PYTHON入門

篠原・吉仲研究室

野崎 裕樹

今日の内容

ctfでつかいそうなpythonの話をします

- 対象者
  - python使ったことない人
  - pythonでとりあえず書いたことがある人
  - ライブラリあんま使わない人
- 非対象者
  - pythonマスター

主にライブラリの紹介になると思います
（途中問題あり)

とりあえず動かしてみる

インタープリターの起動とHello,World

```
$ python
>>> hello = "Hello, World"
>>> hello
'Hello, World'
```

インタープリターの終了

```
>>> exit()
$
```

電卓がわりに使う

四足演算

```
>>> a = 6
>>> b = 2
>>> a + b
8
>>> a - b
4
>>> a * b
12
>>> a / b
3.0
```

ビットシフト

```
>>> 1 << 3
8
```

10進数

10進数の数字を扱う(数字⇄文字)

```
>>> 123
123
>>> str(123)
'123'
>>> int('123')
123
>>> '%08d' % 123
'00000123'
```

16進数

16進数の数字を扱う
基本は0xXXの形で表現できる

- 10 ⇄ 16

```
>>> hex(123)
'0x7b'
>>> int('0x7b', 16)
123
>>> int('7b', 16)
123
```

2進数

2進数の数字を扱う
基本は0bxxxxxxの形で表現される

```
>>> bin(123)
'0b1111011'
>>> int('0b1111011', 2)
123
>>> format(123, 'b')
'1111011'
>>> print(format(123,'016b'))
0000000001111011
```

文字⇄数字

charと数値の変換

```
>>> ord('A')
65
>>> chr(65)
'A'
>>> hex(65)
'0x41'
>>> chr(0x41 + 1)
'B'
```

文字列/リスト

文字列と配列の変換

```
>>> list('abc')
['a', 'b', 'c']
>>> list(map(ord, 'abc'))
[97, 98, 99]
>>> list(map(chr, [97,98,99]))
['a', 'b', 'c']
>>> list(map(hex, map(ord, 'abc')))
['0x61', '0x62', '0x63']
>>> "".join(['a','b','c'])
'abc'
```

部分文字列

文字列中の一部分をとる

```
>>> text = "JinnnoKazunari"
>>> text[0]
'J'
>>> text[0:6]
'Jinnno'
>>> text[-1]
'i'
>>> text[:6]
'Jinnno'
>>> text[6:]
'Kazunari'
```

文字列分解・連結

特定の文字とかで分割したり結合したり

```
>>> '83 85 71 65 84 65 75 85'.split()
['83', '85', '71', '65', '84', '65', '75', '85']
>>> '53:45:43:43:4f:4e'.split(':')
['53', '45', '43', '43', '4f', '4e']
>>> ' '.join(['83', '85', '71', '65', '84', '65', '75', '85'])
'83 85 71 65 84 65 75 85'
>>> ':'.join(['53', '45', '43', '43', '4f', '4e'])
'53:45:43:43:4f:4e'
```

応用

CTF問題（仮）：83 85 71 65 84 65 75 85

```
>>> a = '83 85 71 65 84 65 75 85'.split(' ')
>>> a
['83', '85', '71', '65', '84', '65', '75', '85']
>>> b = list(map(chr, map(int, a)))
>>> b
['S', 'U', 'G', 'A', 'T', 'A', 'K', 'U']
>>> ''.join(b)
'SUGATAKU'
```

CTF過去問(5 ~ 10分)

CSAW CTF 2011 Crypto1

```
87 101 108 99 111 109 101 32 116 111 32 116 104 101 32 50 48
49 49 32 78 89 85 32 80 111 108 121 32 67 83 65 87 32 67 84
70 32 101 118 101 110 116 46 32 87 101 32 104 97 118 101 32
112 108 97 110 110 101 100 32 109 97 110 121 32 99 104 97 108
108 101 110 103 101 115 32 102 111 114 32 121 111 117 32 97
110 100 32 119 101 32 104 111 112 101 32 121 111 117 32 104
97 118 101 32 102 117 110 32 115 111 108 118 105 110 103 32 116
104 101 109 32 97 108 108 46 32 84 104 101 32 107 101 121 32
102 111 114 32 116 104 105 115 32 99 104 97 108 108 101 110
103 101 32 105 115 32 99 114 121 112 116 111 103 114 97 112
104 121 46
```

CTF過去問(5 ~ 10分)

CSAW CTF 2011 Crypto2

```
54:68:69:73:20:69:73:20:74:68:65:20:66:69:72:73:74:20:6d:65:73:
73:61:67:65:20:62:65:69:6e:67:20:73:65:6e:74:20:74:6f:20:79:6f:
75:20:62:79:20:74:68:65:20:6c:65:61:64:65:72:73:68:69:70:20:6f:
66:20:74:68:65:20:55:6e:64:65:72:67:72:6f:75:6e:64:20:55:70:72:
69:73:69:6e:67:2e:20:49:66:20:79:6f:75:20:68:61:76:65:20:64:65:
63:6f:64:65:64:20:74:68:69:73:20:6d:65:73:73:61:67:65:20:63:6f:
72:72:65:63:74:6c:79:20:79:6f:75:20:77:69:6c:6c:20:6e:6f:77:20:
6b:6e:6f:77:20:6f:75:72:20:6e:65:78:74:20:6d:65:65:74:69:6e:67:
20:77:69:6c:6c:20:62:65:20:68:65:6c:64:20:6f:6e:20:57:65:64:6e:
65:73:64:61:79:20:40:20:37:70:6d:2e:20:57:65:20:77:69:6c:6c:20:
61:6c:73:6f:20:72:65:71:75:69:72:65:20:61:20:6b:65:79:20:74:6f:
20:62:65:20:6c:65:74:20:69:6e:74:6f:20:74:68:65:20:6d:65:65:74:
69:6e:67:73:3b:20:74:68:69:73:20:77:65:65:6b:1f:73:20:6b:65:79:20
```

答え

```
>>> a = '54:68:69:73:20:69:73:20:74:68:65:20:66:69:72:73:74:20:6d
73:73:61:67:65:20:62:65:69:6e:67:20:73:65:6e:74:20:74:6f:20:79:6f
20:62:79:20:74:68:65:20:6c:65:61:64:65:72:73:68:69:70:20:6f:66:20
68:65:20:55:6e:64:65:72:67:72:6f:75:6e:64:20:55:70:72:69:73:69:6e
2e:20:49:66:20:79:6f:75:20:68:61:76:65:20:64:65:63:6f:64:65:64:20
68:69:73:20:6d:65:73:73:61:67:65:20:63:6f:72:72:65:63:74:6c:79:20
6f:75:20:77:69:6c:6c:20:6e:6f:77:20:6b:6e:6f:77:20:6f:75:72:20:6e
78:74:20:6d:65:65:74:69:6e:67:20:77:69:6c:6c:20:62:65:20:68:65:6c
20:6f:6e:20:57:65:64:6e:65:73:64:61:79:20:40:20:37:70:6d:2e:20:57
20:77:69:6c:6c:20:61:6c:73:6f:20:72:65:71:75:69:72:65:20:61:20:6b
79:20:74:6f:20:62:65:20:6c:65:74:20:69:6e:74:6f:20:74:68:65:20:6d
65:74:69:6e:67:73:3b:20:74:68:69:73:20:77:65:65:6b:1f:73:20:6b:65
20'.split(':')
>>> a
['54', '68', '69', '73', '20', '69', '73', '20', '74', '68', '65'
'66', '69', '72', '73', '74', '20', '6d', '65', '73', '73', '61',
```

エンコード・デコード

BASE64

```
>>> import base64
>>> encoded = 'base64 encode'
>>> base64.b64encode(encoded.encode('utf-8'))
b'YmFzZTY0IGVuY29kZQ=='
>>> decoded = b'YmFzZTY0IGVuY29kZQ=='
>>> base64.b64decode(decoded)
b'base64 encode'
```

全てバイトオブジェクトで扱う

エンコード・デコード

UU

- encode.txt

```
encode uu
```

- python

```
>>> import uu
>>> uu.encode('encode.txt', 'result.txt')
```

- result.txt

```
begin 644 encode.txt
*96YC;V1E('5U"@
                        (空行)
end
```

エンコード・デコード

ROT13

```
>>> import codecs
>>> codecs.encode('hello', 'rot13')
'uryyb'
>>> codecs.decode('uryyb', 'rot13')
'hello'
```

応用(5 ~ 10分)

KSNCTF:Easy Cipher

```
EBG KVVV vf n fvzcyr yrggre fhofgvghgvba pvcure gung ercynprf n
yrggre jvgu gur yrggre KVVV yrggref nsgre vg va gur nycunorg. EBG
KVVV vf na rknzcyr bs gur Pnrfne pvcure, qrirybcrq va napvrag Ebz
Synt vf SYNTFjmtkOWFNZdjkkNH. Vafreg na haqrefpber vzzrqvngryl ns
```

答え

KSNCTF:Easy Cipher

```
EBG KVVV vf n fvzcyr yrggre fhofgvghgvba pvcure gung ercynprf n
yrggre jvgu gur yrggre KVVV yrggref nsgre vg va gur nycunorg. EBG
KVVV vf na rknzcyr bs gur Pnrfne pvcure, qrirybcrq va napvrag Ebz
Synt vf SYNTFjmtkOWFNZdjkkNH. Vafreg na haqrefpber vzzrqvngryl ns
```

```
>>> codecs.encode('EBG KVVV vf n fvzcyr yrggre fhofgvghgvba pvcur
ercynprf n yrggre jvgu gur yrggre KVVV yrggref nsgre vg va gur ny
EBG KVVV vf na rknzcyr bs gur Pnrfne pvcure, qrirybcrq va napvrag
Synt vf SYNTFjmtkOWFNZdjkkNH. Vafreg na haqrefpber vzzrqvngryl ns
, 'rot13')
'ROT XIII is a simple letter substitution cipher that replaces a
the letter XIII letters after it in the alphabet. ROT XIII is an
the Caesar cipher, developed in ancient Rome. Flag is FLAGSwzgxBJ
Insert an underscore immediately after FLAG.'
```

ハッシュ

```
>>> import hashlib
>>> hashlib.sha256(b'Hello World').digest()
b'\xa5\x91\xa6\xd4\x0b\xf4 @J\x01\x173\xcf\xb7\xb1\x90\xd6,e\xbf\
>>> hashlib.sha256(b'Hello World').hexdigest()
'a591a6d40bf420404a011733cfb7b190d62c65bf0bcda32b57b277d9ad9f146e
>>> hashlib.md5(b'Hello World').hexdigest()
'b10a8db164e0754105b7a99be72e3fe5'
```

可能なアルゴリズム一覧

```
shlib.algorithms_available
{'sha256', 'blake2s', 'SHA1', 'shake_256', 'ecdsa-with-SHA1', 'md
'SHA384', 'blake2b', 'sha3_224', 'sha512', 'dsaEncryption', 'sha'
'md5', 'MDC2', 'SHA512', 'sha1', 'MD5', 'shake_128', 'sha384', 'D
'DSA-SHA', 'sha3_512', 'SHA256', 'sha3_256', 'md4', 'sha224', 'MD
'SHA224', 'RIPEMD160', 'dsaWithSHA', 'whirlpool', 'SHA', 'ripemd1
'sha3_384'}
```

ファイルの入出力

test.py

```
a = [1, 2, 3, 4, 5, 6, 7, 8, 9]

 for i in a:
       print(i * 6)
```

ファイルの入出力

```
>>> src = open('./test.py', 'r')
>>> dst = open('./out.py', 'w')
>>> for line in src :
...     dst.write('# ' + line)
...
34
3
14
19
>>> open('./out.py', 'r').read()
'# a = [1, 2, 3, 4, 5, 6, 7, 8, 9]\n# \n# for i in a:\n#'
```

外部コマンドの実行

```
>>> import subprocess
ubprocess.call(["echo", "hello world"])
hello world
0                              #返り値(成功:0, 失敗:1)
>>> command = subprocess.check_output(["echo", "hello world"])
>>> command
b'hello world\n'
```

python3.5系から

```
>>> res = subprocess.run(["echo", "hello world"], stdout=subproce
>>> res.stdout
b'hello world\n'
```

HTTP通信

GET

```
>>> import urllib.request
>>> url = "http://www.example.com"
>>> res = urllib.request.urlopen(url)
>>> res.read()
b'<!doctype html>\n<html>\n<head>\n    <title>Example Domain</tit
<meta charset="utf-8" />\n    <meta http-equiv="Content-type" con
<meta name="viewport" content="width=device-width, initial-scale=
body {\n        background-color: #f0f0f2;\n        margin: 0;\n
font-family: "Open Sans", "Helvetica Neue", Helvetica, Arial, san
{\n        width: 600px;\n        margin: 5em auto;\n        padd
#fff;\n        border-radius: 1em;\n    }\n    a:link, a:visited
text-decoration: none;\n    }\n    @media (max-width: 700px) {\n
#fff;\n        }\n        div {\n            width: auto;\n
border-radius: 0;\n            padding: 1em;\n        }\n    }\n
<h1>Example Domain</h1>\n    <p>This domain is established to be
You may use this\n    domain in examples without prior coordinati
```

応用

問題は配ります