Python 講座　第 1 週

2017/6/22

1　導入

Python は C と比べてかなり少ないコード行数で記述することができるプログラミング言語です。シンプルに記述できるので可読性が高まり作業効率を高めます。実行速度は C に比べて遅くなります。

1.1　インストール

1.1.1　Windows

Windows ではまず初めに公式サイトから Python3.x.x をダウンロードしてきます。ダウンロードが完了したら、インストーラを実行してインストールしてください。python をコマンドプロンプトから実行するためには Path を設定する必要があります。「コントロールパネル」→「システムとセキュリティ」→「システム」→「システムの詳細設定」→「環境変数」から Path の値に Python3.x.x がインストールされている場所を追加します。

1.1.2　Mac

Mac には 2 系のみインストールされています。そのため 3 系を使用するためにはインストールする必要があります。端末上でインストールするためには以下のコマンドを実行します。

```
brew install python3
```

インストールが完了したら PATH の設定を行います。

1.1.3　Linux

Linux には初めから Python がインストールされています。これらには 2 系と 3 系が両方インストールされているので端末上で「python」コマンドを実行すると 2 系が実行されます。3 系で実行したい場合は「python3」コマンドを実行します。

2　基本事項

2.1　インデント

python ではインデントが重要になります。C や Java では { } でブロックを構成していましたが、python ではインデントでブロックを構成します。以下の例のように C では if 文の処理を { } で囲んでいます。python の場合は if 文の処理は 1 つ字下げされています。最後の print 文は字下げされていないため if 文を抜けたことがわかります。

indent

```
# C
if (a  % 2 == 0){
    printf("The variable a is an even number.\n");
}

puts("exit");

# python
if a % 2 == 0:
    print("The variable a is an even number.")

print("exit")

# python (error)
if a % 2 == 0:
print("The variable a is an even number.")  # インデントされていないのでエラーになる
```

2.2　コメント

python ではコード内にコメントを記述する場合は#や”””を使用します。1 行だけのコメントの場合は#, 複数行に渡ってコメントを記述したい場合は””” ”””で囲みます。

2.3　日本語

コード内に日本語を記述したい場合以下のようなコードではエラーとなります。

```
print("Hello World") # "Hello World を出力する。
```

日本語のような ASCII コード以外の文字を記述するためにはコードの先頭に「encode: 」でエンコードルー

ルを指定します。

```
# coding: utf-8
print("Hello World") # "Hello World を出力する。
```

3 出力

Python でコンソール上に出力するためには print 関数を使用します。

print

```
# coding: utf-8

print("Hello Python!!")
```

実行結果

```
Hello Python!!
```

4 変数

4.1 型

C や Java では整数型 (int, short, long, ...), 浮動小数点数型 (float, double) でしたが、python3 では整数型 (int), 浮動小数点数型 (float) です。以下に数値・文字列型をまとめます。

型	値	例
int	整数	1, 2, 3
float	浮動小数点数	1.2, 5.66, 0.11
bool	論理値	True or False
complex	虚数	3.14j, 2.1j
str	文字列	”Hello”, ”Python”

※ 型は type 関数で調べることができます。

type

```
# coding: utf-8

print(type(123))

print(type("Mojiretsu"))
```

実行結果

```
<class 'int'>
<class 'str'>
```

4.2 動的型付け

Python の特徴として「動的型付け」があります。C や Java のように変数宣言時に型を明示する必要がありません。変数の型は値から自動で判定してくれます。

verb

```
# coding: utf-8

a = 5  # 整数
print(type(a))

s = "Python"  # 文字列
print(type(s))
```

4.3 入力

Python でコンソールから値を読み込む場合は input 関数を使用します。input 関数の引数に文字列を渡すことができます。引数に渡された文字列は入力待ちの時にコンソールに出力されます。input 関数で値を読み込むとそれは文字列 (str) として読み込まれます。値を整数として読み込みたい場合は int 関数を使用します。

input

```
# coding: utf-8

str 型で読み込む
n = input()
print(n)
print(type(n))

# int 型で読み込む
m = int(input())
print(m)
print(type(m))

i = input("Input -->")
print(i)
```

実行結果

```
56
56
<class 'str'>
4
4
<class 'int'>
Input -->hello
hello
```

5 if 文

python における if 文は次のように記述します。

```
if 条件 1:
    処理 1
elif 条件 2:
    処理 2
else:
    処理 3
```

C や java と違っている部分としては C や Java では「if else（条件) { }」だった部分が「elif 条件: 」となっているので注意してください。また、C や Java では switch 構文がありましたが python ではありません。もし、switch 構文を使いたい場合は「if elif else」を使用して表します。

switch

```
# C
int a = 1;
switch(a){
    case 0:
        printf("a is 0.\n");
        break;
    case 1:
     printf("a is 1.\n");
        break;
    case 2:
     printf("a is 2.\n");
        break;
    default:
        printf("OK\n");
}

# python
a = 1
if a == 0:
    print("a is 0.")
elif a == 1:
    print("a is 1.")
elif a == 2:
    print("a is 2.")
else:
    print("OK");
```