情報処理概論

九州大学　工学部地球環境工学科　講義資料　担当：木村

04.定回反復・不定回反復

九州大学
KYUSHU UNIVERSITY

処理の「流れ」を制御する

１）プログラムは基本的に**上から下へ順番に実行**される

２）制御構造によって**条件分岐**や**反復**を行うことができる

条件分岐
　if - else 文，　if – elif – else 文

反復
　for 文，while 文

定回／不定回反復とその必要性

定回反復：あらかじめ決められた回数同じ処理を繰り返す

例：
- 3以下の奇数を表示せよ.
  print(“1,3”);
- 20以下の奇数を表示せよ.
  print(“1,3,5,7,9,11,13,15,17,19");
- 10000以下の奇数を表示せよ.
  print(“1,3,5,7,9,….");　　全部書けるのか？？

不定回反復：ある条件を満たす間、同じ処理を繰り返す

例：0が入力されるまで入力値を加算していく.

for 文による定回反復

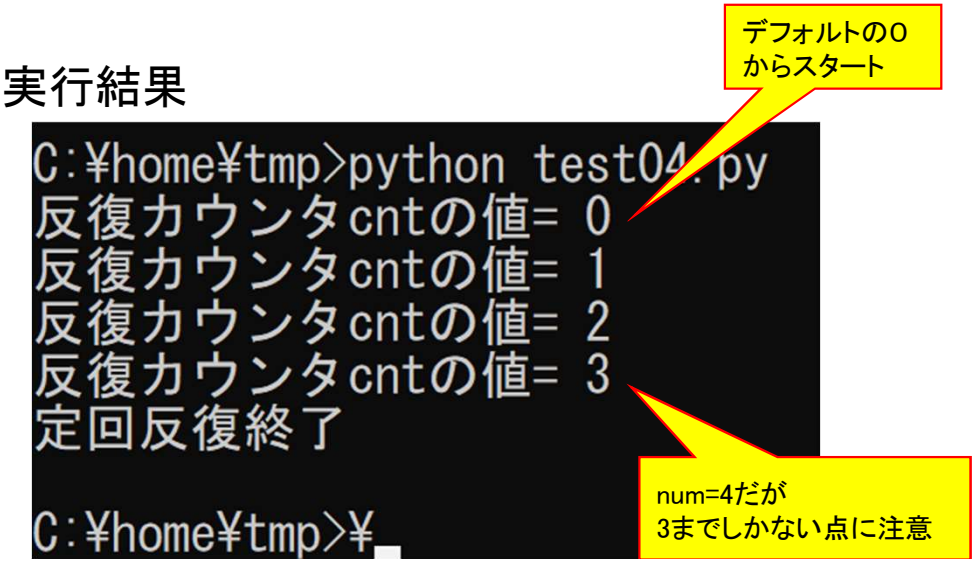
```
for 変数 in range([start,]stop[,step]) :
    ブロック
```

- 変数が start, …., stop未満の間ブロックを繰り返し実行する
- start は省略可能（デフォルト値は0）
- stepは省略可能（デフォルト値は1）

例：

```
num = 4
for cnt in range( num ):
    print(“反復カウンタcntの値=“,cnt)
print(“定回反復終了")
```

実行結果



for 文の使用例

入力された値以下で、3の倍数であるものをすべて表示する

```
print(“自然数を入力してください”)
num = int(input())
for x in range( 1, num ):
        remainder = x % 3
        if( remainder == 0 ):
                print( cnt,  end=“ “)
print(“終了")
```

1からnumまでブロックAを繰り返す

xを3で割った余りを代入

このサンプルコードのままでは、
不具合が生じる場合がある

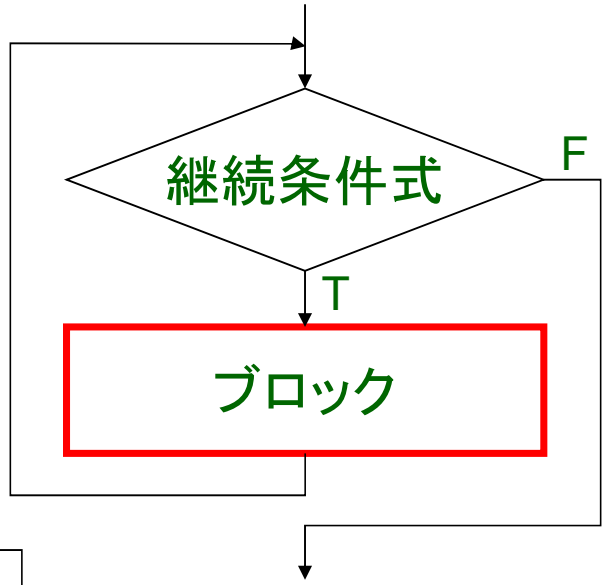
どんな不具合？
不具合をなくすには？

実行結果

```
C:¥home¥tmp>python test04_2.py
自然数を入力してください
10
3 6 9 終了

C:¥home¥tmp>¥
```

while 文による不定回反復

```
while 継続条件式：
    ブロック
```



- 継続条件式が成り立っている間，ブロックを実行する
- 条件が成り立たなければ，ループは終了する．
- **ブロックの実行前に条件を評価する**ので，ブロックが1回も実行されない場合もある．

T： 真（条件が成立）
F： 偽（条件が不成立）

while 文の使用例

```
# 0が入力されるまで入力値を加算していく.
sum = 0
x = 1  # 変数xを用意するため

while x != 0:  # 0が入力されるまで繰り返される

        x = input("足す数を入力して下さい:")
        x = int(x)
        sum += x

print("sum ={}“, sum))
```

sum = sum + x

実行結果

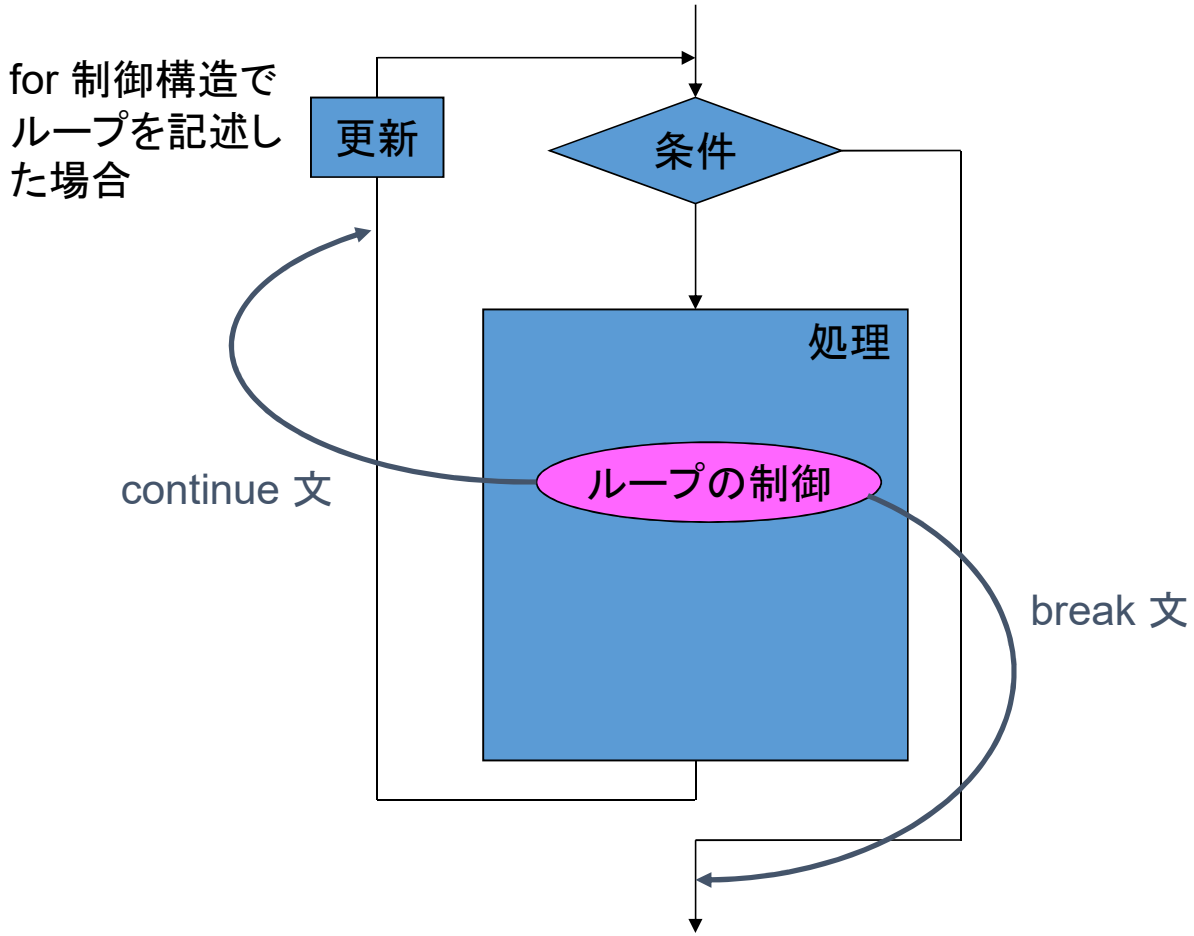
```
C:¥home¥tmp>python test04_3.py
足す数を入力して下さい:2
足す数を入力して下さい:3
足す数を入力して下さい:4
足す数を入力して下さい:5
足す数を入力して下さい:0
sum =14

C:¥home¥tmp>¥
```

ループの途中での実行の制御

- ループの途中で，
  繰り返している処理を…

  - 終了させたい．
    （ループから抜ける．）
    - break 文

  - ループ内の残りの処理を省きたい．
    （ループは続行）
    - continue 文
    - ループのブロックで使用可能



繰り返し処理の表現方法いろいろ

例： 1から10までの合計を計算する

- 以下の3種類のループはどれを使っても構わない.

for ループ

```
sum = 0
for k in range(1,11):
        sum += k
#print(sum)
```

while ループ

```
k = 1
sum = 0
while k <= 10:
        sum += k
        k += 1
```

変数 k はループ変数
（ループの回数を数えるための変数）

whileとbreak

```
sum = 0
k = 1
while True:
        sum += k
        k += 1
        if k > 10:
                break
```

継続条件式が常にTrue
→無限ループ

停止条件を満たしたら
break文でループを抜ける

通常のwhile文とwhile-break文を使ったループの違い

```
while 継続条件式：
    ブロック
```

- **継続条件式**が成り立っている間，ブロックを実行する
- 条件が成り立たなければ，ループは終了する．
- **ブロックの実行前に条件を評価する**ので，**ブロックが1回も実行されない場合もある．**

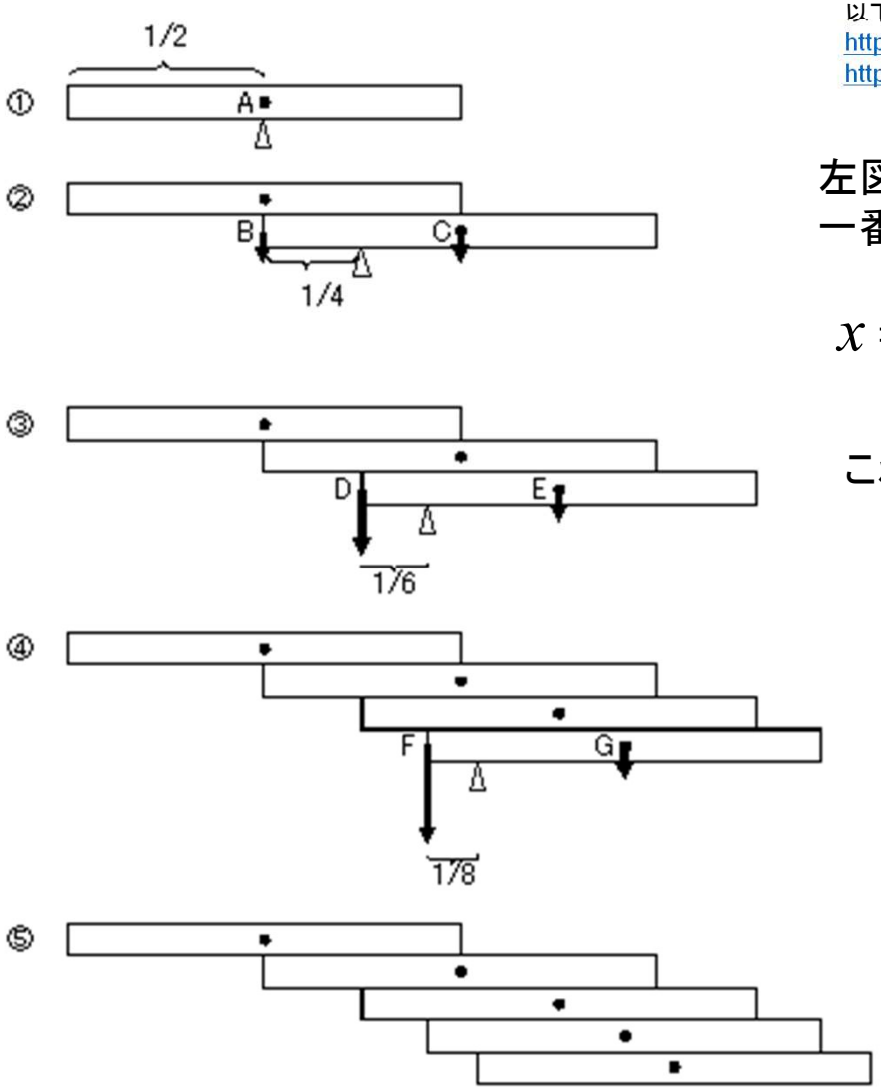
```
while True：
    ブロック
    if 停止条件式：
        break
```

継続条件式が常にTrue
→無限ループ

停止条件を満たしたら
break文でループを抜ける

- **停止条件式**が成り立ったらループを抜ける
- **ブロックの実行後に条件を評価する**ので，**ブロックは少なくとも1回は必ず実行される．**

練習問題：積み木と調和級数



以下のブログより
http://ameblo.jp/eau-libre/entry-10835418809.html
http://homepage3.nifty.com/kuebiko/science/39th/sci_39.htm

左図のように、長さ1の積み木をnコ積むと、
一番上の積み木は机の端より

$$x = \frac{1}{2} + \frac{1}{4} + \frac{1}{6} + \cdots + \frac{1}{2n} = \frac{1}{2}\sum_{k=1}^{n}\frac{1}{k}$$

これだけの長さだけ張り出すことができる。

$\sum_{k=1}^{n}\frac{1}{k}$ は調和級数と呼ばれ、
n→∞で発散する。

ここで、机の端からの距離 xの値が入力されたら、
そこまで積み木が張り出すのに必要な積み木の個数 n を
求めるには？

ヒント1： while文を利用する

```
print("机の端からの距離を入力してください")

dest = float(input())

num=0

x=0

while x < dest :

    num = num + 1.0
    x = x + (0.5 / num)

print("必要な積み木の個数：",num)
```

目標とする机の端からの距離入力

ブロックの個数

実際にブロックが机の端から張り出す距離

ブロックを１コ追加

追加分の張出し距離を追加

課題１：素数の判定プログラム

入力された値が正の整数であるとき、これが素数かどうかを判定するプログラムを作成せよ。

※「素数」とは、１とそれ自身の数で割り切れない数（2,3,5,7,11,…）

ヒント１：　for文を利用すること
ヒント２：　for文のカウンタの値がスタートする値と終了するときの値に注意
ヒント３：　**プログラムの途中で処理を終了**するときはbreakの代わりに **quit()** を使う

プログラムの先頭行にコメント文で自分の氏名と学籍番号を入れておくこと

← この記号より左側の文字列はコメントになる

課題２：数当てゲーム

0から99までのランダムな数を生成し、その数をプレイヤが当てるまで数字入力を繰返す「数当てゲーム」を作成せよ。
ただし入力された値が正解より大きい／小さい場合はその旨を表示し、正解した場合は数字入力の回数を報告して終了すること。

※ランダムな整数を発生させる方法

```
import random
x = random.randint( a, b )
```

randomモジュールを利用

a ≦ x ≦ b の整数であるような一様な乱数xを生成する

ヒント1： while文を使用すること

まとめ

定回反復　for 文, 不定回反復　while 文

for文の場合、カウンタがスタートする値と終了する値に注意

ループを途中で抜け出したい場合：break文

ループ内の残りの処理を省いて、再度ループ処理を行いたい場合：continue文

第4回　レポート課題提出方法

課題1のプログラムの後ろに、コメント行を数行挿入してから課題2のプログラムをそのままつなげ、
下記の課題提出用フォルダへ、課題の番号と提出者が分かるようにファイル名を以下のようにしてアップロードせよ
第4回1TE19xxxZ名前.py
https://share.iii.kyushu-u.ac.jp/public/lRbwAAVITI5A2X4BE45t6TqQIE0UQSQUl5Bap_kZ_sjy

講義資料、および上記フォルダへのリンクは下記ホームページから
http://sysplan.nams.kyushu-u.ac.jp/gen/edu/InfoProcess/2019/index.html