

情報 I

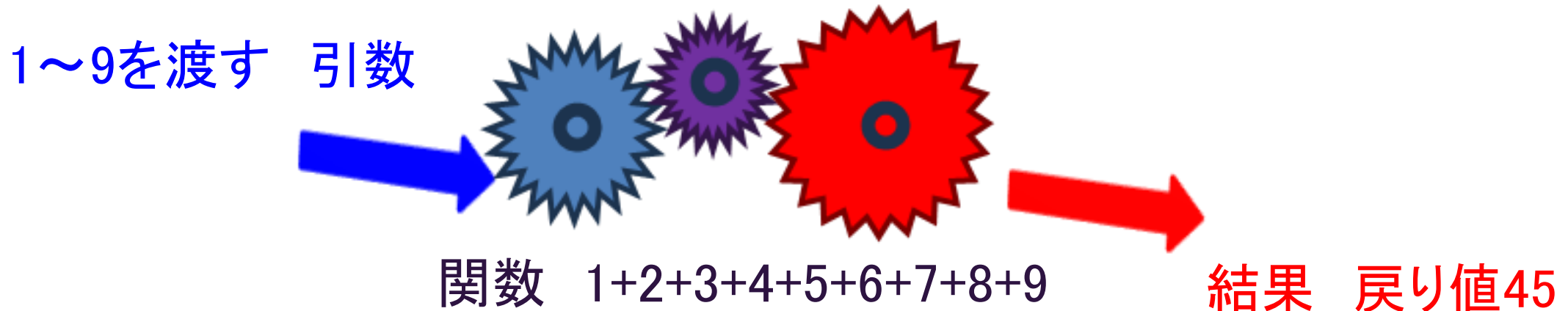
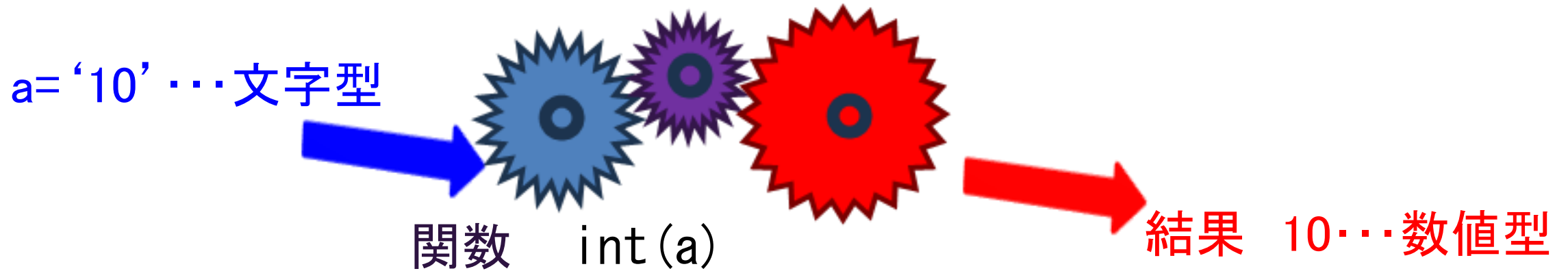
4-1 Python作法1

4-13 関数モジュール

関数の集積体アプリ=ライブラリ活用のメリット



関数def (define) : 複数処理をまとめてワンアクションで実行
ここまでの範囲でPythonが内蔵しているint(x)、float(y)などを活用している。加えて自分で作ることができる。



4-13-1 税込み価格を計算

```
def f(x):           # 関数function(x)を定義
    p=x*1.1        # p は本体価格 x を 1.1 倍する税込価格
    return p       # 税込み価格(戻り値)を返せ
print(f(200))      # 本体価格を200(引数)として関数処理を実行し戻り値を表示
```



```
def f(x):
    p=x*1.1
    return p
print(f(200))
print(int(f(200)))
```



4-13-2 合計値を求める関数(1)

```
def add(x,y):  
    total=0  
    total=sum(range(x,y))  
    return total  
print(add(1,10))
```

2変数をx,yとして関数addを設定
合計値totalの初期値0
合計値totalはx以上y未満で合計せよ
合計値totalを返せ
1~9(引数)を合計して、戻り値を表示




```
def add(x,y):  
    total=0  
    total=sum(range(x,y))  
    return total  
print(add(1,10))
```



4-13-2 合計値を求める関数(2)

```
def add(x,y):          # 2変数をx,yとして関数addを設定
    total=0           # 合計値の初期値0
    for i in range(x,y): # 変数 i はx以上y未満で反復
        total += i     # 合計値totalに変数 i を加えて合計値とする
    return total       # 合計値totalを戻り値とする
print(add(1,10))      # x=1、y=10としてadd関数を走らせて戻り値を表示
```



```
def add(x,y):
    total=0
    for i in range(x,y):
        total += i
    return total
print(add(1,10))
```



(13-4) 2次方程式における解の公式を実装

cmath

```
import cmath
```

```
def g(a,b,c):
```

```
    D = (b**2- 4*a*c)
```

```
    If D>0 or D==0:
```

```
        p = (-b+D**0.5)/(2*a)
```

```
        q = (-b-D**0.5)/(2*a)
```

```
        return p,q
```

```
    else:
```

```
        r= -b/(2*a)
```

```
        i= ((D**0.5)/(2*a)).imag
```

```
        p= complex(r, i)
```

```
        q= complex(r, -i)
```

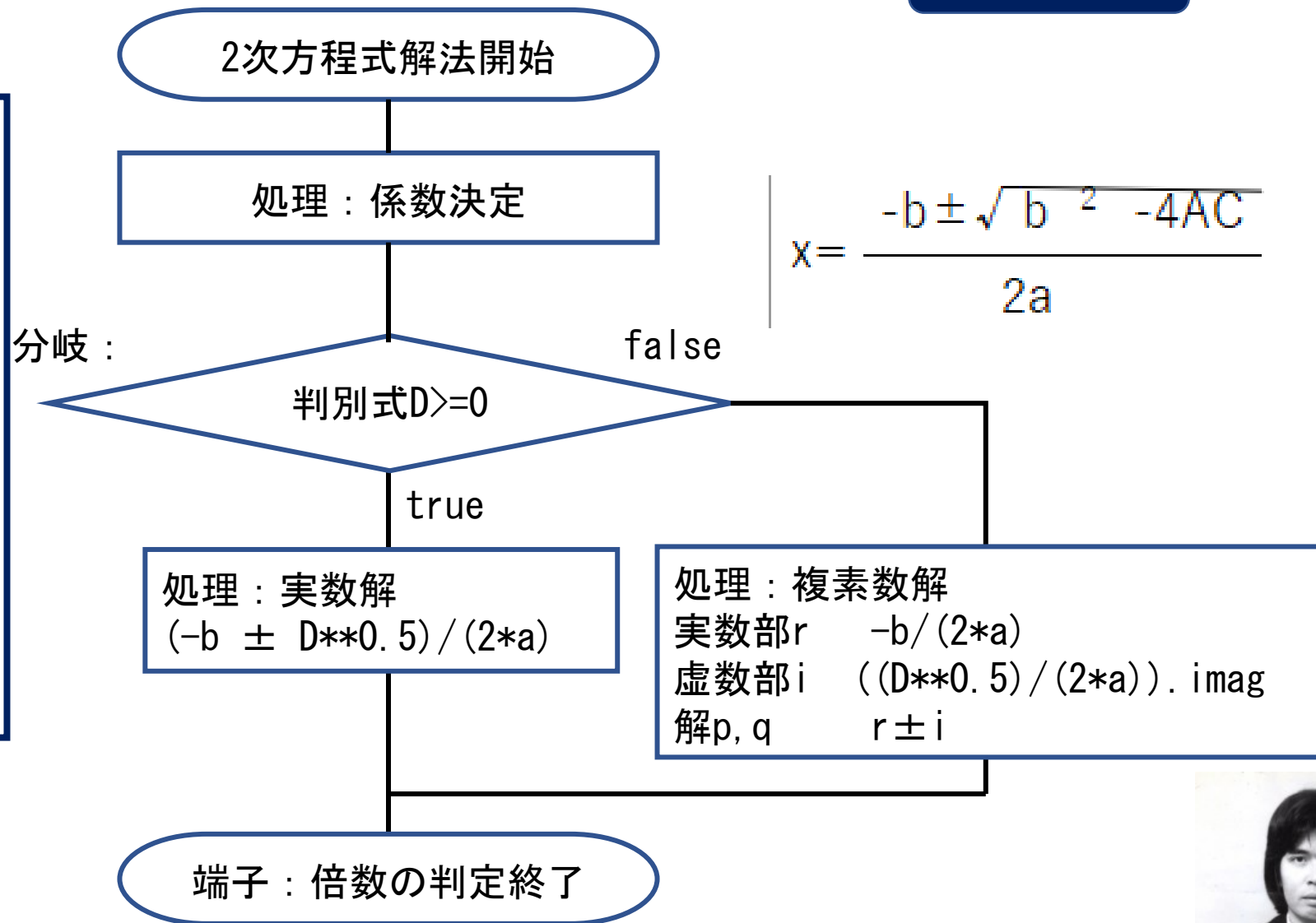
```
        return p,q
```

```
    a=float(input(" a "))
```

```
    b=float(input(" b "))
```

```
    c=float(input(" c "))
```

```
    print(g(a,b,c))
```



$$x = \frac{-b \pm \sqrt{b^2 - 4AC}}{2a}$$



(26-6) 二次方程式解法 $X^2-3x+2=0$

sympy

```
import sympy
```

```
a=float(input("a"))
```

```
b=float(input("b"))
```

```
c=float(input("c"))
```

```
x=sympy.Symbol('x') # 変数定義
```

```
s=sympy.solve(a*x**2 + b * x + c) # 方程式の種類を選ばない！
```

```
print(s)
```

目的に特化した関数をアプリ化
↓
モジュール・ライブラリ…ツール

```
import sympy
```

```
x = sympy.Symbol('x')
```

```
ans=sympy.solve(90*x**4 - 405*x**3 - 225*x**2 + 1890*x)
```

```
print(ans)
```

4次方程式 $90x^4-405x^3-225x^2+1890x=0$

