2022年度・鼓山塾

# K-15　デジタル社会の技術（プログラミング）
# 4. くり返し処理

伊東栄典

九州大学情報基盤研究開発センター

ito.eisuke.523@m.kyushu-u.ac.jp

# 参考資料


Python ゼロからはじめるプログラミング
著者　　：三谷純
出版社　：翔泳社
発売日　：2021/5/24
ISBN　　：9784798169460

講義用のスライドも提供
https://mitani.cs.tsukuba.ac.jp/book_support/python/

本資料も，上記のスライドを援用しています。

# 1. プログラムの制御構造（復習）

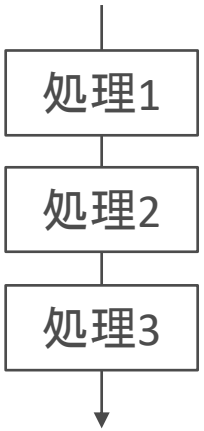
1. プログラムの制御構造（復習）
2. if 文による条件分岐
3. 比較演算子
4. 論理演算子
5. 演算子の優先度とカッコ

# 1. プログラムの制御構造　（再掲）

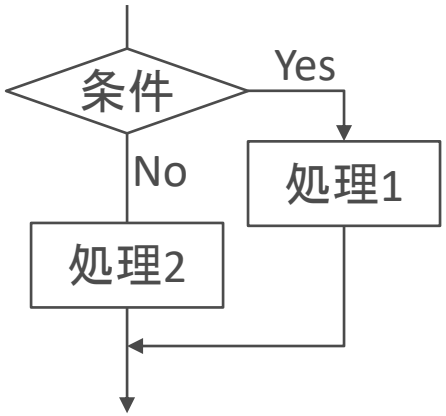
アルゴリズムおよびプログラムの制御構造は3種類

| 制御構造s | 内容 |
|---|---|
| 順次 (Sequence) | 1つ1つの処理を順番に行う |
| 分岐 (Branch, Jump) | ある条件に応じて異なる処理を実行する |
| 反復 (Iterative, Loop) | ある条件が満たされている間はその処理を繰り返し実行する |

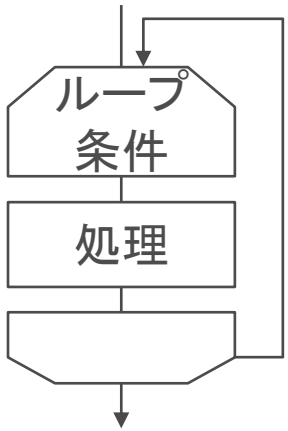
順次



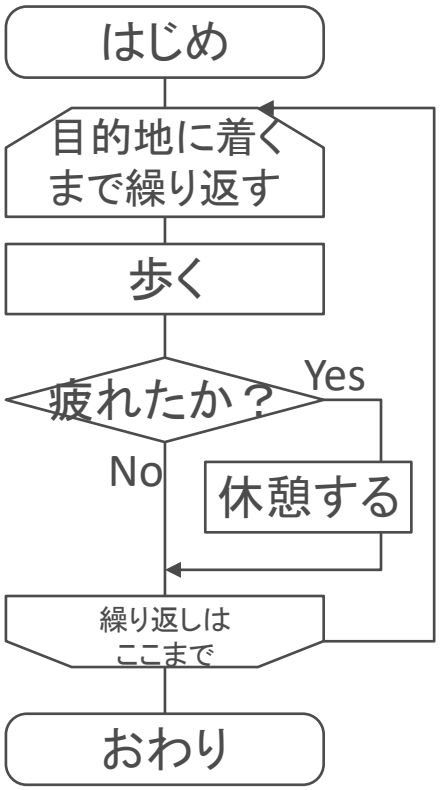

分岐




反復

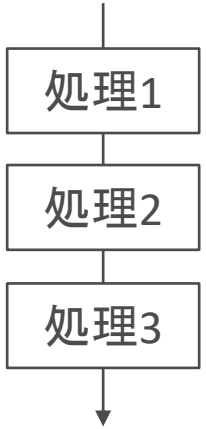



例：歩行



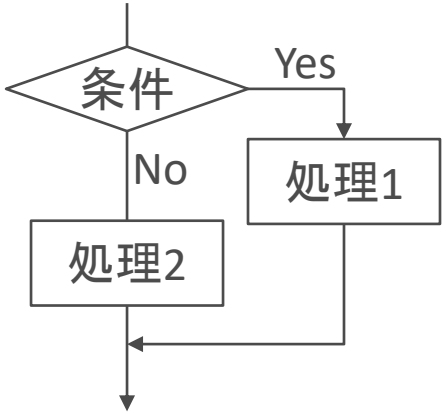
# 制御構造とプログラム例 (Python言語)

順次



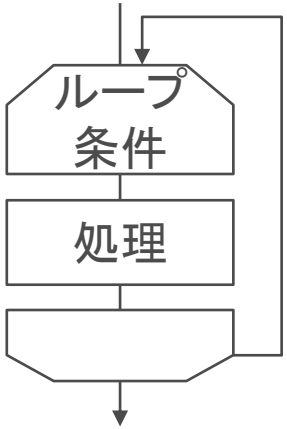

```
print(“Hello World”)
x = 1+2
print(x)
```

分岐




```
age = 16
if(age>=18):
  print(“You can vote”)
else:
  print(“You can’t”)
```

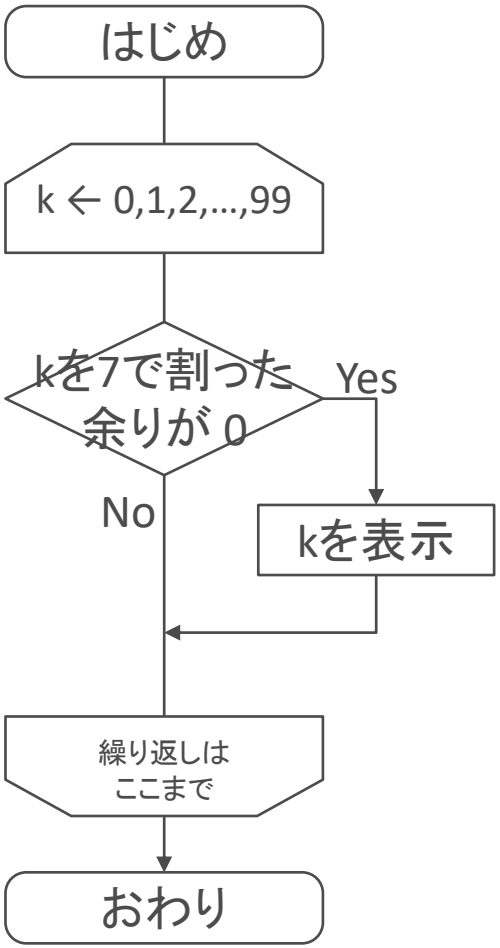
反復




```
sum = 0
for x in range(0..10):
  sum = sum + x

print(“合計 ”, sum)
```

例：0から99までの整数のうち，
7で割ったら余りが0となる
数を表示




Pythonプログラム

```
for k in range(100):
    if(k%7==0):
        print(k)
```

# 2. 繰り返し処理とは

1. プログラムの制御構造（復習）
2. 繰り返し処理とは

# 2. 繰り返し処理とは

- コンピュータは繰り返し処理が大得意
  - 計算機は，いつまででも同じ事を，間違えずに，繰り返せる。
  - （人間は，すぐに飽きるし，間違える。
- 大量のデータを計算機で処理することは，計算機に適している。
- Pythonの繰り返し処理は２つ
  - **for 文**
    - 一定回数，もしくはリストの全要素に対して，一連の処理を繰り返す。
  - **while 文**
    - while文が示す条件が成立している間，一連の処理を繰り返す。

# 3. for文

1. プログラムの制御構造（復習）
2. 繰り返し処理とは
3. for文

# 3. for文

- for文の構文（書き方）

```
for 変数 in 反復可能オブジェクト:
    処理内容
```

  - 「反復可能オブジェクト」から1つずつ要素をとりだして「変数」に代入。
  - 「処理内容」を繰り返す。

- 例

```
for x in [10, 20, 30, 40, 50]:
  print(x)
```

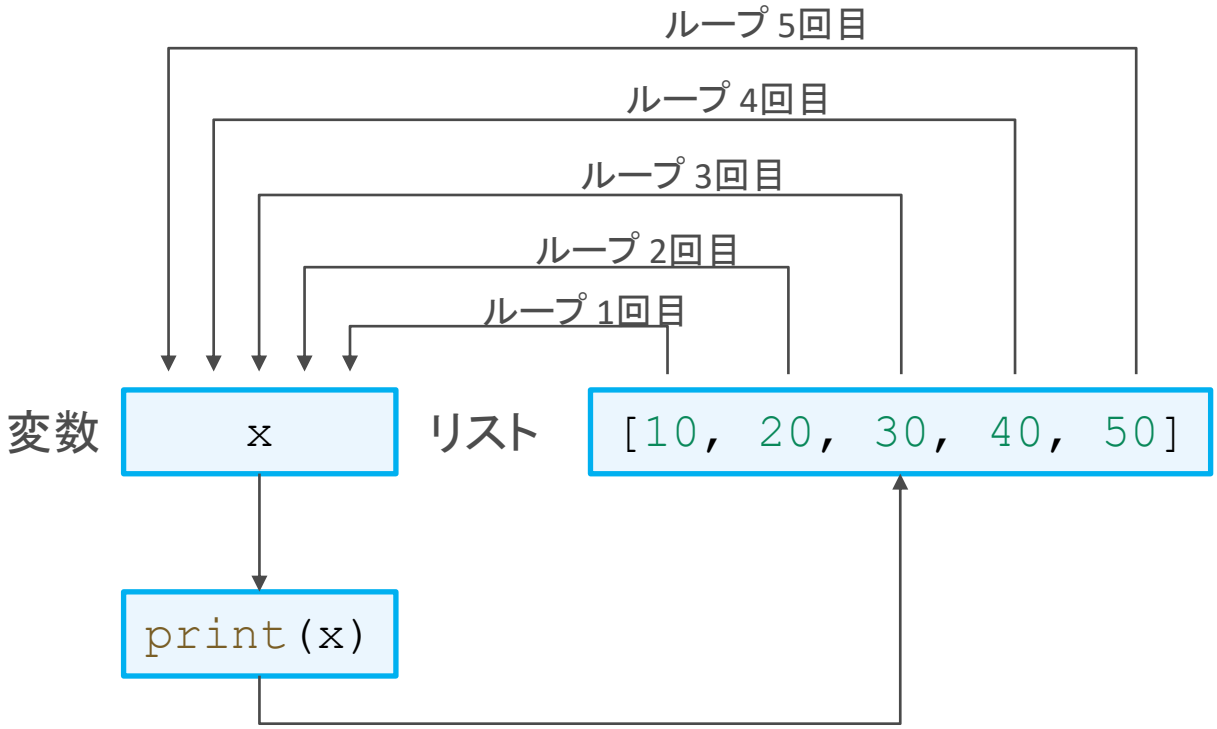
```
10
20
30
40
50
```

```
lis = [10, 20, 30, 40, 50]
for x in lis:
  print(x)
```

```
10
20
30
40
50
```

# 3.1 for文の処理の流れ

```
for x in [10, 20, 30, 40, 50]:
  print(x)
```



# 3.2 range オブジェクト

- 固定回数の繰り返しで良く使う

```
for i in range(10):
  print(i)
```

変数 i に 0, 1, …, 9 の値が順番に代入される。

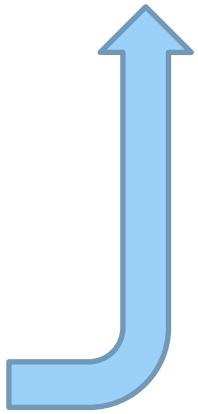
```
0
1
2
:
9
```

```
begin = 1
end   = 10
skip  = 2
for i in range(begin, end, skip):
  print(i)
```

```
1
3
5
7
9
```

※ 変数 begin, end, skip に整数値が入ってるとする

| rangeの生成方法 | 得られる整数の列 |
|---|---|
| rand(end) | 0から「end-1」までの整数 |
| range(begin, end) | beginから「end-1」までの整数 |
| range(start, end, skip) | beginから「end-1」までの整数。<br>ただし増分は skip の値。 |



# range オブジェクト

```
>>> list(range(10))
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> list(range(3, 10))
[3, 4, 5, 6, 7, 8, 9]
>>> list(range(1, 30, 10))
[1, 11, 21]
```

0から9までの数字が並びます

3から9までの数字が並びます

29を超えない範囲で1から10ずつ値が増えます

```
for i in range(100, 201, 5):
    print(i)
```

100から始まり5ずつ増える値が、200に達するまで順番に代入されます

実行結果

```
100
105
110
(略)
200
```

いろいろ試してみよう！

# 3.3 繰り返し処理の流れの変更

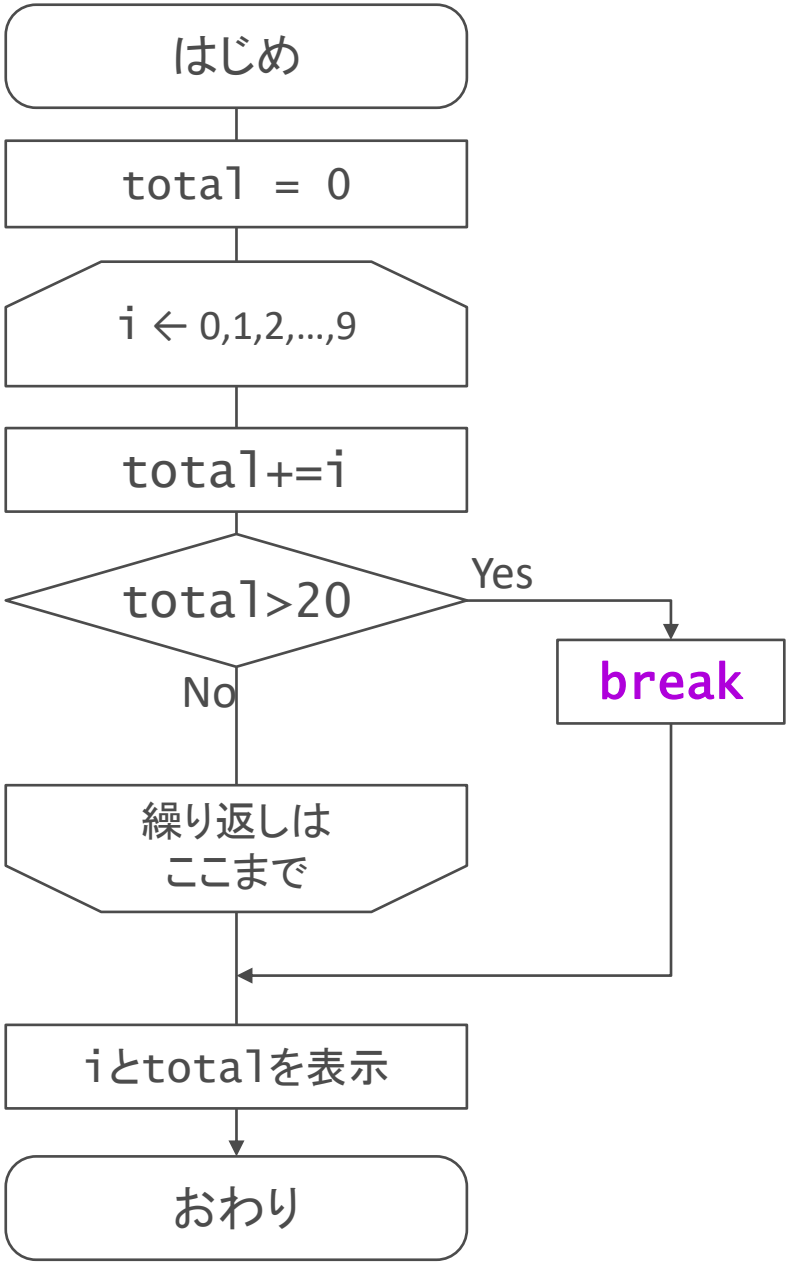
- 繰り返えされる処理内容を，途中で変更したい場合
- **break文**
  - 繰り返しを途中で中断
- **continue文**
  - continueが出たら，そのターンの繰り返し処理を終了
  - 繰り返し処理の最初に戻り，次の繰り返し処理を行う

- どちらも if文と組み合わせることが多い

# break文

```
total = 0
for i in range(10):
  total += i
  if total > 20:
    break

print(i, total)
```



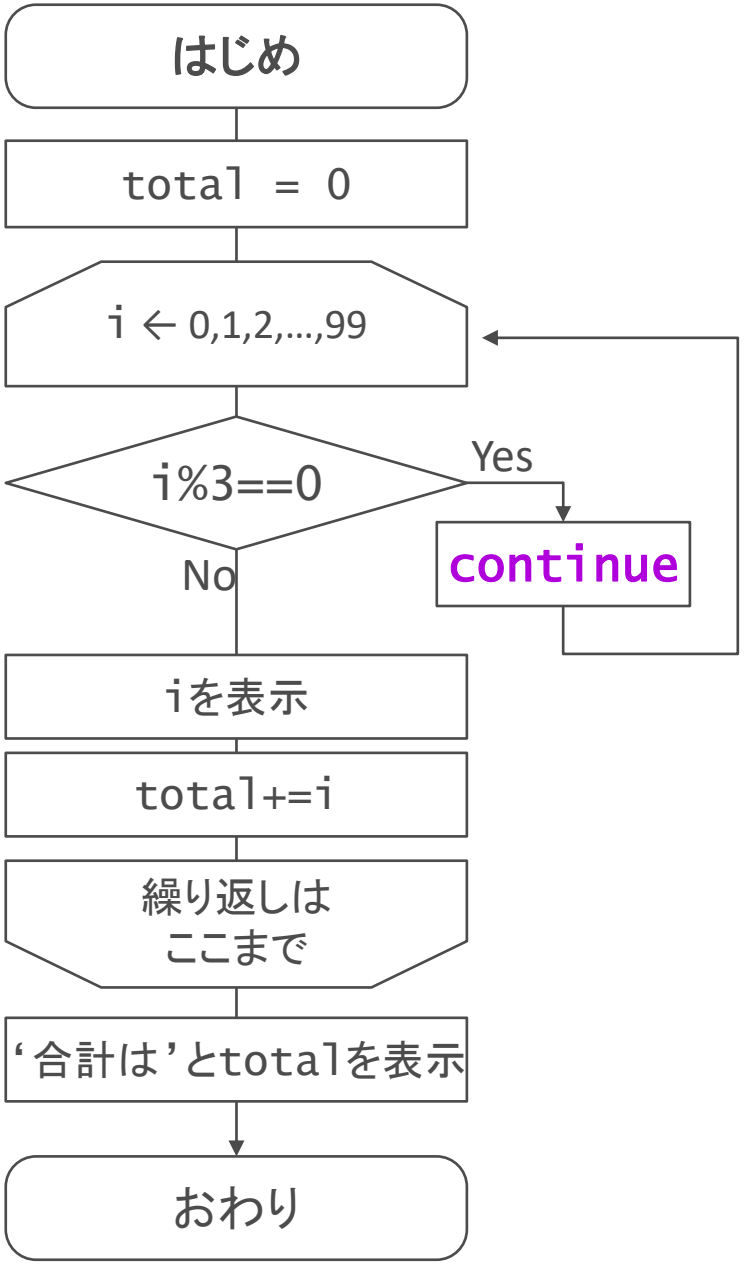
# continue文

ループ内の処理をスキップする

```
total = 0
for i in range(100):
  if i%3==0:
    continue
  print(i)
  total += i

print('合計は', total)
```

```
1
2
 :
97
98
合計は 3267
```



# 3.4 繰り返し処理の入れ子

```
for a in range(1,4):
  print('a=', a)
  for b in range(1,3):
    print('    b=', b)
```

for文の中に，for文が有る

```
a= 1
    b= 1
    b= 2
a= 2
    b= 1
    b= 2
a= 3
    b= 1
    b= 2
```

内側の繰り返し

内側の繰り返し

内側の繰り返し

外側の繰り返し

# 4. while文

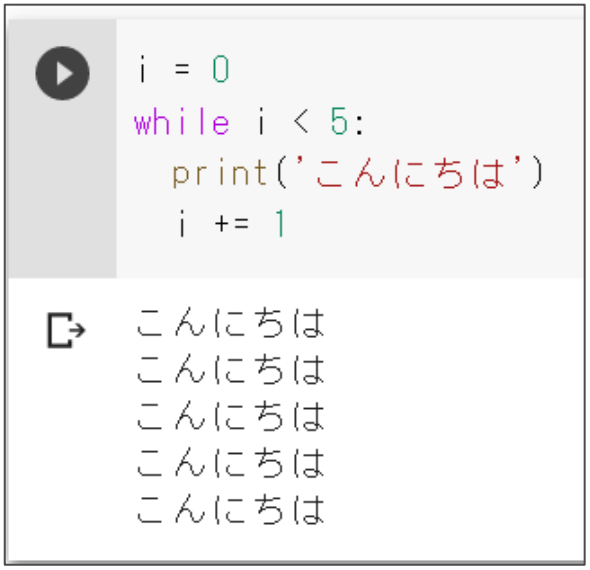
1. プログラムの制御構造（復習）
2. 繰り返し処理とは
3. for文
4. while文

# 4. while文

- 構文
  - 条件式の値が True の間、処理内容を繰り返す

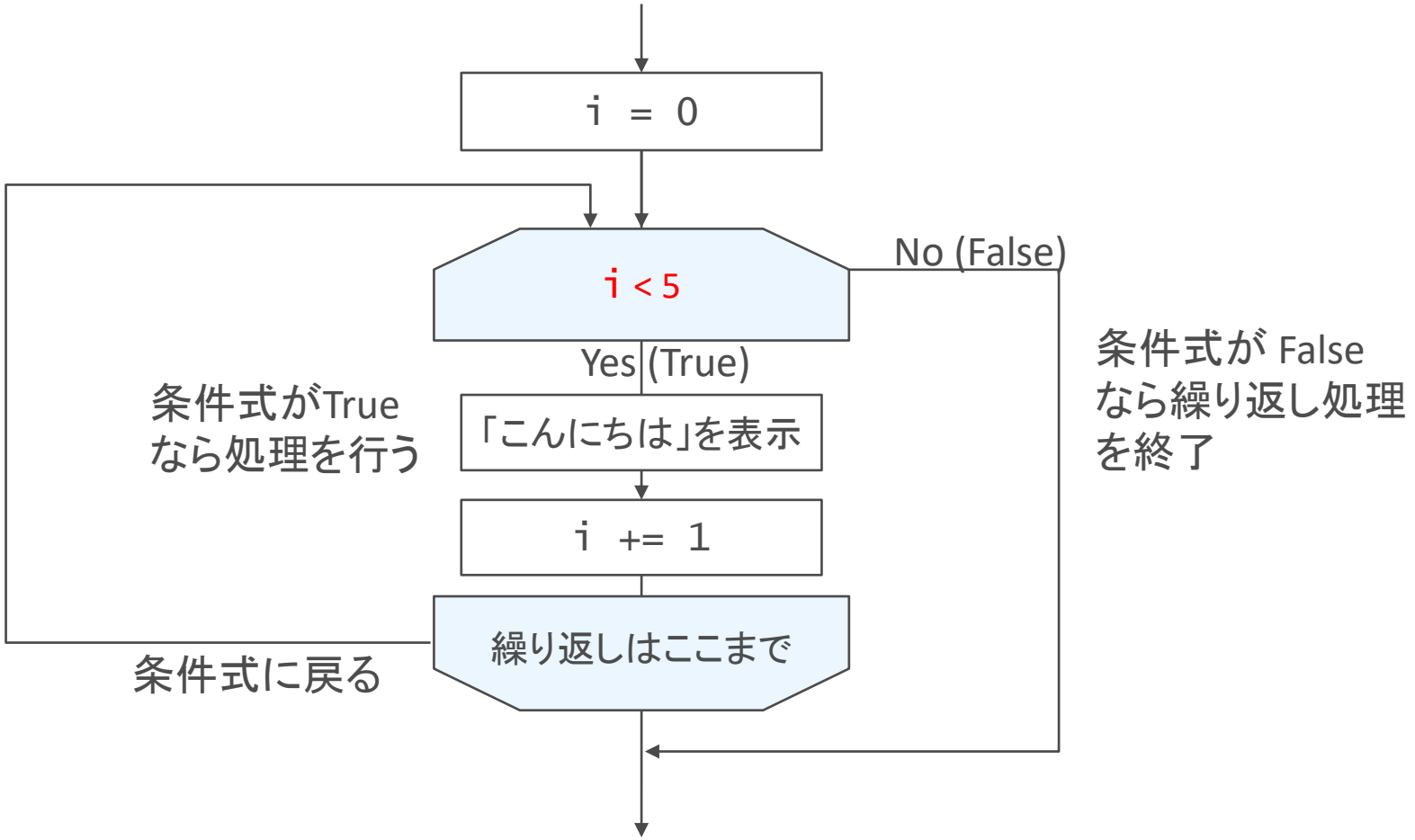
```
while 条件式:
    処理内容
```

```
i = 0
while i < 5:
  print('こんにちは')
  i += 1
```



# 処理の流れ

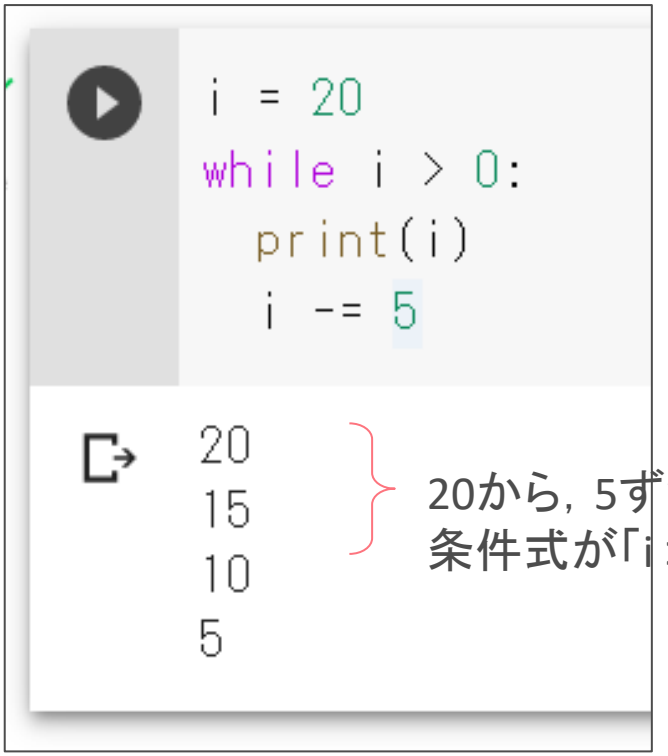
```
i = 0
while i < 5:
   print('こんにちは')
   i += 1
```



# while文の例

```
i = 20
while i > 0:
  print(i)
  i -= 5
```

変数 i の値が0より大きければ，次のブロックの処理を行う
　変数 i の値を出力
　i の値を5引いて，i を上書き




20から，5ずつ小さくなる値が表示
条件式が「i > 0」なので，iが0になると終了

# 5. 練習問題

1. プログラムの制御構造（復習）
2. 繰り返し処理とは
3. for文
4. while文
5. 練習問題

# 問題 1

- 10から20までの整数を順番に足し合わせて、その結果を出力するプログラムを作ってください。
- for文を使う場合と，while文を使った場合の2つのプログラムを作成してください。

# 問題 1 （解答）

- 10から20までの整数を順番に足し合わせて、その結果を出力するプログラムを作ってください。
- for文を使う場合と，while文を使った場合の2つのプログラムを作成してください。

for文

```
total = 0
for i in range(10, 21):
    total += i
print(total)
```

while文

```
total = 0
i = 10
while i < 21:
    total += i
    i += 1
print(total)
```

# 問題 2

- 問題1で作成したfor文を使ったプログラムコードに対して、15だけは足し合わせしないように変更してください。変更は continue 文を用いてください。

for文

```
total = 0
for i in range(10, 21):
    total += i
print(total)
```

# 問題 2 （解答）

- 問題1で作成したfor文を使ったプログラムコードに対して、15だけは足し合わせしないように変更してください。変更は continue 文を用いてください。

for文

```
total = 0
for i in range(10, 21):
    if i == 15:
        continue
    total += i
print(total)
```

# 問題 3

- 以下のコードは，リスト scores に格納されている要素のうち，値が60より大きい数が何個あるかを数えるプログラムの一部です。空欄部分を埋めて，プログラムを完成させてください。

```
scores = [65, 80, 40, 92, 76, 52]
count = 0  # 値が60より大きい要素の数
for i in scores:
    空欄

print(count)  # 結果を出力
```

# 問題 3 （解答）

- 以下のコードは，リスト scores に格納されている要素のうち，値が60より大きい数が何個あるかを数えるプログラムの一部です。空欄部分を埋めて，プログラムを完成させてください。

```
scores = [65, 80, 40, 92, 76, 52]
count = 0  # 値が60より大きい要素の数
for i in scores:
    if i>=60:
        count += 1

print(count)  # 結果を出力
```