

pf-14. さまざまなプログラミング言語

(Python 入門)

URL: <https://www.kkaneko.jp/pro/pf/index.html>

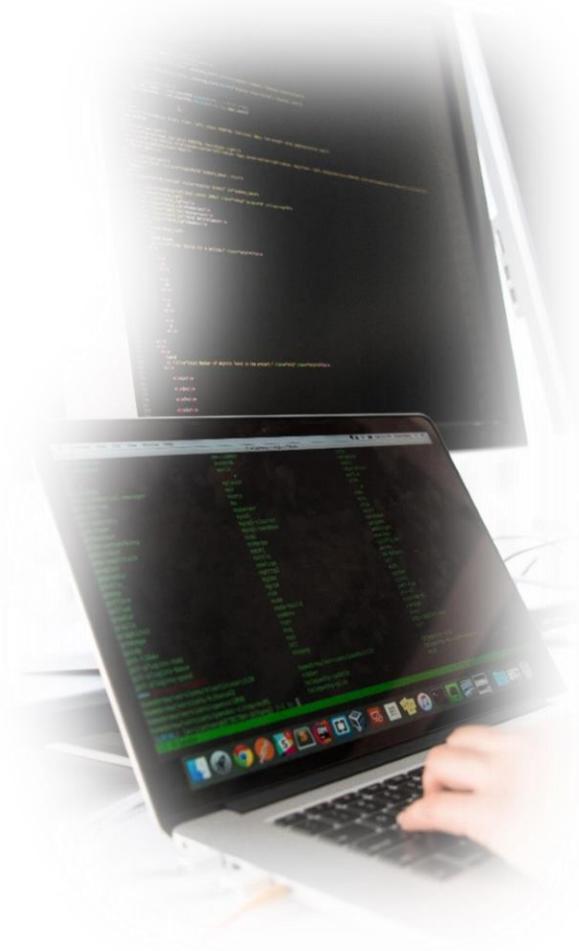
金子邦彦



プログラミングを学ぶときに気を付けること



- **プログラミング言語は多種多様**
- それぞれの言語に、特性と利用シーンがある
- **プログラミングの基本理念と基礎知識を理解することが重要.**
- **一つのプログラミング言語で基本を身につけることで、他の言語への適応もスムーズに進むでしょう**



```
x = 100
if (x > 20):
    print("big")
else:
    print("small")
s = 0
for i in [1, 2, 3, 4, 5]:
    s = s + i
print(s)
```

Python

```
public class Main {
    public static void main(String[] args) throws Exception
    {
        int x = 100;
        if (x > 20) {
            System.out.printf("big%n");
        } else {
            System.out.printf("small%n");
        }
        int s = 0;
        for(int i = 1; i <= 5; i++) {
            s = s + i;
        }
        System.out.printf("%d%n", s);
    }
}
```

Java

さまざまな
プログラミング言語

```
#include <stdio.h>
int main(void){
    int x, s, i;
    x = 100;
    if (x > 20) {
        printf("big%n");
    } else {
        printf("small%n");
    }
    s = 0;
    for(i = 1; i <= 5; i++) {
        s = s + i;
    }
    printf("%d%n", s);
    return;
}
```

C

さまざまなプログラミング言語



- Python
 - C
 - Java
 - JavaScript
 - R
 - Octave
 - Scheme
- など

ここで行う作業

1. 20 より大きければ「big」,
さもなければ「small」と表示
2. $0 + 1 + 2 + 3 + 4 + 5$ を求める

なぜプログラミング言語は たくさんあるのでしょうか？



それぞれ
特徴があ
る

Python	Java	C / C++	R	SQL	MATLAB / Octave
どのコンピュータでも同じプログラムが実行可能	シンプルで、実行も簡単. 初心者にとって学びやすい.	コンピュータの性能を最大限引き出すために適する.	データ処理に特化したコマンド言語	データベースに特化したコマンド言語	数値計算, 信号処理などに特化したコマンド言語

Python プログラム見本



```
x = 100
if (x > 20):
    print("big")
else:
    print("small")
s = 0
for i in [1, 2, 3, 4, 5]:
    s = s + i
print(s)
```

- シンプルで、実行も簡単. 初心者にとって学びやすい.
- 多種多様なパッケージを利用することで、初心者でも容易に強力な機能を追加できる.

Java プログラム見本



```
public class Main {  
    public static void main(String[] args) throws Exception {  
        int x = 100;  
        if (x > 20) {  
            System.out.printf("big%n");  
        } else {  
            System.out.printf("small%n");  
        }  
        int s = 0;  
        for(int i = 1; i <= 5; i++) {  
            s = s + i;  
        }  
        System.out.printf("%d%n", s);  
    }  
}
```

- Javaはどのコンピュータでも同じプログラムが実行可能
- Windows、Linux、そしてAndroidアプリなど、異なる環境でも同じソースコードで動作
- このように、Java は互換性が高く、広範なアプリケーション開発に適する

C プログラム見本



```
#include <stdio.h>

int main(void){
    int x, s, i;
    x = 100;
    if (x > 20) {
        printf("big¥n");
    } else {
        printf("small¥n");
    }
    s = 0;
    for(i = 1; i <= 5; i++) {
        s = s + i;
    }
    printf("%d¥n", s);
    return;
}
```

- CとC++はコンピュータの性能を最大限引き出すために適する
- 細かな制御や高速な実行に向いている
- チューニングにより最適化できる。高度なプログラミングやパフォーマンス重視のアプリケーション開発に適する

R プログラム見本



```
x <- 100
if (x > 20) {
  print("big")
} else {
  print("small")
}
s <- 0
for (i in c(1,2,3,4,5)) {
  s <- s + i
}
print(s)
```

- Rはデータ処理に特化したコマンド言語
- データ専門家にも適する
- Rは豊富な統計やデータ解析の機能を提供
- データの可視化やモデリングなどの作業を効率的に行うことが可能

Octave プログラム見本



```
x = 100
if (x > 20)
    printf("big¥n")
else
    printf("small¥n")
endif
s = 0
for i = [1 2 3 4 5]
    s = s + i
endfor
printf("%d", s)
```

- 数値計算や信号処理などに特化したコマンド言語
- 行列計算や信号処理などの科学技術計算に向いている
- 高度な数値演算やデータ解析が容易に行える

JavaScript プログラム見本



```
process.stdin.resume();
process.stdin.setEncoding('utf8');
var util = require('util');
var x = 100;
if (x > 20) {
    process.stdout.write('big¥n');
} else {
    process.stdout.write('small¥n')
}
var s = 0;
for(var i = 1; i <= 5; i++) {
    s = s + i;
}
process.stdout.write(util.format('%d¥n', s));
```

- インタラクティブなウェブページの作成に適する
- そのとき、ユーザーとのリアルタイムな対話、動的なコンテンツの表示が可能
- 幅広い種類の OS でサポートされている

Scheme プログラム見本



```
(define (decide x)
  (cond
    ((> x 20) "big")
    (else "small")))

(define (sum n)
  (cond
    ((= n 0) 0)
    (else (+ (sum (- n 1)) n))))

(begin
  (print (decide 100))
  (print (sum 5)))
```

- シンプルで明確な構文を持つ
- 関数型プログラミング言語
- 強力な再帰処理や高階関数の活用が簡単にできる

さまざまな種類のプログラミング言語



- プログラミングの**基本理念と基礎知識**を理解していくことが重要
- 一つの言語で基礎を身につけることで、**他の言語への適応もスムーズに進む**

なぜプログラミング言語はたくさんあるのか？

- **異なるニーズや目的に対応**
- 広範な用途に適するもの（Python, Java, JavaScript など）もあれば、特定の領域でより強力な機能を提供するものも
- **自分の目標や学びたいことに応じて言語を選ぶことが重要。**
複数の言語を使い分けることもある。