



プログラミング言語 第五回

担当: 篠沢 佳久
栗原 聡

2019年5月13日



本日の内容

- 制御構造
- 条件文
 - 論理式(復習)
 - 条件文(if文)
- 繰り返し(1)
 - 回数の決まった繰り返し
- 練習問題



提出物について

- 提出されたファイル中, コピーされたと判断されるものがあります.
- レポートのコピーは厳禁です. 絶対にコピーはしないで下さい.



条件文

論理式(復習)

条件文(if文)



プログラムの構造

- 基本的には上から順に実行される

```
print( "一番目の数値を入力してください: " )  
line = input()  
x1 = float( line )  
print( "二番目の数値を入力してください: " )  
line = input()  
x2 = float( line )  
print( x1 , "+" , x2 , "=" , x1+x2 , sep=' ' )
```

実行順番

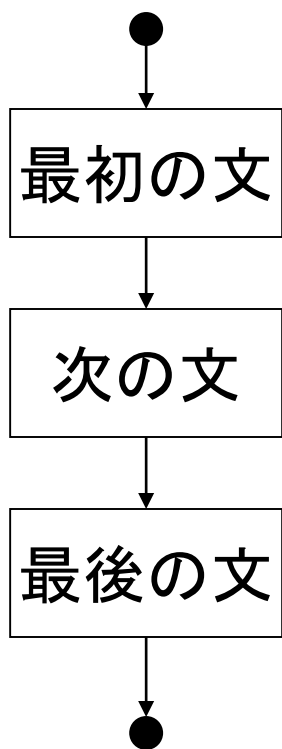


- 実行順番を変えることも必要
 - 制御構造

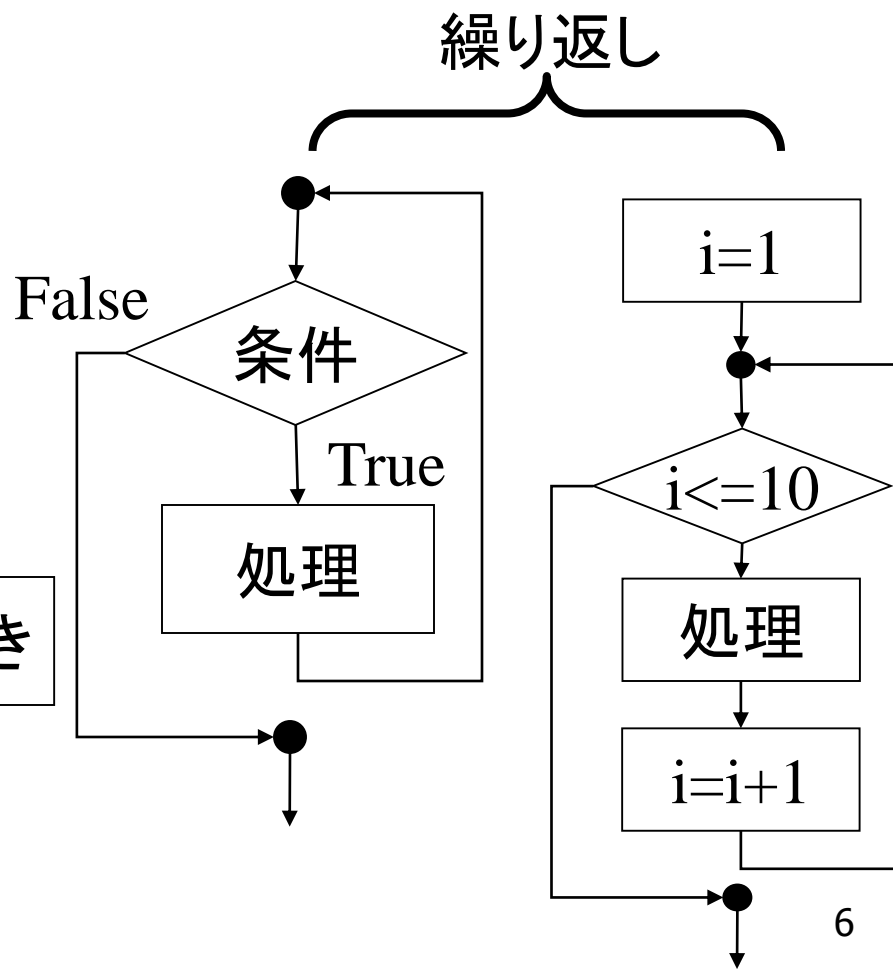
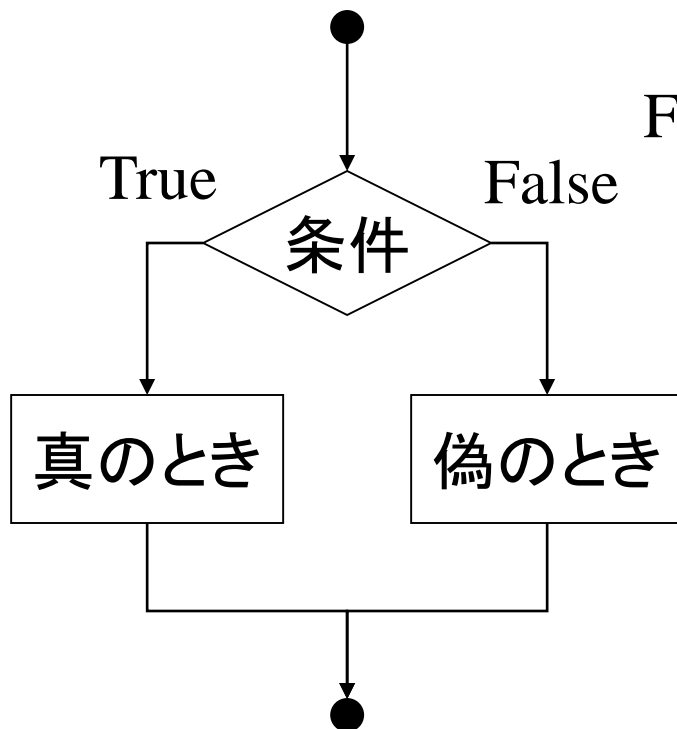
制御構造

流れ図(フローチャート)

文の並び(接続)



If文(分岐)





条件文 (if文)

プログラムを書いている際には

- ある論理式がTrueの場合には，処理Aを行なう
 - Falseの場合には処理Bを行なう
- という要求が発生する



条件文 (if文) の種類①

① if 論理式:

文でも可

式1

② if 論理式:

式1

else:

式2

③ if 論理式0:

if 論理式1:

式11

else:

式12

else:

if 論理式2:

式21

else:

式22



条件文 (if文) の種類②

④ if 論理式1:

式1 # 論理式1がTrueのときの値

elif 論理式2:

式2 # 論理式2がTrueのときの値

elif 論理式3:

式3 # 論理式3がTrueのときの値

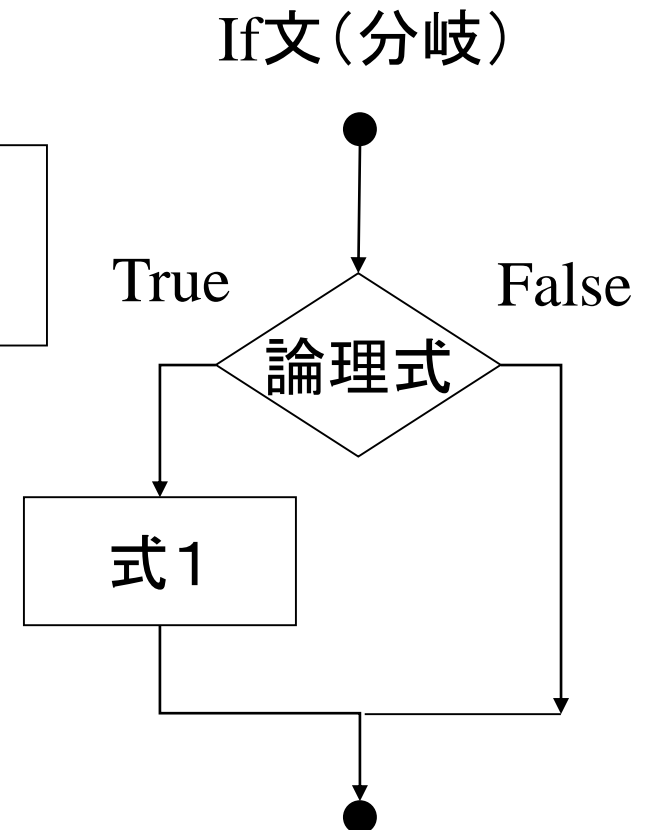
else:

式4 # 論理式1,2,3がFalseのときの値

if文①

if 論理式:

式1 # 論理式がTrueのときの値





if文①(例)

```
x = 5
```

```
if x < 0:  
    print( x )
```

```
x = 0
```

```
if x < 0:  
    print( x )
```

```
x = -5
```

```
if x < 0:  
    print( x )
```

print(x) が実行されるのは？



if文①(例)

(print文が実行されるか考えなさい)

```
x = -5  
if x != 0:  
    print( x )
```

```
x = -5  
y = -10  
if x < 0 and y < 0:  
    print( x )
```

```
x = -5  
if not x == 0:  
    print( x )
```

```
x = -5  
y = 10  
if x < 0 or y < 0:  
    print( x )
```

if文の構造①

```
a=5
```

```
if a > 0:
```

```
    if a < 10:
```

```
        print( a )
```

if文中にif文を書くことも可能
「aが0より大きい」かつ
「aが10より小さい」

同じif文

```
a=5
```

```
if a > 0 and a < 10:
```

```
    print( a )
```

if文の構造②

インデントが重要

○

```
if a > 0:
    if a < 10:
        print( a )
```

○

```
if a > 0 and a < 10:
    print( a )
```

×

```
if a > 0:
if a < 10:
    print( a )
```

×

```
if a > 0 and a < 10:
print( a )
```

if文の構造③

インデントが重要

```
if a > 0:
```

```
    if a < 10:
```

```
        print( a )
```

```
if a > 0 and a < 10:
```

```
    print( a )
```

「慣例的」に半角空白4個空ける

if文の構造③

```
a=5
```

```
if a > 0:
```

```
    if a < 10:
```

```
        x = a*10
```

```
    if a >= 10:
```

```
        x = a*100
```

```
print( x )
```

if文中にif文を書くことも可能

aが10より小さい

aが10以上

if文の構造④

同じ動作のプログラム

```
a=5
```

```
if a > 0:
```

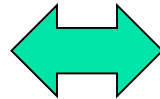
```
    if a < 10:
```

```
        x = a*10
```

```
    if a >= 10:
```

```
        x = a*100
```

```
print( x )
```



```
a=5
```

```
if a > 0:
```

```
    if a < 10:
```

```
        x = a*10
```

```
if a > 0:
```

```
    if a >= 10:
```

```
        x = a*100
```

```
print( x )
```

if文①(例)

(aの値を考えなさい)

```
a = ?  
if a > 0:  
    if a > 10:  
        a += 1  
    a += 1  
print( a )
```

aの値はどうなるでしょうか

a = -100

a = 0

a = 1

a = 10

a = 100

a += 1 は
a = a + 1 を計算



論理式(復習)

- 論理値(真偽値)を計算する式を論理式と呼ぶことにします
- 論理式の基本は、数式または文字列式(の値)と数式または文字列式(の値)とを比較演算子を用いて比較する式です。これを and/or/not で組み合わせます



比較演算子

演算子	用途	例	演算結果
==	等	3==2	False
>	大	4 > 2	True
<	小	4 < 2	False
>=	大 or 等	4>=2	True
<=	小 or 等	4<=2	False
!=	非等	3!=2	True
in	含まれる	"x" in "xyz"	True



論理演算子

演算子	用途	例	演算結果
not	否定	not 3==2	True
and	かつ	2==2 and 4>2	True
or	または	2==3 or 4>2	True

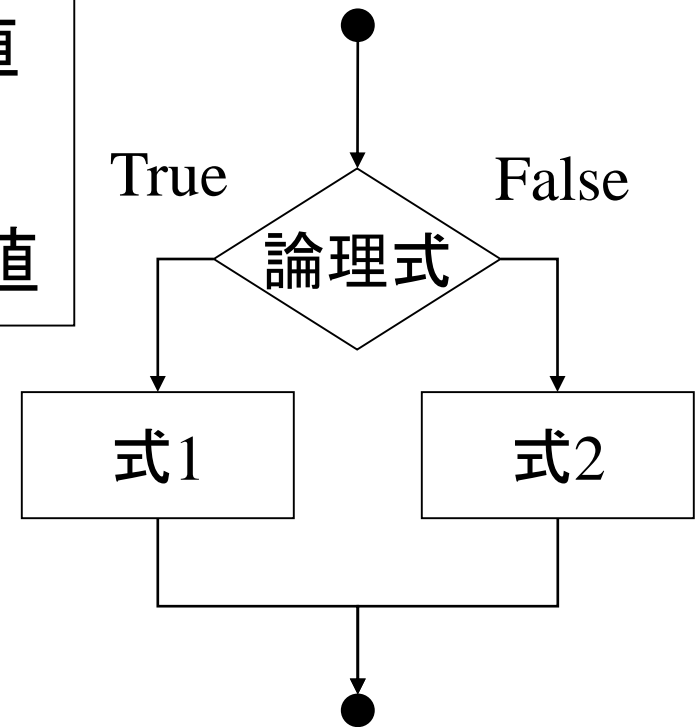
if文②

if 論理式:

式1 # 論理式がTrueのときの値

else:

式2 # 論理式がFalseのときの値



if文(例)

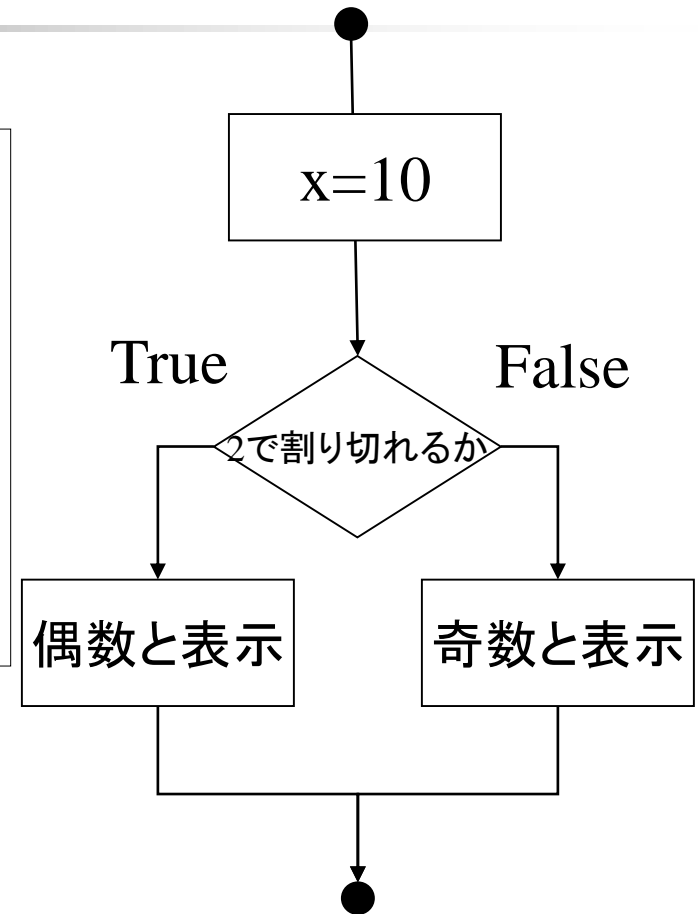
```
x=10
```

```
if x % 2 == 0:
```

```
    print( x , "は偶数です" )
```

```
else:
```

```
    print( x , "は奇数です" )
```



if文②(例)

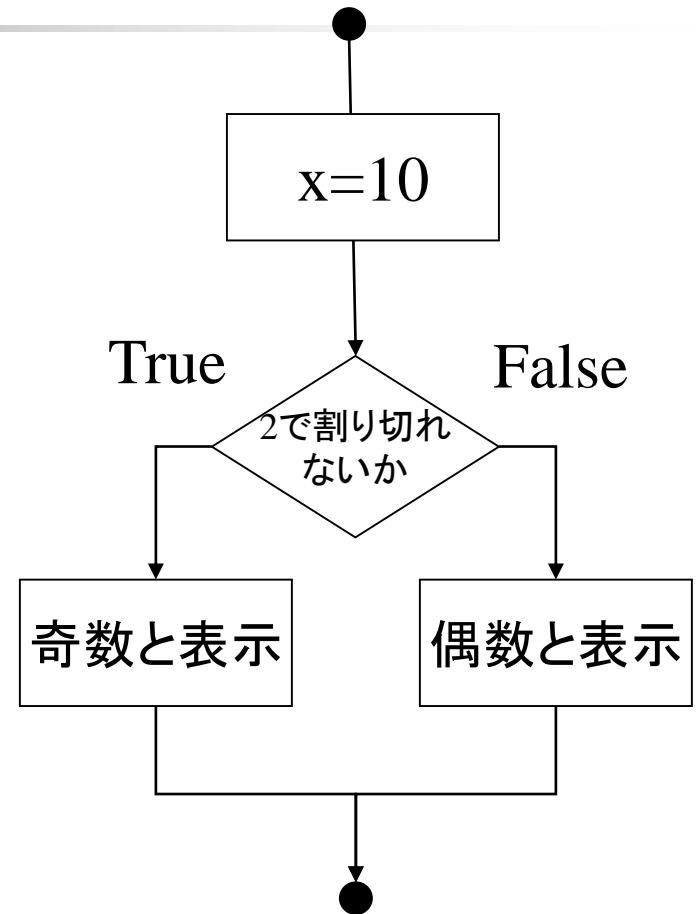
```
x=10
```

```
if x % 2 != 0:
```

```
    print( x , "は奇数です" )
```

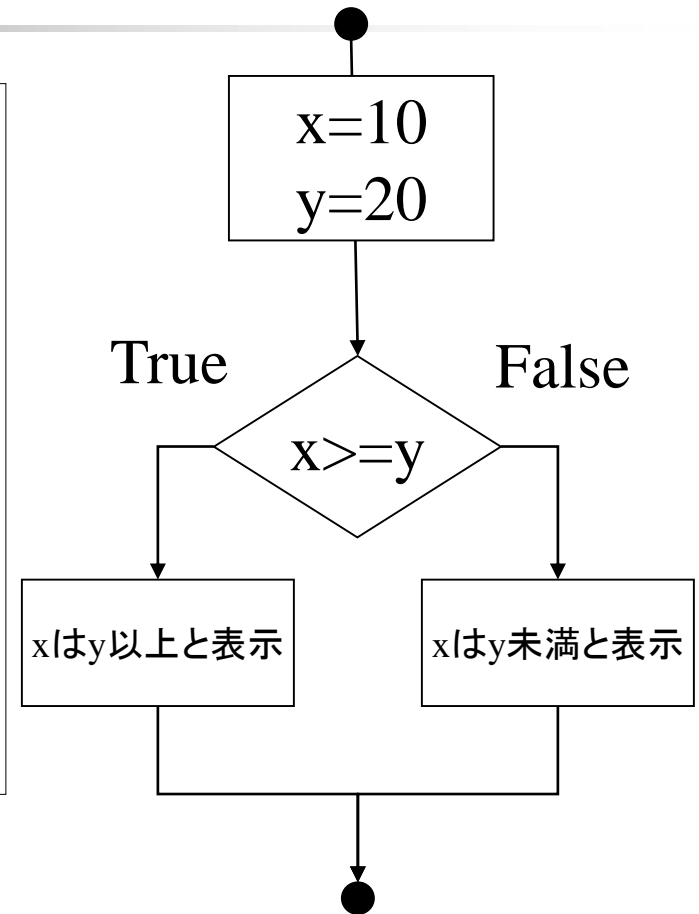
```
else:
```

```
    print( x , "は偶数です" )
```



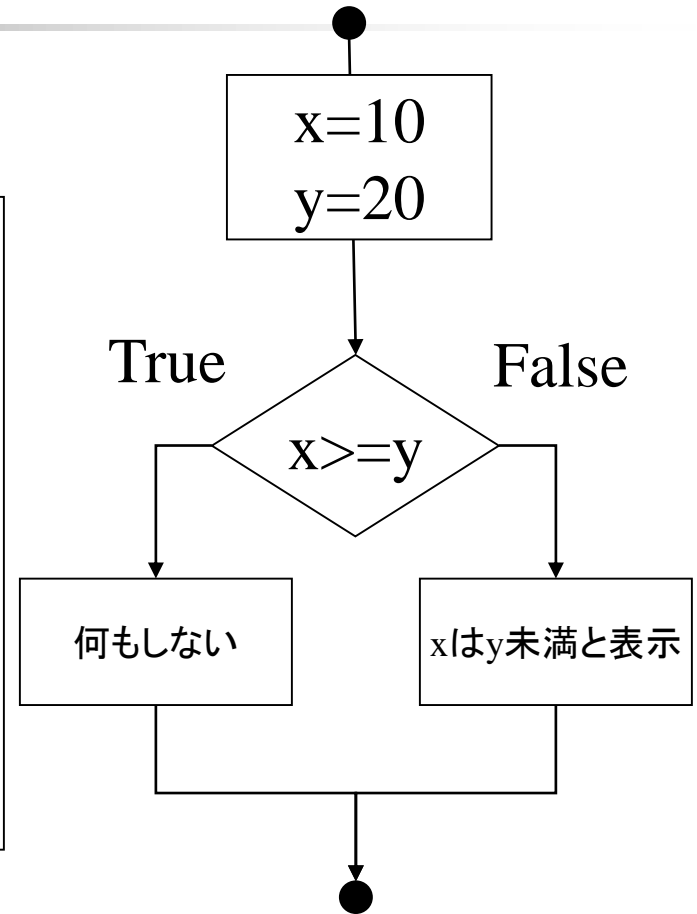
if文②(例)

```
x=10
y=20
if x >=y:
    print( x , "は" , y , "以上" )
else:
    print( x , "は" , y , "未満" )
```



if文②(例)

```
x = 10
y = 20
if x >= y:
    pass      #何もしない
else:
    print( x , "は" , y , "未満" )
```



if文③

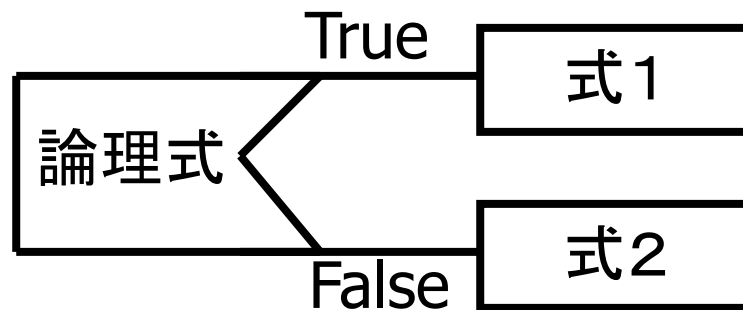
if 論理式:

式1

else:

式2

正式にはPAD図といいます
ここでは分岐図とでもいいましょうか



if文でさらに分岐できるか
→ もちろん分岐できます

if文③

if 論理式0:

if 論理式1:

式11

else:

式12

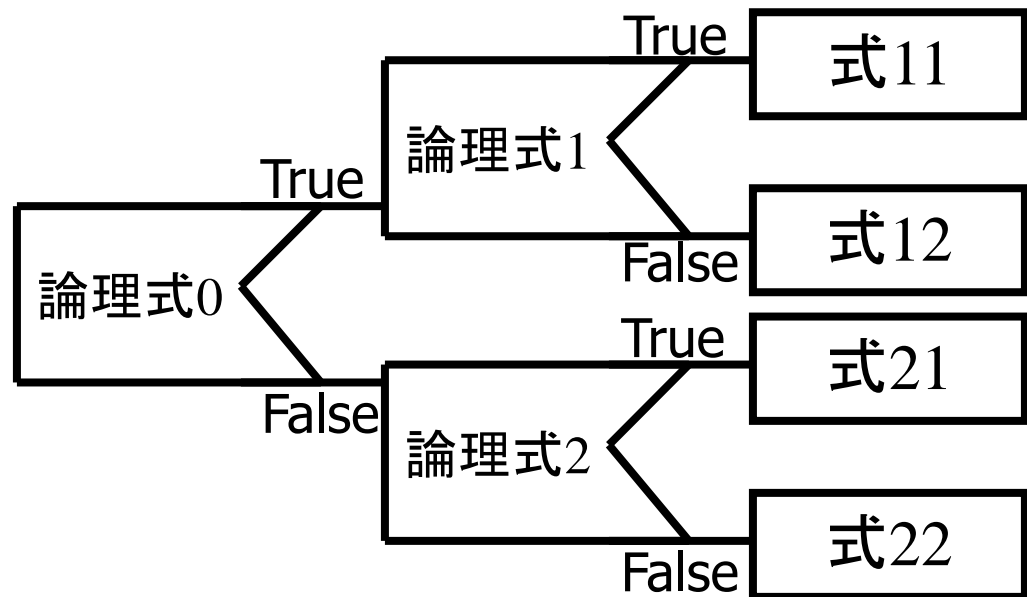
else

if 論理式2:

式21

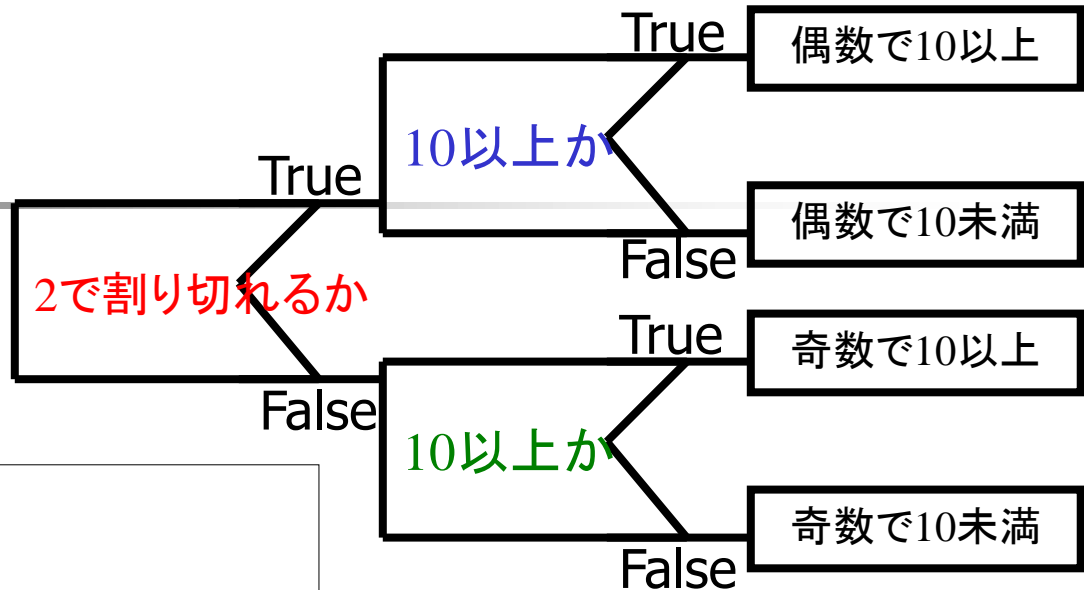
else:

式22



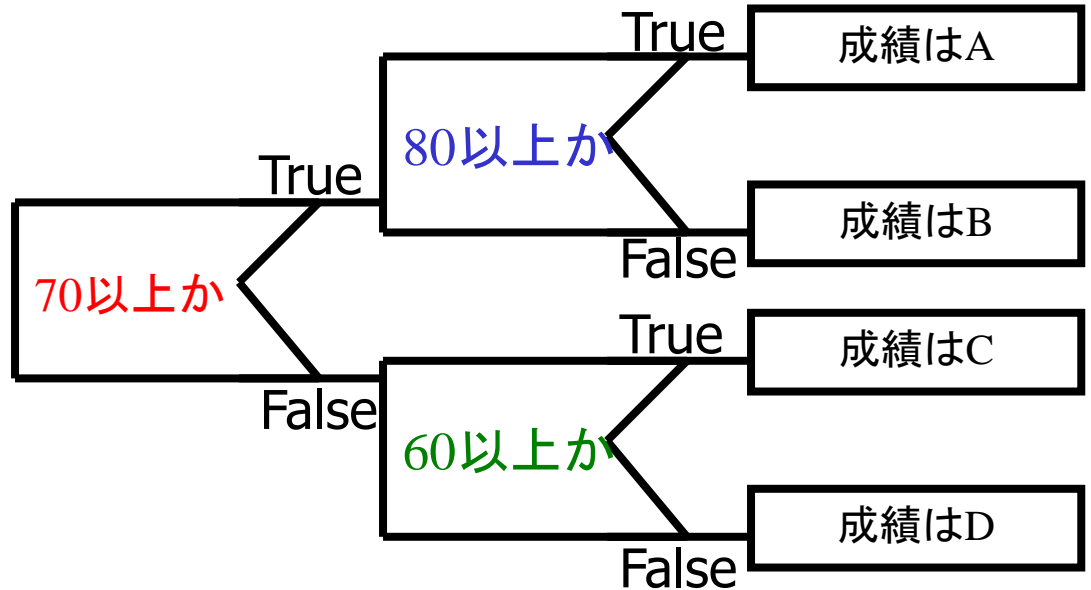
if文③(例)

```
x=10
if x % 2 == 0:
    if x >= 10:
        print( x , "は偶数で10以上" )
    else:
        print( x , "は偶数で10未満" )
else:
    if x >= 10:
        print( x , "は奇数で10以上" )
    else:
        print( x , "は奇数で10未満" )
```



if文③(例)

```
score = 75
if score >= 70:
    if score >= 80:
        grade = "A"
    else:
        grade = "B"
else:
    if score >= 60:
        grade = "C"
    else:
        grade = "D"
print( "Your score" , score , "corresponds to" , grade )
```





if文③(例)

```
a = ?  
if a > 10:
```

```
    if a > 100:  
        a += 1  
    else:  
        a -= 1
```

```
else:
```

```
    if a > -10:  
        a += 10  
    else:  
        a -= 10
```

```
print( a )
```

aの値はどうなるでしょうか

a = -100

a = -10

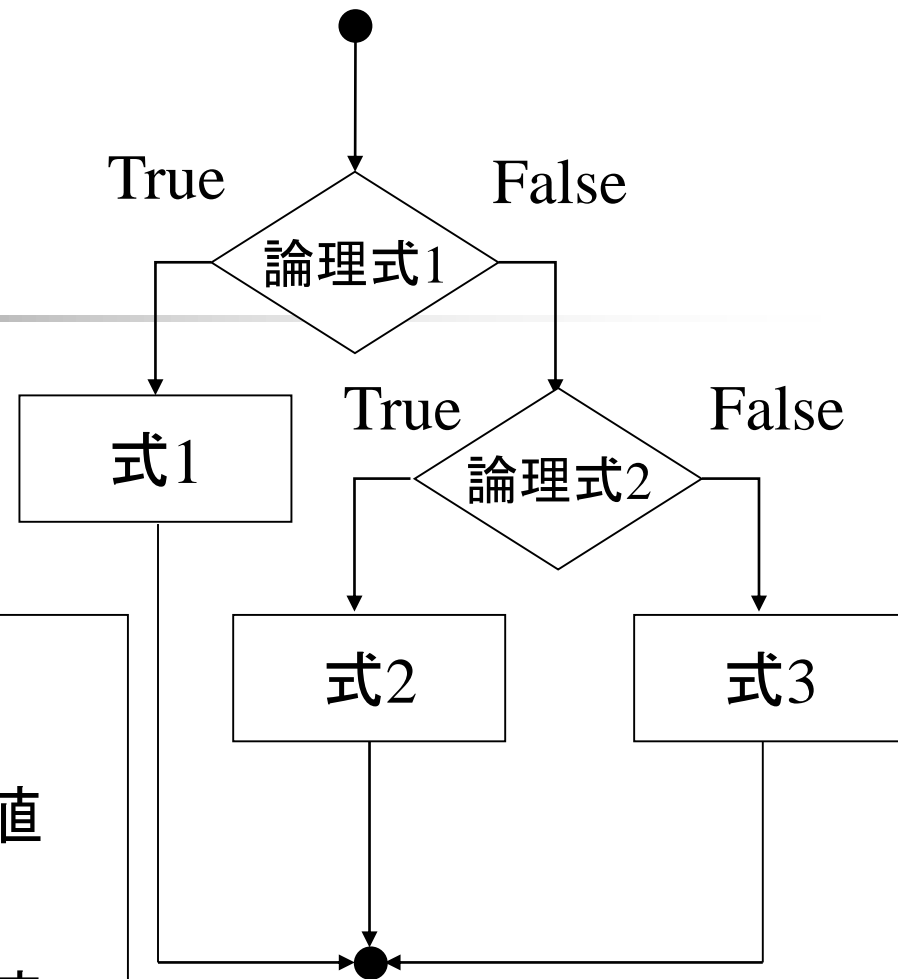
a = 0

a = 10

a = 100

a = 1000

if文④



if 論理式1:

式1 # 論理式1がTrueのときの値

elif 論理式2:

式2 # 論理式2がTrueのときの値

else:

式3 # 論理式1,2がFalseのときの値

if文④(例)

x=5

if x >= 10:

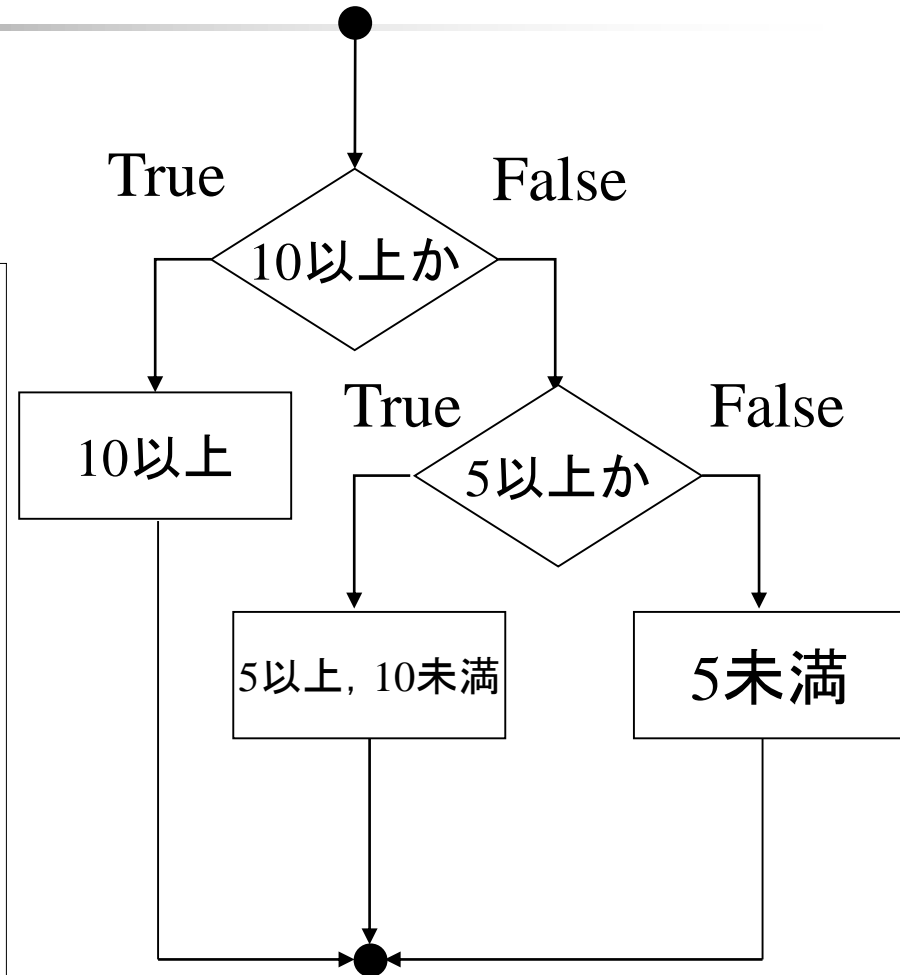
print(x , "は10以上")

elif x >= 5:

print(x , "は5以上, 10未満")

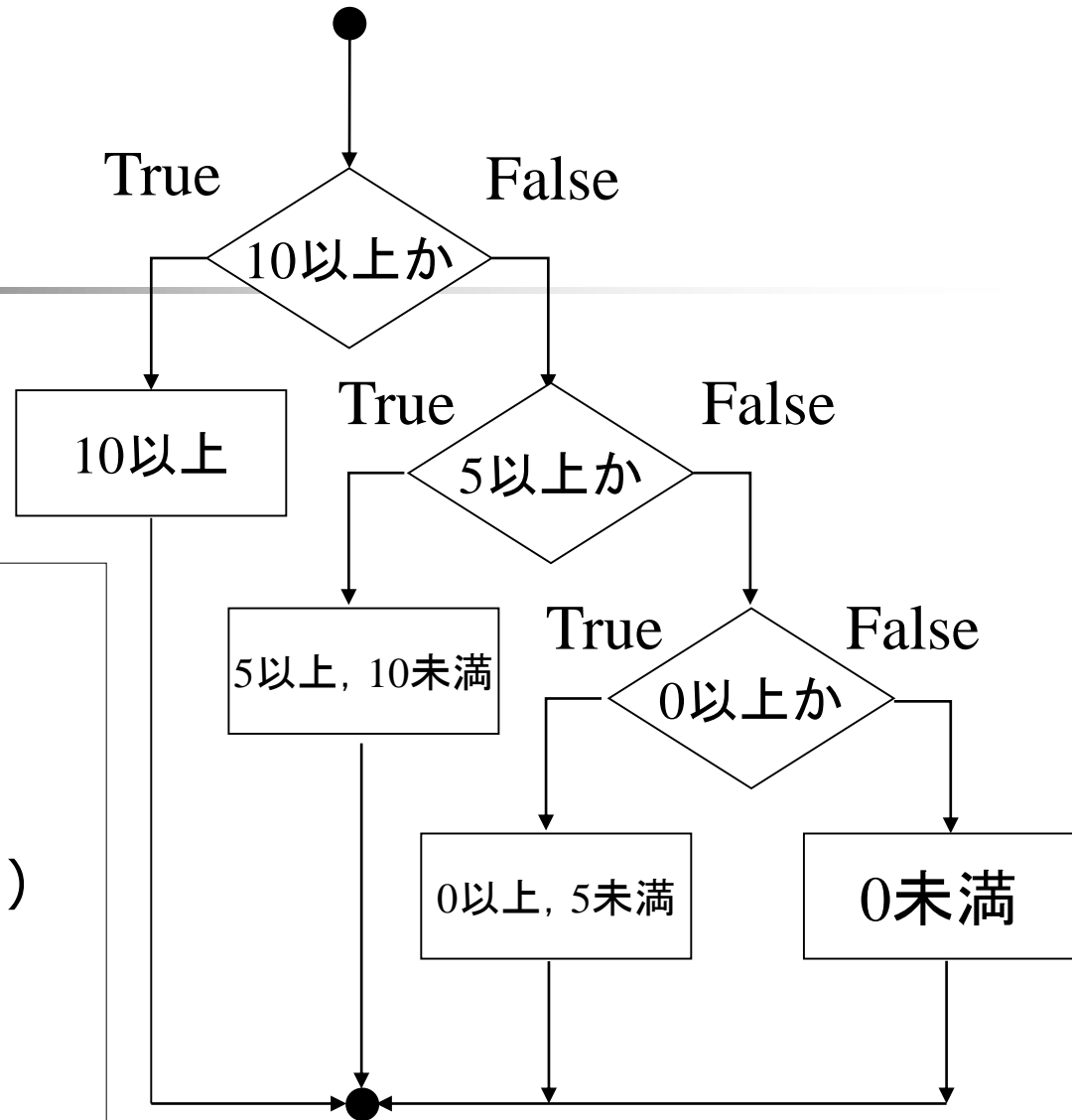
else:

print(x , "は5未満")



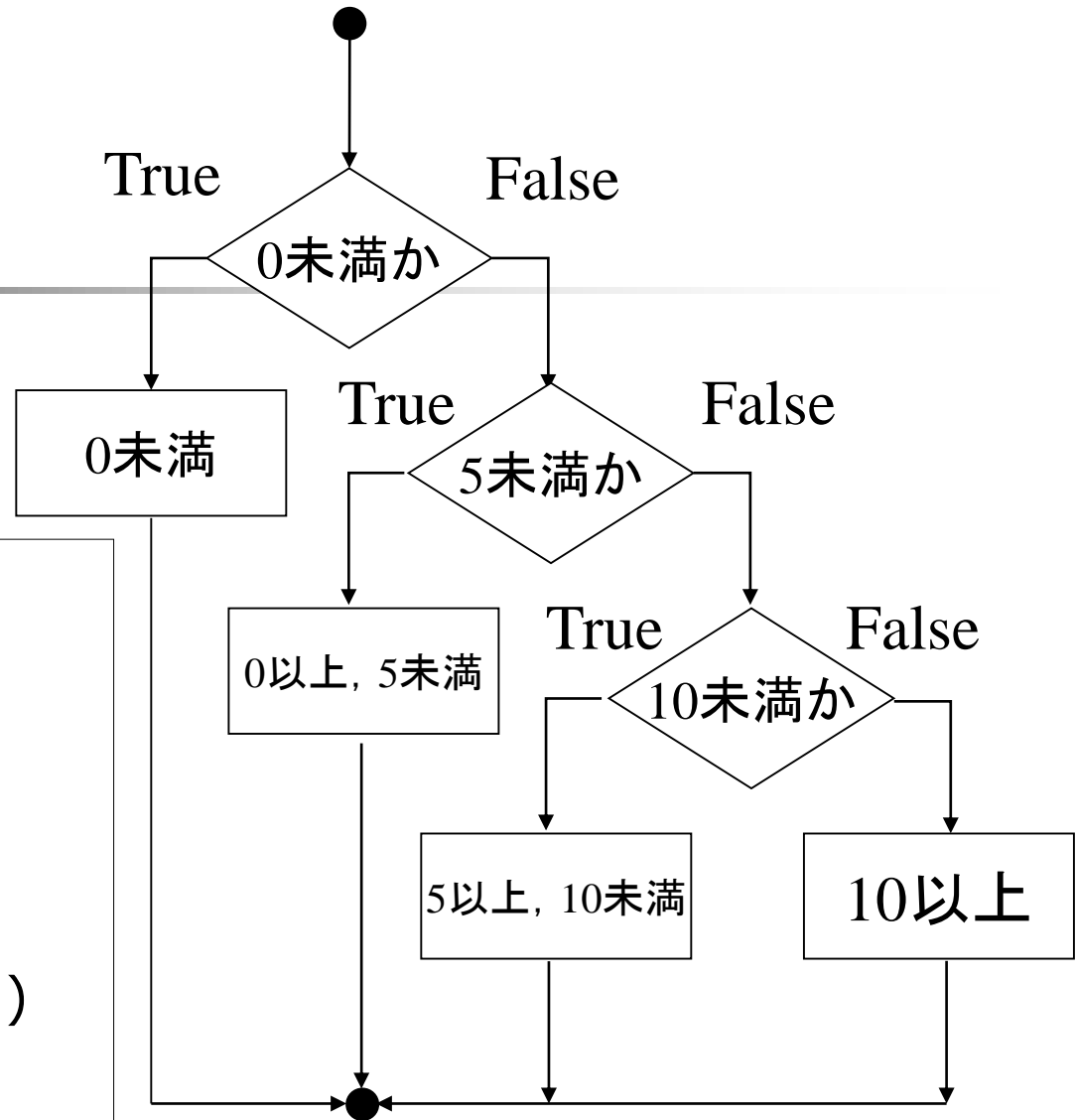
if文④(例)

```
x = 5
if x >= 10:
    print( x , "は10以上" )
elif x >= 5:
    print( x , "は5以上, 10未満" )
elif x >= 0:
    print( x , "は0以上, 5未満" )
else:
    print( x , "は0未満" )
```



if文④(例)

```
x=5
if x < 0:
    print( x , "は0未満" )
elif x < 5:
    print( x , "は0以上, 5未満" )
elif x < 10:
    print( x , "は5以上, 10未満" )
else:
    print( x , "は10以上" )
```



前のページと同じプログラムです



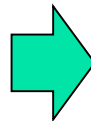
読みやすいプログラム

- プログラムは文法的に間違えていなければ動きます
- しかし、読みやすいプログラムを書くことを心がけましょう
- インデント(字下げ)をすることによって、分かりやすいプログラムになります
- pythonは条件文、繰り返し(for文, while文)において適切にインデントを用いない場合、エラーとなります



インデント①

```
x = 5  
y = -10  
if x+y < 0:  
print( x )  
print( y )
```



```
x = 5  
y = -10  
if x+y < 0:  
    print( x )  
    print( y )
```

x+y < 0 の場合, 二つの文が実行されると直感的にも分かる

インデント②

```
x = 5
y = -10
if x+y < 0:
```

```
    print( x )
    print( y )
```

$x+y < 0$ の場合、二つの
print文が実行

```
x = 5
y = -10
if x+y < 0:
```

```
    print( x )
```

```
print( y )
```

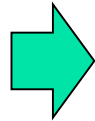
$x+y < 0$ の場合、
print(x)が実行

print(y)はif文によらず必ず実行

インデント③

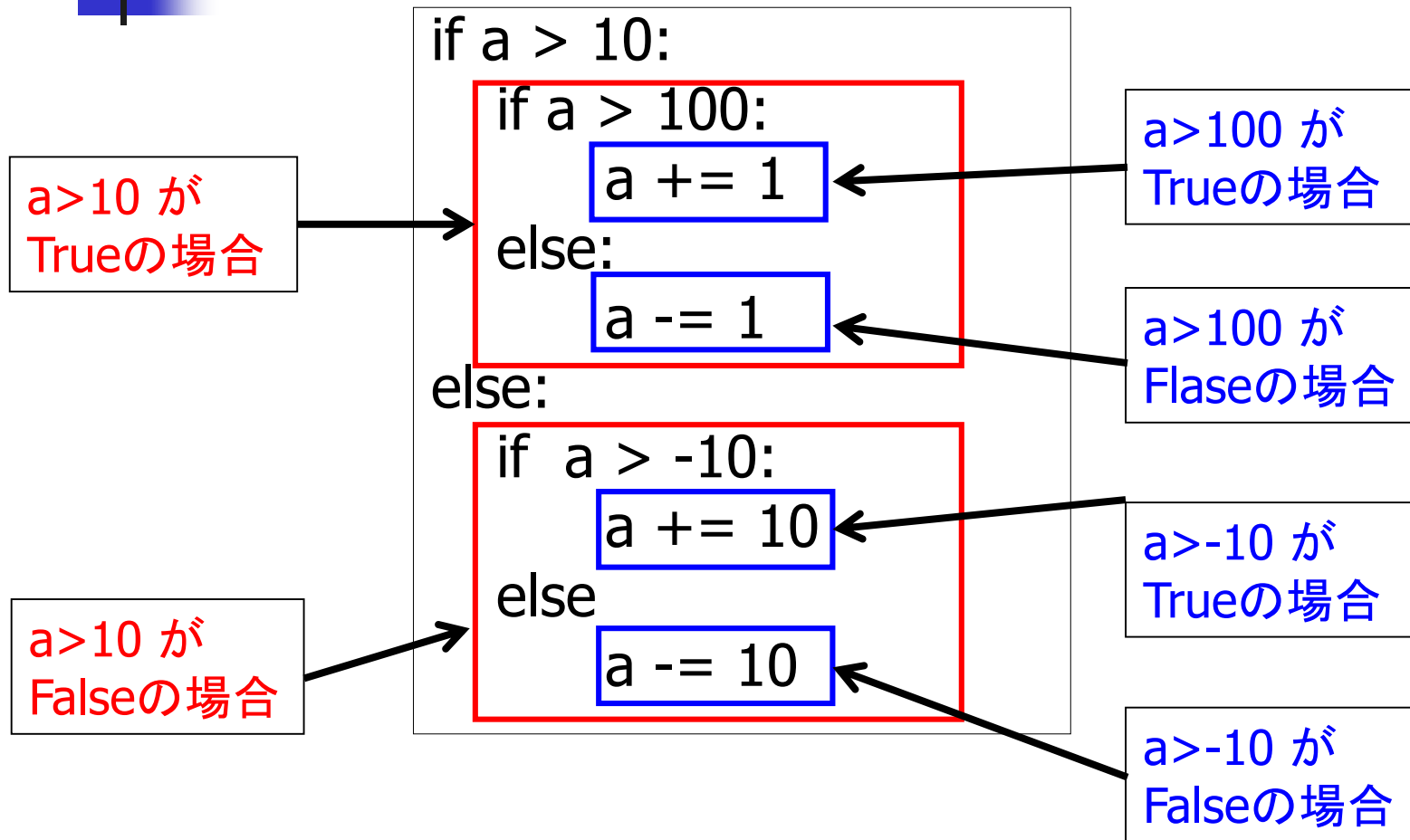
左のプログラムと比べて条件文によってどこか実行されるか理解しやすい

```
if a > 10:  
if a > 100:  
    a += 1  
else:  
    a -= 1  
else:  
if a > -10:  
    a += 10  
else:  
    a -= 10
```



```
if a > 10:  
    if a > 100:  
        a += 1  
    else:  
        a -= 1  
else:  
    if a > -10:  
        a += 10  
    else:  
        a -= 10
```

インデント③





インデントのつけ方の例①

空白4個空ける

```
if a > 10:  
    if a > 100:  
        a += 1  
    else:  
        a -= 1  
else:  
    if a > -10:  
        a += 10  
    else:  
        a -= 10
```

インデントのつけ方の例②

空白8個
空ける

```
if a > 10:
    if a > 100:
        a += 1
    else:
        a -= 1
else:
    if a > -10:
        a += 10
    else:
        a -= 10
```



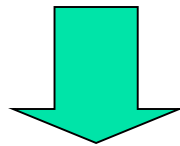
回数の決まった繰り返し

for文



繰り返しの必要性①

- 1から10の整数を出力したい
- `print("1 2 3 4 5 6 7 8 9 10")`



- 1から1000の整数を出力したい
- `print("1 2 ... 1000")`
 - と書けばよいが...



繰り返しの必要性②

プログラムを書いている際には、

- 同じ処理をある回数だけ行ないたい
- ある条件が成立するまで、同じ処理を繰り返したい
- 値を変化させながら、同じ処理を繰り返したい

という要求が発生する



繰り返しの必要性③

- 繰り返しを行なう際に考えること
 - 何を繰り返すのか
 - 何回繰り返しを行なうのか
 - どのような条件で繰り返しを停止するのか



繰り返しの必要性④

- 1から1000の整数を出力したい
- 何を繰り返すのか
→出力を繰り返す
- どういう条件で繰り返しを停止するのか
→1000まで出力したら停止する



回数の決まった繰り返し①

```
for 変数 in range(繰り返し回数):  
    式
```

```
for i in range(5):  
    print( "python" )
```

pythonを5回表示するプログラム

```
>python example-1.py  
python  
python  
python  
python  
python
```




回数の決まった繰り返し②

「こんにちは」を10回表示させるプログラム

```
for i in range(10):  
    print( i , "回目のこんにちは" )
```

iには0,1,2,...,9が代入される

```
>python example-1.py
```

```
0 回目のこんにちは  
1 回目のこんにちは  
2 回目のこんにちは  
3 回目のこんにちは  
4 回目のこんにちは  
5 回目のこんにちは  
6 回目のこんにちは  
7 回目のこんにちは  
8 回目のこんにちは  
9 回目のこんにちは
```

回数の決まった繰り返し③

「こんにちは」を10回表示させるプログラム

```
for i in range(10):  
    print( i , "回目のこんにちは" )
```

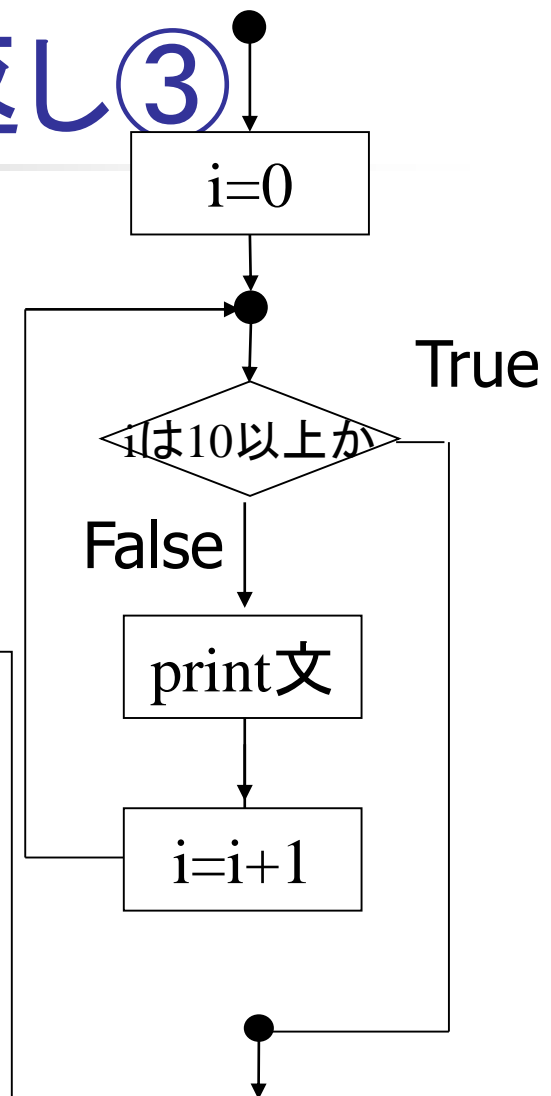
① $i=0$

② i は10以上ではないのでprint文を実行

③ $i=i+1 \rightarrow i=1$

④ i は10以上ではないのでprint文を実行

最後, i が10以上になったらprint文を実行せずに終了

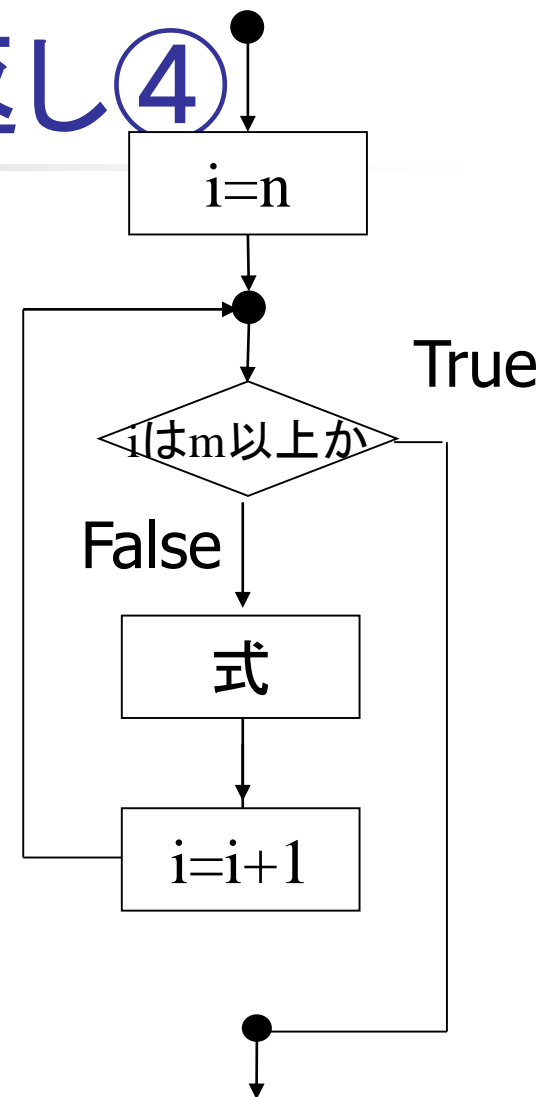


回数の決まった繰り返し④

```
for i in range(n,m):  
    式
```

($m-n-1$)回, 式が繰り返される

変数*i*には*n*から*m*-1まで代入される





回数の決まった繰り返し⑤

「こんにちは」を10回表示させるプログラム

```
for i in range(1,11):  
    print( i , "回目のこんにちは" )
```

iには1,2,3,...,10が代入される

```
>python sample.py  
1 回目のこんにちは  
2 回目のこんにちは  
3 回目のこんにちは  
4 回目のこんにちは  
5 回目のこんにちは  
6 回目のこんにちは  
7 回目のこんにちは  
8 回目のこんにちは  
9 回目のこんにちは  
10 回目のこんにちは
```



回数の決まった繰り返し⑥

「こんにちは」を10回表示させるプログラム

```
for i in range(5,16):  
    print( i , "回目のこんにちは" )
```

iには5,6,7,...,15が代入される

```
>python sample.py  
5 回目のこんにちは  
6 回目のこんにちは  
7 回目のこんにちは  
8 回目のこんにちは  
9 回目のこんにちは  
10 回目のこんにちは  
11 回目のこんにちは  
12 回目のこんにちは  
13 回目のこんにちは  
14 回目のこんにちは  
15 回目のこんにちは
```

繰り返しの例①

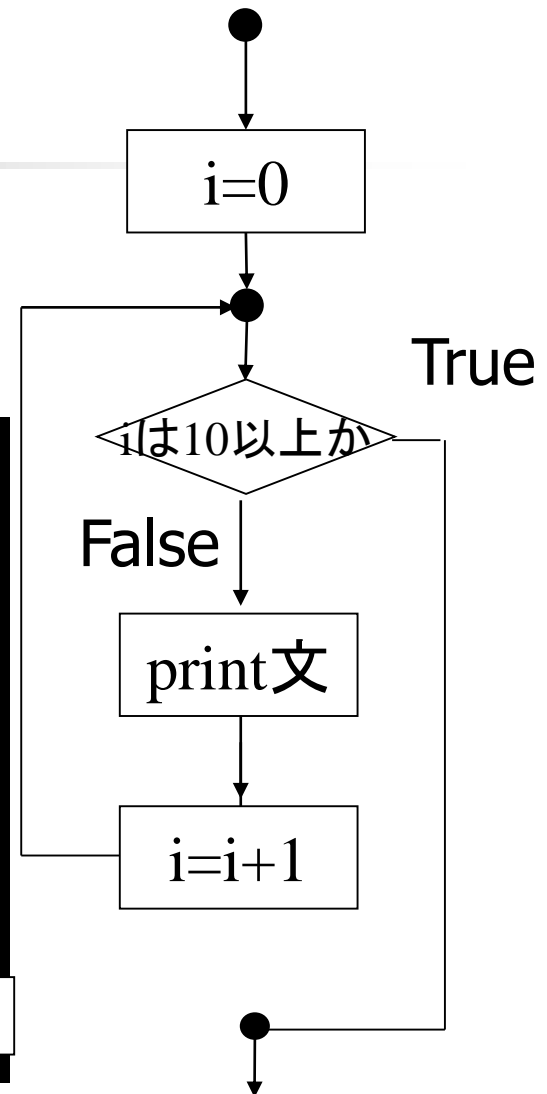
10のべき乗を表示するプログラム

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

```
for i in range(0,10):  
    print( 10**i )
```

iには0,1,2,...,9が代入される

```
1 i=0  
10 i=1  
100 i=2  
1000  
10000  
100000  
1000000  
10000000  
100000000  
1000000000 i=9
```



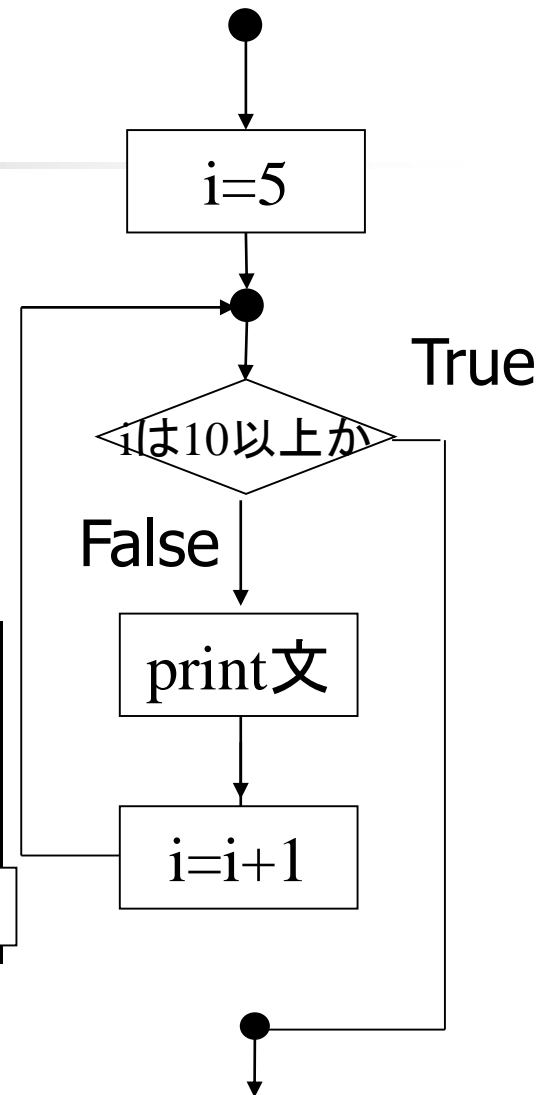
繰り返しの例②

10のべき乗を表示するプログラム

```
for i in range( 5,10 ):
    print( 10**i )
```

iには5,6,7,8,9が代入される

100000	i=5
1000000	i=6
10000000	
100000000	
1000000000	i=9



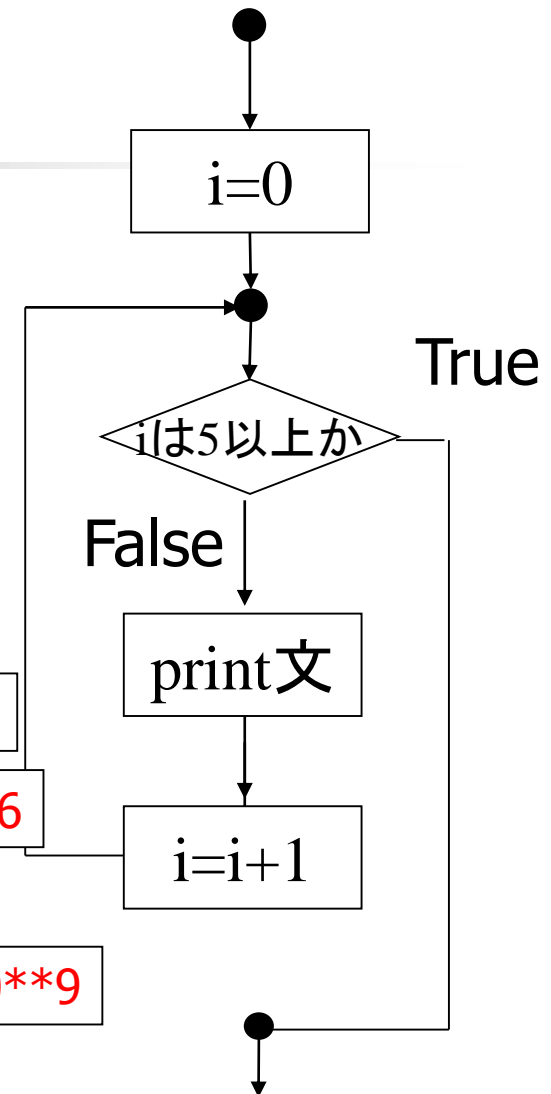
繰り返しの例②'

10のべき乗を表示するプログラム

```
for i in range( 5 ):
    print( 10**(i+5) )
```

iには0,1,2,3,4が
代入される

100000	i=0	10**5
1000000	i=1	10**6
10000000		
100000000		
1000000000	i=4	10**9



繰り返しの例③

10のべき乗を表示するプログラム

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

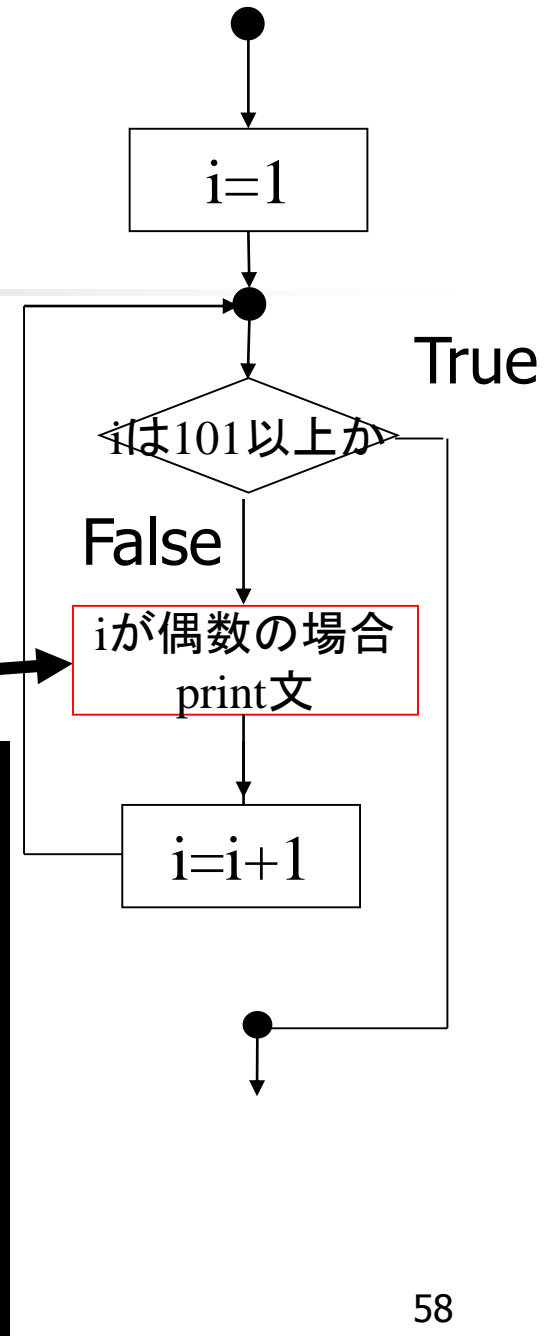
iには0,1,2,...,9が代入される

10000000000	i=0	10**10
1000000000	i=1	10**9
100000000		
10000000		
1000000		
100000		
10000		
1000		
100	i=8	10**2
10	i=9	10**1

繰り返しの例④

100以下の偶数を表示するプログラム

```
for i in range(1,101):  
    if i % 2 == 0:  
        print( i )
```



2 `i=2`
4 `i=4`
6
8
10
12

98
100 `i=100`

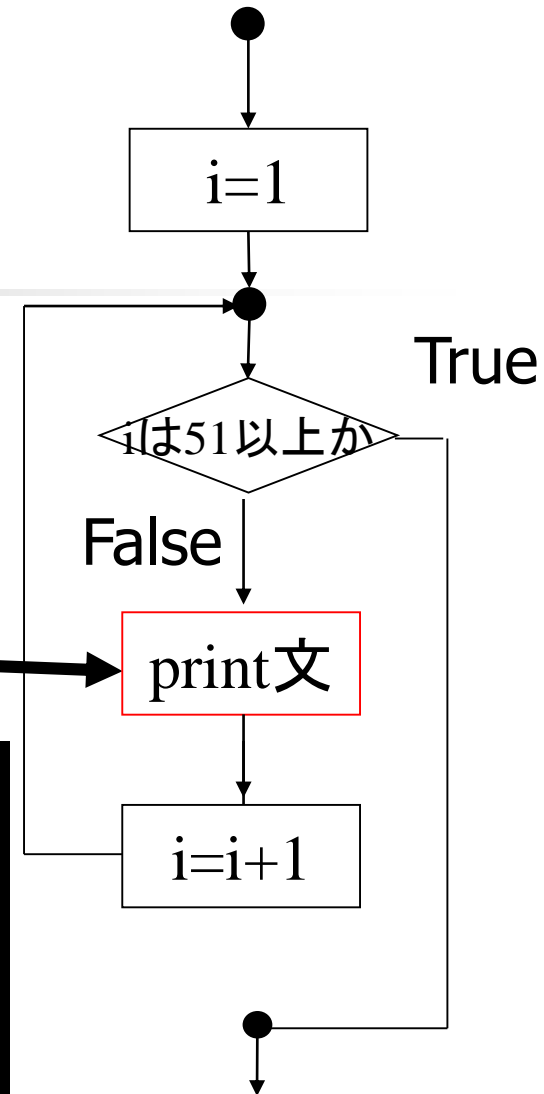
繰り返しの例④'

100以下の偶数を表示するプログラム

```
for i in range(1,51):  
    print( i*2 )
```

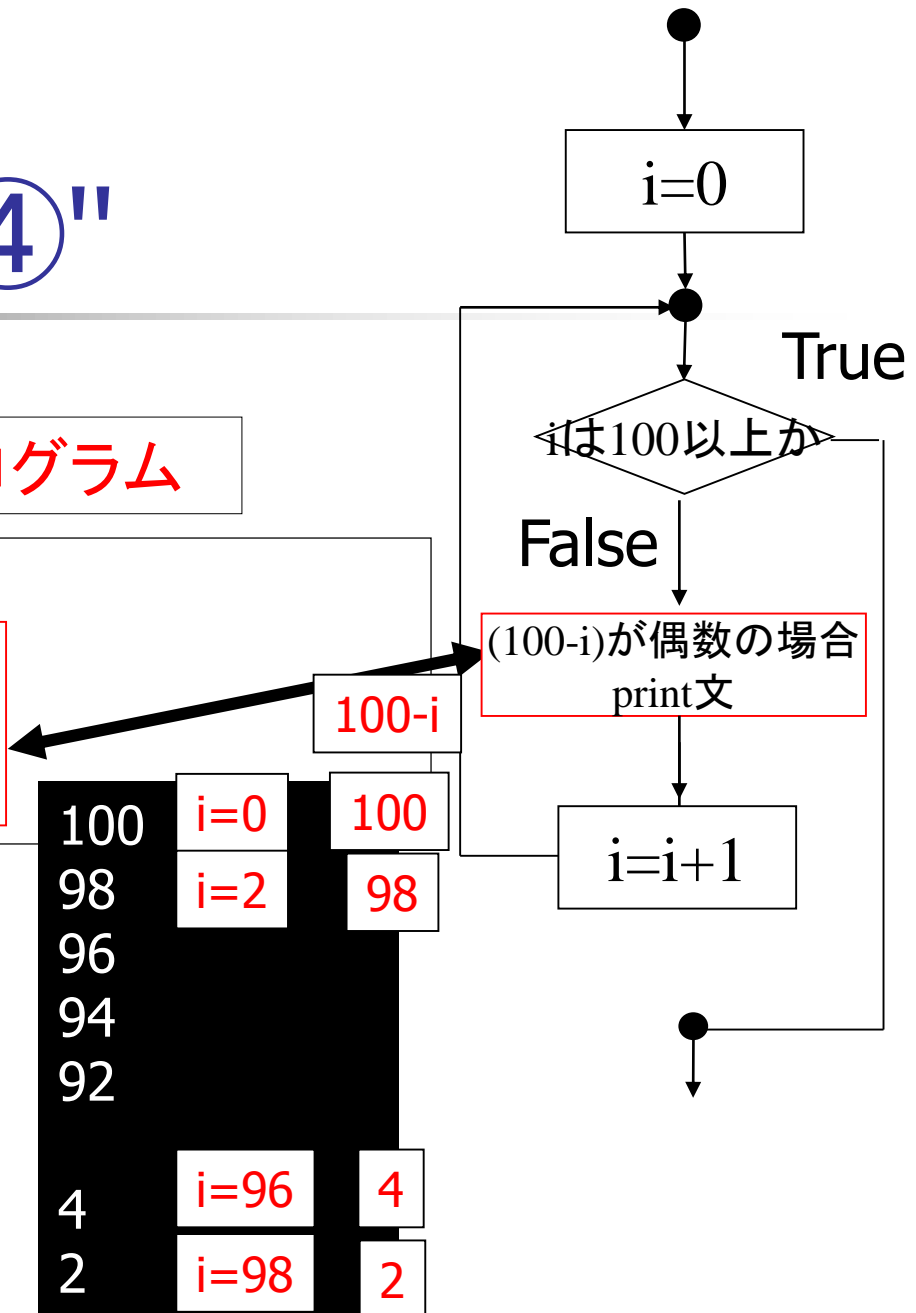
iの2倍の値を出力

2	i=1
4	i=2
6	
8	
10	
12	
98	i=49
100	i=50



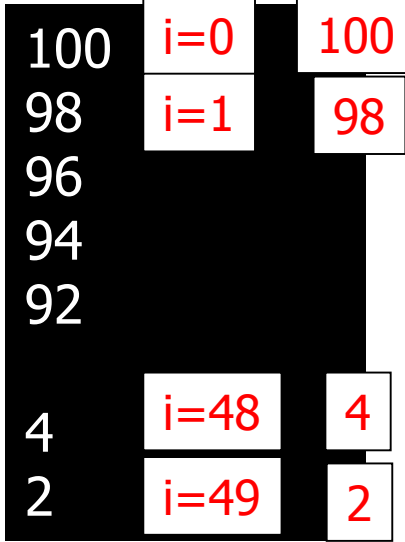
100以下の偶数を表示するプログラム

```
for i in range(100):  
    if (100-i) % 2 == 0:  
        print( 100-i )
```



100以下の偶数を表示するプログラム

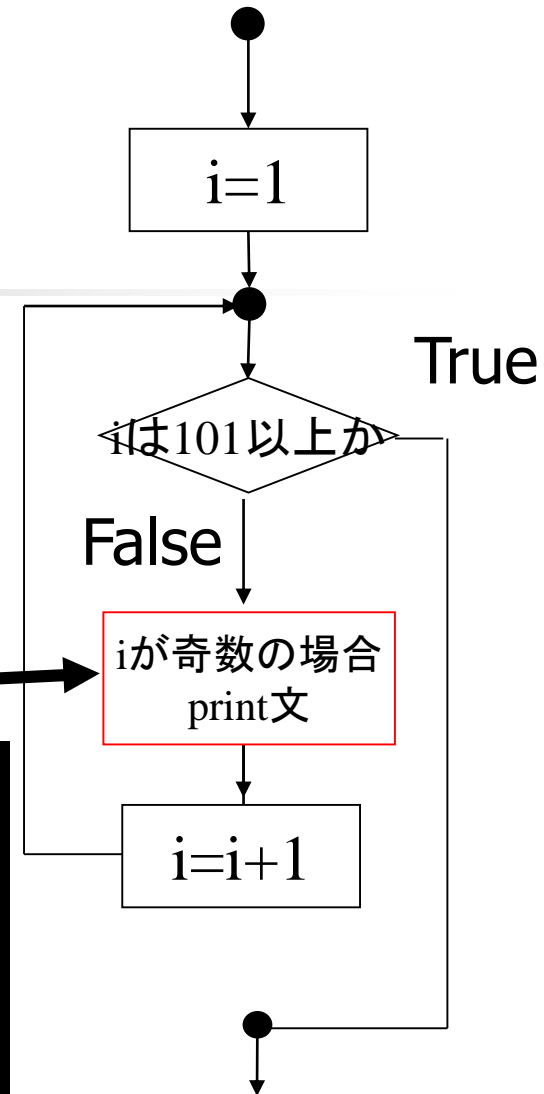
```
for i in range(50):  
    print( (50-i)*2 )
```



繰り返しの例⑤

100以下の奇数を表示するプログラム

```
for i in range(1,101):  
    if i % 2 != 0:  
        print( i )
```



1	i=1
3	i=3
5	
7	
9	
11	
97	i=97
99	i=99

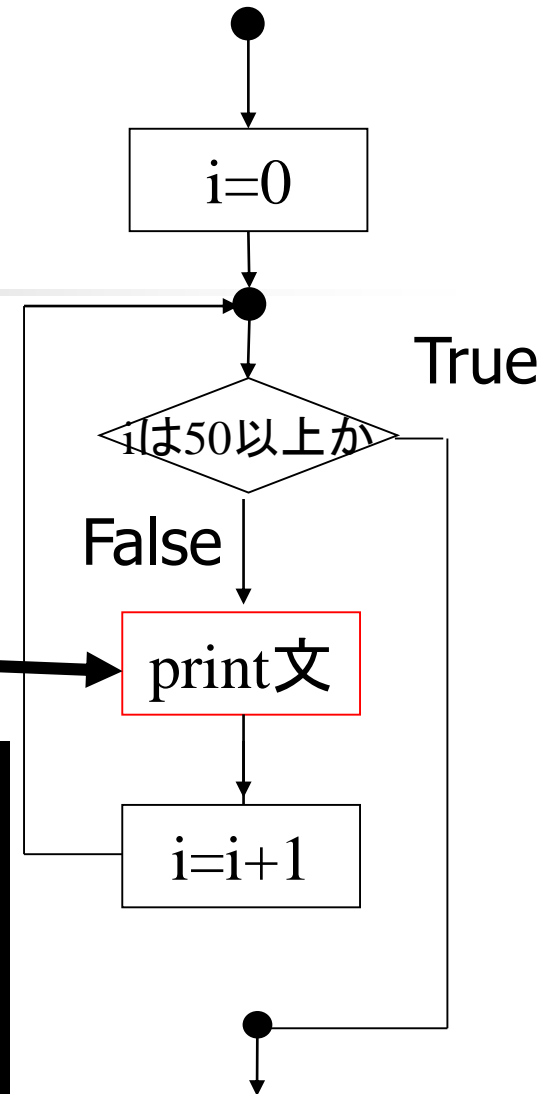
繰り返しの例⑤'

100以下の偶数を表示するプログラム

```
for i in range(50):  
    print( i*2+1 )
```

($i*2+1$)の値を出力

1	i=0
3	i=1
5	
7	
9	
11	
97	i=48
99	i=49

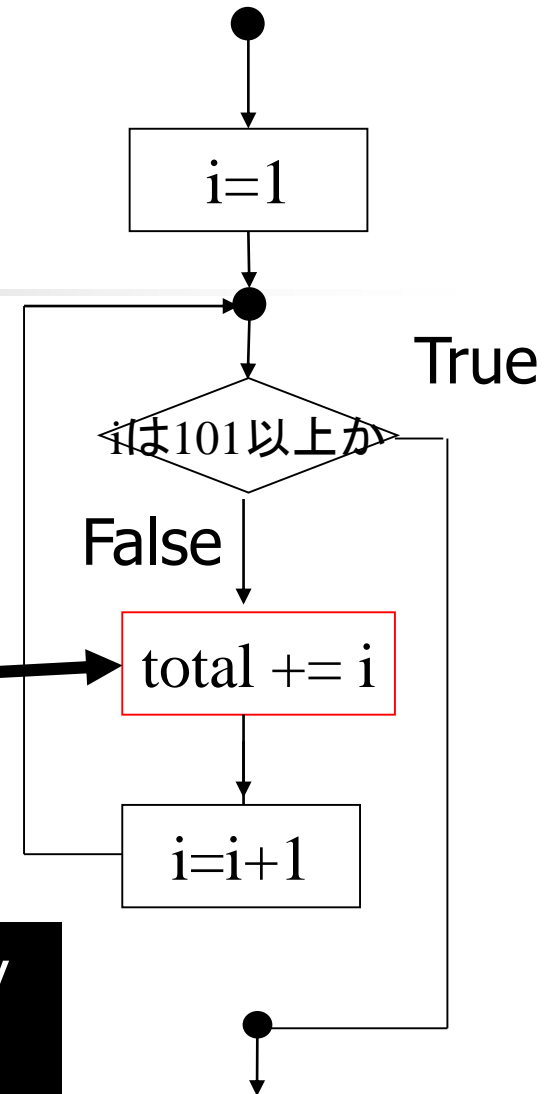


繰り返しの例⑥

100以下の整数の和を求めるプログラム

```
total = 0
for i in range(1,101):
    total += i
print( "合計は" , total )
```

```
> python sample.py
合計は 5050
```

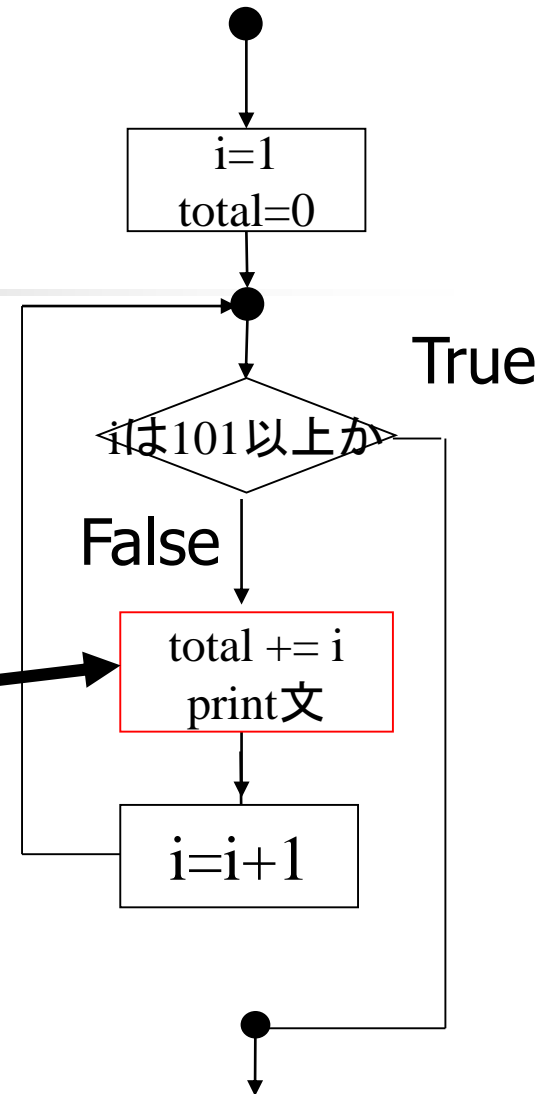


繰り返しの例⑥'

100以下の整数の和を求めるプログラム

```
total = 0
for i in range(1,101):
    total += i
    print( i , total )
print( "合計は" , total )
```

実行結果を見て、動作を理解して下さい



繰り返しの例⑥'

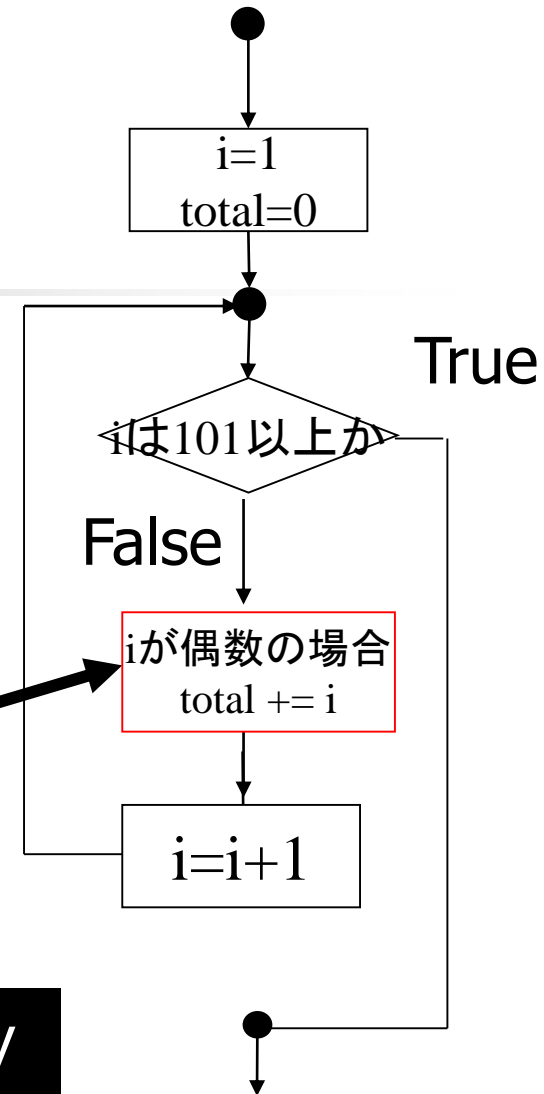
i	total	
1	1	i=1, totalは0 → total +=i により totalは1
2	3	i=2, totalは1 → total +=i により totalは3
3	6	i=3, totalは3 → total +=i により totalは6
4	10	i=4, totalは6 → total +=i により totalは10
5	15	i=5, totalは10 → total +=i により totalは15
6	21	i=6, totalは15 → total +=i により totalは21
7	28	
8	36	
9	45	
10	55	
11	66	
12	78	i=12, totalは66 → total +=i により totalは78

繰り返しの例⑦

100以下の偶数の合計を求めるプログラム

```
total = 0
for i in range( 1,101 ):
    if i % 2 == 0:
        total += i
print( total )
```

```
> python sample.py
2550
```

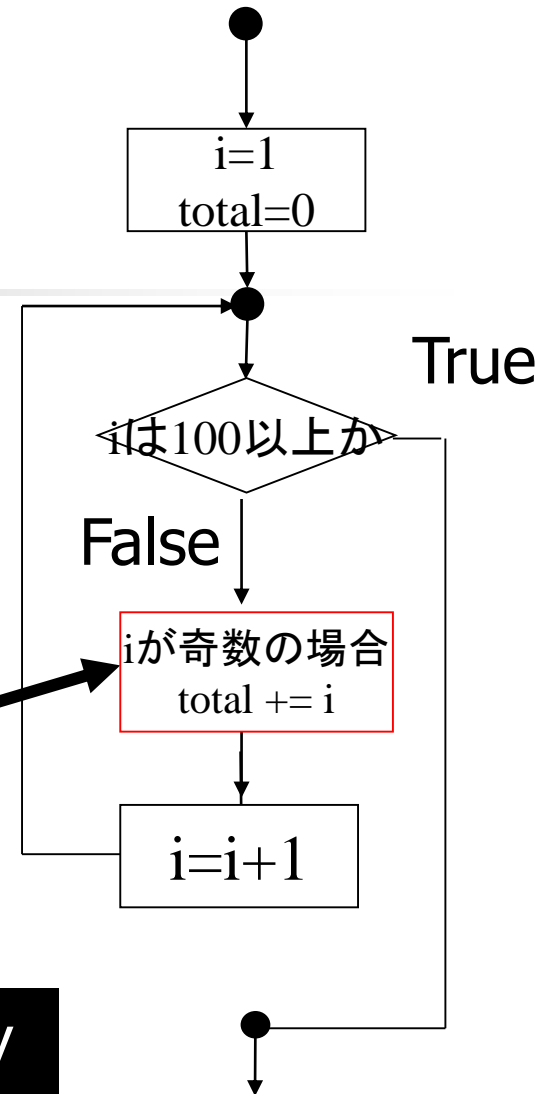


繰り返しの例⑦'

100以下の奇数の合計を求めるプログラム

```
total = 0
for i in range( 1,100 ):
    if i % 2 != 0:
        total += i
print( total )
```

```
> python sample.py
2500
```

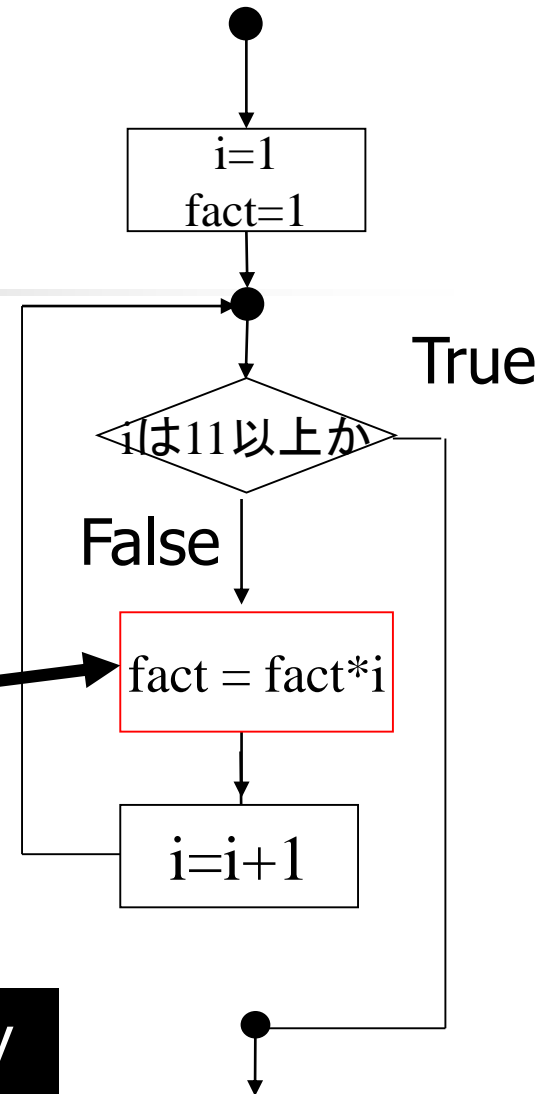


繰り返しの例⑧

10までの階乗を求めるプログラム

```
fact = 1
for i in range( 1,11 ):
    fact = fact * i
print( fact )
```

```
> python sample.py
3628800
```





練習問題

練習①～④(時間があれば⑤も行なって下さい)
問題画面の通りに表示させなくてもよいです

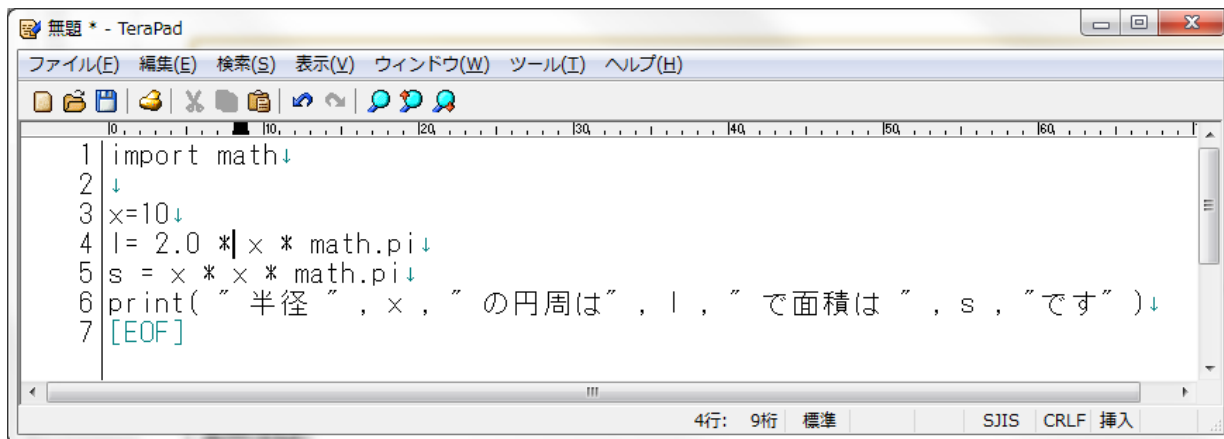


練習問題を提出する上での注意

- プログラムについて
 - エディターをキャプチャーした図ではなく、テキストのままWordに貼り付けて下さい
- 実行結果について
 - 対話型シェルで一文ごと実行するのではなく、pythonコマンドで実行して下さい

練習問題を提出する上での注意

(×)



```
1 import math↓
2 ↓
3 x=10↓
4 l= 2.0 *| x * math.pi↓
5 s = x * x * math.pi↓
6 print( " 半径 " , x , " の円周は" , l , " で面積は " , s , " です" )↓
7 [EOF]
```

The screenshot shows a TeraPad window with a menu bar (File, Edit, Search, View, Window, Tools, Help) and a toolbar. The code is written in a monospaced font. Line numbers 1 through 7 are on the left. Blue arrows point to the end of lines 1, 2, 3, 4, 5, and 6. A vertical line is placed after the first space in line 4. The status bar at the bottom indicates '4行: 9桁 標準' and 'SJIS CRLF 挿入'.

(○)

```
import math
x=10
l= 2.0 * x * math.pi
s = x * x * math.pi
print( " 半径 " , x , " の円周は" , l , " で面積は " , s , " です" )
```


練習問題を提出する上での注意

(×)

```
コマンド プロンプト - python
C:\Users\shino>python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:54:40) [MSC v.1900 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import math
>>> x=10
>>> l= 2.0 * x * math.pi
>>> s = x * x * math.pi
>>> print( " 半径 ", x, " の円周は ", l, " で面積は ", s, " です" )
 半径  10   の円周は 62.83185307179586   で面積は  314.1592653589793   です
>>>
```

(○)

```
C:\Windows\system32\cmd.exe
C:\Users\shino\Desktop>python 4-2.py
 半径  10   の円周は 62.83185307179586   で面積は  314.1592653589793   です
C:\Users\shino\Desktop>
```



練習①

- 成績判定のプログラムを作成します.
- 点数をキーボードから整数値で読み込み, 点数が80点以上はA, 70点以上はB, 60点以上はC, 60点未満はDと印字するプログラムを作成しなさい. ただし, 条件式④(if elif else)を用いて書きなさい
- 29ページのスライドを参考にしなさい.

```
> python 5-1.py  
点数 > 70  
70 点は B です
```



練習②

- クラス分けをします. K組で学籍番号が61809800より小さい, もしくはL組で学籍番号が61812700より小さい場合は, 「教室は703」と印字し, それ以外は「教室は704」と印字するプログラムを書きなさい. クラス名, 学籍番号はキーボードから入力できるようにしなさい.

```
> python 5-2.py  
クラス名? L  
学籍番号? 61801000  
教室は703
```

```
> python 5-2.py  
クラス名? K  
学籍番号? 61813000  
教室は704
```



練習③

- 1から10までの整数について、偶数か奇数かを判定するプログラムを書きなさい
- 「繰り返しの例⑤」を改良すればよい

```
> python 5-3.py  
1 は奇数です  
2 は偶数です  
3 は奇数です  
4 は偶数です  
5 は奇数です  
6 は偶数です  
7 は奇数です  
8 は偶数です  
9 は奇数です  
10 は偶数です
```



練習④

- キーボードから整数 n ($n > 1$)を読み込み, 1から n までの二乗和を求めることができるプログラムを書きなさい
- 「繰り返しの例⑥」を改良すればよい

```
> python 5-4.py  
n? 10  
合計は 385
```



練習⑤

- 下記の式の値を求めるプログラムを書きなさい。計算は浮動小数点演算で行ないなさい

$$x = 4 \left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} \cdots + (-1)^n \frac{1}{2n+1} \right)$$

$$n = 0, 1, 2, \dots, 100$$

```
> python 5-5.py  
3.1514934010709914
```



本日の練習問題

- 練習問題①～④(時間があれば⑤も行なって下さい)を行なって下さい
- プログラム(テキストで貼り付けて下さい)と実行結果をワープロに貼り付けてkeio.jp の教育支援システムから提出して下さい