関数を定義する

初めてのプログラミング

2019年度

只木進一（理工学部）

サンプルプログラムの取得

- GitHubRepositoryを指定
  - https://github.com/first-programming-saga/functions

関数を定義する目的

- 機能毎にプログラムを作る
  - 動作確認が容易
  - 繰り返し利用
- 小さな関数を沢山書くのが良い

関数を定義する：引数の無い関数 Functions without arguments

```
def 関数名():
    関数の実体
```

- 関数の中で、何か処理する
  - 必要に応じて結果を返す
- 結果を戻す
  - return 文
- 呼び出しは、単に関数名を使う
  - 戻り値があれば、代入する

functions/simpleFunctions.ipynb

関数を定義する:引数の有る関数 Functions with arguments

```
def 関数名(引数並び):
    関数の実体
```

- 引数
  - 関数に渡す変数
  - 全て参照渡し
    - int, float, str, などは元の値は変わらない
    - mutable変数は元の値が変わる
- 結果を戻す
  - return 文

functions/argumentsTest.ipynb

引数と戻り値のある関数

Functions with arguments and return values

```
def quadraticFunction(x,a,b,c):
    y = a*x*x + b*x + c
    return y

a = 1
b = -2
c = 1
for i in range(100):
    x = i * 0.1
    y=quadraticFunction(x,a,b,c)
    message = f'f({x})={y}'
    print(message)
```

$$f(x) = ax^2 + bx + c$$

functions/quadraticFunction.ipynb

再帰的関数
Recursive function

```
def factorial(n):
    if n==1:
        return 1
    return n * factorial(n-1)

n = 5
a = factorial(n)
print(a)
```

再帰的に呼び出し

$$n! = \begin{cases} n \times (n-1)! & n > 1 \\ 1 & n = 1 \end{cases}$$

再帰的関数：関数が、自身で定義されている

functions/factorial.ipynb

引数と戻り値のある関数
タプルを戻す例

```
def isPrime(n):
    result = False
    if n <= 0:
        message=f'引数は正でなければならない'
    elif n < 2:
        message = f'{n}は素数ではない'
    elif n == 2:
        message = f'{n}は素数である'
        result = True
    elif n % 2 == 0:
        message = f'{n}は偶数であり、素数ではない'
    else:
        m=int(math.sqrt(n))
        for k in range(3,m+1,2):#forで記述
            if n % k == 0:
                message = f'{n}は{k}で割り切れるため、素数ではない'
                break
        else:#ループの最後まで至った場合
            message = f'{n}は素数である'
            result = True
    return result,message
```

functions/isPrime.ipynb

引数がリストの関数

```
import math

def stat(data):#dataは数値のリスト
    n = len(data)
    s=0#和を保存
    for x in data:#data中のすべてに対して繰り返し
        s += x
    average = s/n#平均
    return n,average

#データリスト
data=[3,5,7,2,3,6,1,4,9,2]
n,average=stat(data)
print('データ数 :',n)
print('平均:',average)
```


functions/stat0.ipynb

変数の有効範囲：scope

- 関数の引数や関数内で定義された変数
  - 関数の内部だけで有効：ローカル変数
  - 別の関数に同じ名前が現れても別物
- 関数の外で定義された変数
  - グローバル変数
  - 多用は要注意

引数名を指定した呼び出し

- 引数名を指定することで、引数順序に従わずに利用できる

```
#引数名の指定
x1,x2=quadratic(c=1,b=2,a=1)
print(x1,x2)
```


functions/quadratic.ipynb

引数の省略

```
def squareSum(list,s=0):
    for d in list:
        s += d*d
    return s

data = [4,2,6,4,1]
s = squareSum(data)
print(s)
```

次回

- 13章「テキストファイルの読み込みと書き出し」