現象数理1（2020年度 第7回授業）

黒岩大史 kuroiwa@riko.shimane-u.ac.jp

今回の授業日と課題〆切

対面：11月19日　オンデマンド推奨期間：11月12日〜11月25日　課題〆切：11月25日

はじめての関数

今回は「関数」と呼ばれるものについて学習します。ここでの関数は、数学でいう関数と同じようなもので、変数（引数（ひきすう））に対する値（戻り値（もどりち））を定めたようなものです。

例えば「二つの整数の最大公約数を求める」「ある自然数が素数かどうかを判定する」「配列（次回学習内容）の内容を表示する」など、本文中で何度も使うような機能を関数にします。12と15の最大公約数を求める方法と、382と254の最大公約数を求める方法は同じですから、一般の自然数xとyの最大公約数を求める方法を関数に記述するのです。

自然数xとyを与える　→　関数内部で最大公約数を求める　→　結果を得る

一般に関数では、「何か数や分量」を与え、それに対する「結果」を得ます。前者のことを**引数（ひきすう）**、後者のことを**戻り値（もどりち）**と言います。

1　関数の例

例1. 「二つの整数を与えて最大公約数を求める部分」を関数にしたプログラムを作成してみましょう。(以下では説明のために行番号をつけています。また判りやすさのため、ユークリッドの互助法を使わず説明します)

[ファイル名 2020-7-1.py]

```
1 def gcd(x, y):
2     n=1
3     for i in range(1,x):
4         if(x%i==0 and y%i==0):
5             n=i
6     return n

7 print(gcd(30, 42))
8 print(gcd(164, 287))
```

実行結果

```
6
41
```

このプログラムは関数の部分と本文との行間を開けて書いています。一見判りにくいと思うので、まずはあえて関数の部分（1〜6行目）を無視してみます（下記参照）。そうすると、gcd(30, 42)や

gcd(164, 287) の意味が判らないことを除くと**簡単なプログラム**になります。関数を使うことで、それ以外の部分は非常にシンプルになります。

```
7 print(gcd(30, 42))
8 print(gcd(164, 287))
```

← gcd(30, 42) の値と、gcd(164, 287) の値を画面に出す（非常にシンプル）。

gcd の意味は上で無視した 1〜6 行目に書かれており（下記参照）、最終的に gcd(30, 42) が 30 と 42 の最大公約数 6 を表します。さて、上記プログラムを実行した際の動作について書きます。

[基本] 関数の定義が沢山あっても、**関数は最初には実行されず、関数以外の部分が上から順に実行**されます。途中で**関数が呼ばれると、関数が上から順に実行され、関数が終了すれば呼ばれた位置に戻り続きを実行**します。これを上のプログラムに当てはめると次のようになります。

- 関数以外の部分は 6, 7 行目です。6 行目の print(gcd(30, 42)) が実行されると、関数 gcd が呼び出されます（30, 42 は引数と呼ばれます）。
- 関数 gcd において、x=30、y=42 としてこの関数内部に書いてあることが実行されます。
- 5 行目までで n の値は 30 と 42 の最大公約数、つまり n=6 となります。
- 6 行目の「return n」の実行で関数は終了し、それと同時に gcd(30, 42) の位置に n の値 6 が戻され、結果として print(6) が実行されます。
- 7 行目の print(gcd(164, 287)) も同様にして実行されます。つまり関数 gcd が x=164、y=287 として呼び出され、結果として n=41 が得られ、return n によって print(41) が実行されます。

雰囲気は判ったでしょうか。以下に関数の約束事を書きます。

一般に関数は次の形をしています。

```
def 関数名 (引数, ⋯ ):
    処理
    return 戻り値
```

- 例 1 の関数の「関数名」は gcd です。関数名には (予約語等以外の) 好きな名前を付けることができます。
- 関数 gcd の引数は、x と y です。一般には引数の数は何個でも (0 個でも) 良いです。関数内で用いた引数は、この関数内のみで有効です（関数外に値を持ち出せません）。
- 処理部分で計算などを行い、計算結果を「戻り値」の部分に書きます。return が実行されれば（関数内の途中の位置だったとしても）直ちにこの関数の実行が終わり、本文の関数を呼び出した場所に戻り値が入ります。
- 計算結果が不要の場合には、return を省略できます。

例 2. 1 つの数を引数とし、その絶対値を戻り値とする関数 abs を用いてプログラミングしてみましょう。

[ファイル名 2020-7-2.py]

```
def abs(x):
    if(x>0):
       return x
    else:
       return -x

a = int(input("整数を入力して下さい："))
print(abs(a))
```

実行例

```
整数を入力して下さい：4
      4
```

```
整数を入力して下さい：-3
      3
```

このプログラムの関数について、次のことがわかります。

- 関数名は abs
- 関数 abs の引数は x
- 関数 abs の戻り値は、引数であたえた数の絶対値になる。実際、この関数では、x が正ならば return x が、そうでなければ return -x が実行される。

例 3. 2 数を引数とし、大きい方の値を戻り値とする関数 max を用いてプログラミングしてみましょう。

[ファイル名 2020-7-3.py]

```
def max(x, y):
   if(x < y):
      return y
   else:
      return x

a = int(input("数を入力して下さい："))
b = int(input("数を入力して下さい："))
print(大きいのは",max(a, b))
```

実行例

```
整数を入力して下さい：5
整数を入力して下さい：7
      7
```

2 練習問題

以下の問題では、本文は各自適当と思うものでよいです。

(1) 変数 n を引数とし、$\sum_{i=1}^{n} i$ を戻り値とする関数 sigma を用いてプログラムを作成して下さい。和の計算には必ずしも for 文や while 文を使う必要はありません。また n が 0 以下ならば戻り値は 0 として下さい。なお、n が整数でない場合の動作は、どうであっても構いません。(2020-7-4.py)

(2) 2 変数 a と n を引数とし、a^n を計算する関数 power を用いてプログラムを作成して下さい。ただし、a が 0 のときや n が整数でない場合の動作は、どうであっても構いません。(2020-7-5.py)

(3) キーボードから入力した整数が素数かどうか判定するプログラムを作成して下さい。その際、関数名は primetest とし、引数 x の値が素数なら 1、そうでないなら 0 を戻り値とする関数を使って判定して下さい。(2020-7-6.py)

(4) 2 以上、キーボードから入力した整数以下に、何個の素数があるかを数えるプログラムを作成して下さい。その際、2 以上 n 以下の素数の個数を数える関数 countprime を作成し、また関数 countprime 内では上で作った関数 primetest を用いて下さい。(2020-7-7.py)

3 まとめ

前回は関数を学習しました。

- 関数には引数（ひきすう）と戻り値（もどりち）がある。
- 関数内には機能を実現するための手続きを記述する。
- 関数を一度定義しておけば、どこでも何度でも使える。

関数の仕組みを理解し、例題は何も見ずに書けるレベルになりましょう。

補足：例１再考

ユークリッドの互除法は、最大公約数を求めるための優れたアルゴリズムです。例１ではあまり良くないアルゴリズムを使ったため、ユークリッドの互除法を使った場合を以下に示します。左側はスタンダードな書き方で、while で余りが 0 になるまで繰り返します。「x=y」および「y=r」を、まとめて「x,y=y,r」と書くこともできます。右側は関数内で同じ関数を呼び出す機能（再帰呼び出し）を使ったもので、これも余りが 0 でない限り新たに gcd(y,r) を呼び出し続け、最後に余りが 0 になった段階で y を return します（再帰呼び出しは慣れると便利ですが、慣れるまでは全く使えない場合が多いので、理解できなくても気にする必要はありません）。

[ファイル名 2020-7-8.py]

```
def gcd(x, y):
    r=x%y
    while r>0:
        x=y
        y=r
        r=x%y
    return y

print(gcd(30, 42))
print(gcd(164, 287))
```

[ファイル名 2020-7-9.py]

```
def gcd(x, y):
    r=x%y
    if r==0:
        return y
    else:
        return gcd(y,r)

print(gcd(30, 42))
print(gcd(164, 287))
```