プログラミングゼミ③
-Python-

2017/05/09

朝倉研究室M1

尾頭　尚人

今日の内容

- オブジェクトの型
- 基本構文（for文・if文・while文）
- 例外処理
- 関数
- csv(データ解析)

オブジェクト

- プログラムで処理する対象.　型によって扱い方が変わる.

	型	表記	例
数値	整数	int	1
数値	浮動小数点数	float	3.14
	文字列	str	‘python’
配列	リスト	list	[1,2,3]
配列	タプル	tuple	(1,2,3)
配列	辞書	dict	{'Month':‘5','day’:'20'}
	ファイル	file	csvファイル

```
In [15]: a = 10
         b = '5'
         c = [0,1,2]
         type(a), type(b), type(c)

Out[15]: (int, str, list)
```

type( ) で
型を調べられる

基本構文　~for文~

繰り返し処理(ループ)で使用

```
for 変数名 in リストや文字列など:
        ’’’実行する内容’’’
```

```
In [1]: a = [2,8,4]
        for i in a:
            print i

        2
        8
        4
```

リストaから0番目の要素2を取り出し出力

リストaから1番目の要素8を取り出し出力

リストaから2番目の要素4を取り出し出力

変数 i

基本構文　~for文~

(例)for文を使ってリストの全要素の和を計算

```
In [2]: a = range(1,6)
        total = 0
        for i in a:
            total += i
        print total

        15
```

リスト[1, 2, 3, 4, 5]を生成

初期値設定

total = total + i と同じ
繰り返す度にtotalの値が更新
(total=0+1=1
 total=1+2=3
 total=3+3=6….)

ちなみにsum関数を使えば一瞬で答えが出る

```
In [3]: sum(a)
Out[3]: 15
```

基本構文　~if文~

条件によって動作を変えることができる

```
if 条件1:
    '''実行する内容'''
elif 条件2 :
    '''実行する内容'''
else :
    '''実行する内容'''
```

条件がもっとあるなら
elifを用いて増やす

どの条件にも当てはまらない場合

```
In [9]: a = 555
        if a%3==0:
            print 'a=3n'
        elif a%3==1:
            print 'a=3n+1'
        else:
            print 'a=3n+2'

        a=3n
```

%：余剰
a はどの形で表されるか？

基本構文　~while文~

for文と同じく、繰り返し処理で使用

条件を満たしている間はずっと繰り返す

```
while 条件:
    '''実行する内容'''
```

(例)a<100の間、倍になってく

```
In [1]: a = 1
        while a<100:
            a = a*2
            print a
```

初期値設定

```
2
4
8
16
32
64
128
```

以下の例だと**無限ループ**する
(動作が重くなる可能性があるので、実行しないほうがいいかも)
iPython:■で停止？
Python: Ctrl+Cで強制停止

```
In [ ]: a = 1
        while a>0:
            a = a*2
            print a
```

基本構文　~while文~

whileの別の書き方．特に、条件によって繰り返し処理の仕方が変わるときなどに有効.

```
while True:
        条件(ifなど)：
            break
        '''実行する内容'''
```

breakでループから抜け出せる

(例)初期値a=1で
　0<a≦50のとき、倍になっていき
　50<a≦100のとき、20ずつ増えて
　100<aで終了

```
In [2]: a = 1
        while True:
            if 0<a<=50:
                a = a*2
            elif 50<a<100:
                a += 20
            else:
                break
            print a
```

```
2
4
8
16
32
64
84
104
```

例外処理

何らかのエラーにより実行できないときに、
それを「例外」として扱うことができる

```
try:
    '''実行する内容'''
except:
    '''例外に対して実行する内容'''
```

(例) 10をリストaの各要素で割った値を,空のリストbに順に格納する

```
In [25]: a = [1,5,0,2]
         b = []
         for i in a:
             b.append(10/i)
         print b
         ---------------------
         ZeroDivisionError
```

空のリスト作成

リストbに要素10/ i を追加していく

要素に0があるため**エラー**

```
In [24]: a = [1,5,0,2]
         b = []
         for i in a:
             try:
                 b.append(10/i)
             except:
                 b.append('error')
         print b
         [10, 2, 'error', 5]
```

計算不可の要素を'error'と表示

関数

処理をひとまとめにしたもの
引数(input)を指定したら戻り値(output)が返ってくる

```
def 関数名(引数) :
    '''実行する内容'''
    return 戻り値
```

(例) 3つの数の和、積を求める関数

```
In [41]: def a(x,y,z):
             b = x+y+z
             c = x*y*z
             return b,c

         a(2,3,4)
Out[41]: (9, 24)
```

関数

引数に値を指定

戻り値に指定したb, cが返ってくる

csv (pandasを使用したデータ解析)

In [9]:
```
import pandas as pd
castle = pd.read_csv('kanko_castle.csv')
castle
```

Out[9]:

	year	open of park	all(adult)	all(child)	general(adult)	general(child)	party(adult)	party(child)	discount(adult)	discount(child)
0	H23	366	28199	1645	25191	745	2826	899	182	1
1	H24	364	28944	1224	26094	746	2732	478	118	0
2	H25	364	28727	1024	27281	560	1347	464	99	0
3	H26	364	44052	2157	30857	786	3183	735	10012	636
4	H27	365	56840	2855	36950	968	3841	884	16049	1003

In [12]:
```
castle.shape
```
← 次元数確認　(行数, 列数)

Out[12]: (5, 10)

In [13]:
```
castle.describe()
```
← 各列の統計量の確認(平均,　分散など)

Out[13]:

	open of park	all(adult)	all(child)	general(adult)	general(child)	party(adult)	party(child)	discount(adult)	discount(child)
count	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000
mean	364.600000	37352.400000	1781.000000	29274.600000	761.000000	2785.800000	692.000000	5292.000000	328.000000
std	0.894427	12782.161801	741.054316	4800.578121	145.220522	914.586081	211.743949	7379.720083	467.061559
min	364.000000	28199.000000	1024.000000	25191.000000	560.000000	1347.000000	464.000000	99.000000	0.000000
25%	364.000000	28727.000000	1224.000000	26094.000000	745.000000	2732.000000	478.000000	118.000000	0.000000

まとめと練習

今回のゼミでは、特に汎用性の高い基本構文
を中心に説明した

分からない箇所は調べれば大体出てくるので、
とにかく実際に手を動かして習得しよう

練習

①9頁の例外処理の例をtry,exceptでなく
　if を使って同じ結果を出力

②forを使って [[1,2,3],[4,5,6]]の要素を4頁のように順番に出力

③4頁の例をforでなくwhileを使って同じ結果を出力

④if文、for文を使ってリスト[1,7,3,6,5]の要素の最大値を求める

コードの例

①

```
In [8]: a = [1,5,0,2]
        b = []
        for i in a:
            if i != 0:# != は≠と同じ
                b.append(10/i)
            else:
                b.append('error')
        print b
```

```
[10, 2, 'error', 5]
```

コードの例

②

```
In [9]: a = [[1,2,3],[4,5,6]]
        for i in a:
            for j in i:
                print j
```

```
1
2
3
4
5
6
```

コードの例

③

In [10]:
```
a = [2,8,4]
k = 0
while k<len(a):#len関数で配列の長さ取得
    print a[k]
    k += 1
```

```
2
8
4
```

コードの例

④

```
In [11]: a = [1,7,3,6,5]
         max = a[0]#初期値をaの最初の要素に設定
         for i in a:
             if max >= i:#最大値がiより大きければこのまま
                 max = max
             elif max <= i:#iが最大値より大きければ最大値更新
                 max = i
         print max

         7
```