name)
```

- Visual Studio Codeを開き、新しいコードセルを追加
- 上記のコードを入力して実行

4章 変数を使ってみよう：実行結果

The screenshot shows a Jupyter Notebook cell. On the left, there is a progress bar with a green checkmark icon and the text "0 秒". Next to it is a play button icon. The main area contains Python code:

```
user_name = "侍太郎"  
print(user_name)
```

The output of the code is displayed below the code cell:

侍太郎

4章 変数の中身を入れ替えてみよう

一度定義した変数の中身は、値を代入し直すだけで入れ替えられます

```
user_name = "侍太郎"  
print(user_name)
```

```
user_name = "侍花子"  
print(user_name)
```

4章 変数の中身を入れ替えてみよう：実行結果

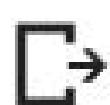


0
秒



```
user_name = "侍太郎"  
print(user_name)
```

```
user_name = "侍花子"  
print(user_name)
```



侍太郎
侍花子

4章 変数を使って計算・連結してみよう

| 変数は数値や文字列と組み合わせて計算や連結が可能です

```
# 整数型と浮動小数点型の足し算
```

```
number1 = 5
```

```
number2 = 2.5
```

```
print(number1 + number2)
```

```
# 文字列型と文字列型の連結
```

```
last_name = "侍"
```

```
first_name = "太郎"
```

```
print(last_name + first_name)
```

4章 変数を使って計算・連結してみよう：実行結果

✓
0
秒



整数型と浮動小数点型の足し算

```
number1 = 5
number2 = 2.5
print(number1 + number2)
```

文字列型と文字列型の連結

```
last_name = "侍"
first_name = "太郎"
print(last_name + first_name)
```



7.5

侍太郎

4章 文字列の中で変数を表示する方法

| f文字列を使うと、文字列の中に変数を埋め込めます

```
last_name    = "侍"
first_name   = "太郎"
sister_name = "花子"

# 3つの変数を文字列内に埋め込んで表示
print(f"私の名前は{last_name}{first_name}です。妹の名前は{sister_name}です。")
```

- 文字列の直前に「f」をつける
- 変数を表示したい部分に {変数名} と記述

4章 フォーマット済み文字列リテラル

| この記述方法を「フォーマット済み文字列リテラル」といいます

```
f"文字列{変数名1}文字列{変数名2}文字列{変数名3}....."
```

- 文字列に変数を埋め込んで表示することを変数展開と呼ぶ

4章 変数展開の実行結果

✓
0
秒



```
last_name = "侍"  
first_name = "太郎"  
sister_name = "花子"
```

3つの変数を文字列内に埋め込んで表示

```
print(f"私の名前は{last_name} {first_name}です。妹の名前は{sister_name}です。")
```

私の名前は侍太郎です。妹の名前は花子です。

4章 変数名のつけ方ルール（命名規則）

| 変数名のつけ方には以下の3つのルールがあります

① 使える文字

半角英字・半角数字・アンダースコア

② スネークケース

単語をアンダースコアで区切る

③ わかりやすい名前

変数の中身がわかる名前にする

4章 ルール① 変数名に使える文字

変数名に使える文字は以下の3種類です

使える文字	例
半角英字 (a~z、 A~Z)	name, Age
半角数字 (0~9)	number1, age2
アンダースコア (_)	user_name

- 全角文字や日本語は基本的に使用を避ける

4章 ルール① 使えない文字

| 以下の文字は変数名に使えず、エラーになります

使えない文字	例
変数名の先頭に数字	1snake
アンダースコア以外の記号	snake\case

4章 ルール② スネークケースで記述する

| Pythonでは変数名を「スネークケース」で付けることが推奨されています

スネークケース

アルファベットで複合語
(複数の単語から成り立つ語) を
記述する際に、単語と単語の間を
_ (アンダースコア) で区切る記法

例：user_name、user_number



snake_case

4章 スネークケースとは

| **単語と単語の間をアンダースコア（_）で区切る記法です**

- 英字はすべて小文字にする
- 1単語のみの場合は、アンダースコアがなくても構わない
- 例： user_name 、 phone_number 、 age

4章 キャメルケースとの違い

| JavaScriptなどでは「キャメルケース」が推奨されています

キャメルケース

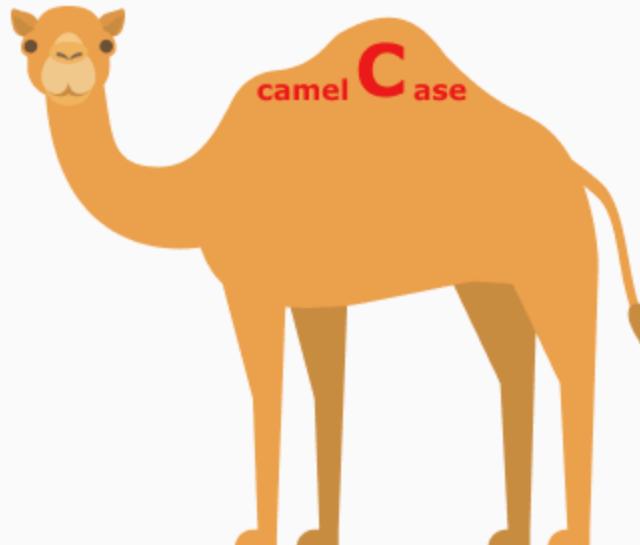
アルファベットで複合語を記述する際に各単語の先頭を大文字にする記法

ローワーキャメルケース

例：userName、userNumber

アッパーキャメルケース

例：UserName、UserNumber



4章 変数名の推奨・非推奨・NG

変数名の例を確認しましょう

推奨

```
snake_case = "◎" # 全部小文字・単語をアンダースコアで区切る  
snake      = "◎" # 1単語の場合はアンダースコアがなくてもOK
```

非推奨（エラーにはならない）

```
camelCase    = "△" # キャメルケースは非推奨  
SNAKE       = "△" # 全角は非推奨
```

NG（そもそもエラーになる）

```
1snake = "x"      # 先頭が数字は不可
```

4章 ルール③ 変数の中身がわかる名前にする



4章 良い変数名の例

| 変数の中身がわかるような名前をつけましょう

変数名	意味
user_name	ユーザー名
user_number	ユーザー番号
age	年齢
phone_number	電話番号
alert_message	警告メッセージ

4章 悪い変数名の例

| 以下のような変数名は避けましょう

悪い例	理由
xyz	変数の中身がわからない
iskeywordbutnothingor_something	変数名が長すぎる
namae	変数名は英語にするべき

本章では以下の内容を学習しました

変数とは

- 文字列や数値などのデータを入れる箱のようなもの
- 変数名 = 変数值 で値を代入するだけで使える

変数の活用

- 値を再代入すれば中身を入れ替えられる
- 数値や文字列と組み合わせて計算や連結ができる
- f文字列で変数展開ができる

変数名のつけ方ルール（命名規則）

- ① 使える文字は半角英字・半角数字・アンダースコア（_）
- ② 変数名はスネークケースで記述する
- ③ 変数の中身がわかるような名前にする

変数はプログラミングの基本中の基本です。しっかり覚えましょう