

Python勉強会@HACHINONE

第2章

Pythonの概要

プログラミング言語の種類

Python勉強会@HACHINOHE

- 低級言語(ハードに近い)、高級言語(人間に近い)
- 汎用言語、特殊用途言語
- インタプリタ(プログラムを直接実行)、コンパイラ(機械語に変換して実行)
- Pythonは、高級汎用インタプリタ言語
 - 静的意味論チェックが甘い、大規模プロジェクト向きでない
 - シンプルで学びやすい、ライブラリが豊富
 - 最新はバージョン3だが、この本はバージョン2

シェル

Python勉強会@HACHINOHE

- シェルの起動: Macの場合
 - アプリケーション→ユーティリティ→ターミナル
 - ターミナルにpythonと入力して[Enter]
 - シェルを抜けるには[Ctrl]+d
- 文字列の表示

```
print 'Yankees rule!'  
print 'But not in Boston!'  
print 'Yankees rule,', 'but not in Boston!'
```



実行すると

```
Yankees rule!  
But not in Boston!  
Yankees rule, but not in Boston!
```

オブジェクトと型

Python勉強会@HACHINOHE

- Pythonでは、データはオブジェクト
- オブジェクトの型の種類
 - スカラ
 - int: 整数
 - float: 浮動小数点数
 - bool: 真偽値 True / False、George Booleの名前より
 - None
 - 非スカラ
 - 文字列など
- 型はtype(オブジェクト)で調べられる

演算子

Python勉強会@HACHINOHE

- intやfloatに対する演算子
 - 結果がintやfloat
 - 和+, 差-, 積*, 商//, 余り%, 除算/, べき乗**
 - 結果がbool
 - 等しい==, 等しくない!=, 大きい>, 以上>=, 小さい<, 以下<=
- boolに対する演算子
 - 結果がbool
 - 論理積and, 論理和or, 否定not

変数、コメント

Python勉強会@HACHINOHE

- 変数は、オブジェクトに紐付けられた名前
 - 名札のようなもの
 - オブジェクトを参照している
- 変数にオブジェクトを紐付けるには

変数 = オブジェクト
- 変数の名前は適切につけるべき
- #以降がコメントになる

分岐プログラム

Python勉強会@HACHINOHE

- if～else

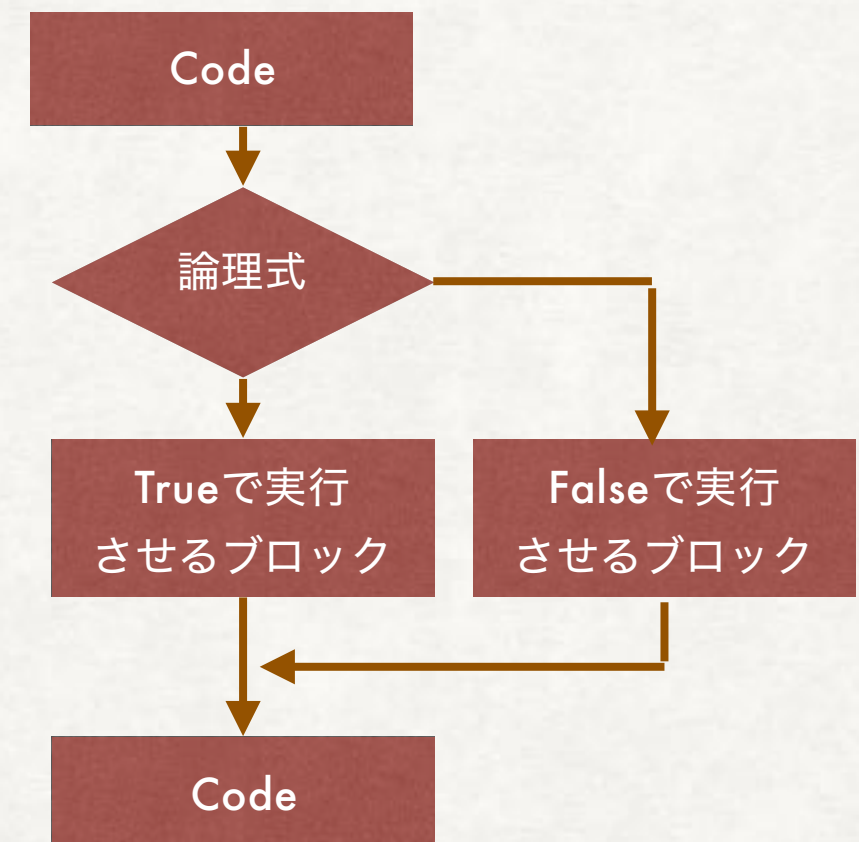
```
if 論理式:  
    論理式の値がTrueのとき実行されるブロック  
else:  
    論理式の値がFalseのとき実行されるブロック
```

- if～elif～else

```
if 論理式:  
    コードブロック  
elif 論理式:  
    コードブロック  
else:  
    コードブロック
```

- 実行時間

- 分岐プログラムは、分岐しない直線的プログラムよりも実行時間は短い
- 実行時間が入力データの量によらない場合、定数時間で実行されるという



指練習: 2.2節

Python勉強会@HACHINOHE

```
# -*- coding: utf-8 -*-
# 3つの数
x = 10
y = 13
z = 9

# 最大の奇数(0の場合は奇数がなかった)
max_odd = 0

if x % 2 == 1:
    max_odd = x
if y % 2 == 1 and y > max_odd:
    max_odd = y
if z % 2 == 1 and z > max_odd:
    max_odd = z

if max_odd != 0:
    print '最大の奇数は' + str(max_odd) + 'です'
else:
    print '奇数はありません'
```


文字列

Python勉強会@HACHINOHE

- 文字列のリテラルは、"か"でくる
- オーバーロード: 多重定義
 - 「+」は数値同士なら足し算、文字列同士なら連結
- 文字列はシーケンス型的一种
- シーケンス型
 - len(オブジェクト)で、長さ
 - オブジェクト[添字]で、添字番目の要素にアクセス
 - オブジェクト[start:end]で、start番目からend-1番目の要素にアクセス

入力

Python勉強会@HACHINOHE

- `raw_input(プロンプト)`で入力を文字列として取得できる

```
>>> name = raw_input('Enter your name: ')
Enter your name: George Washington
>>> print 'Are you really', name, '?'
Are you really George Washington ?
>>> print 'Are you really ' + name + '?'
Are you really George Washington?
```

- 文字列などを整数に変換するには型変換を行う

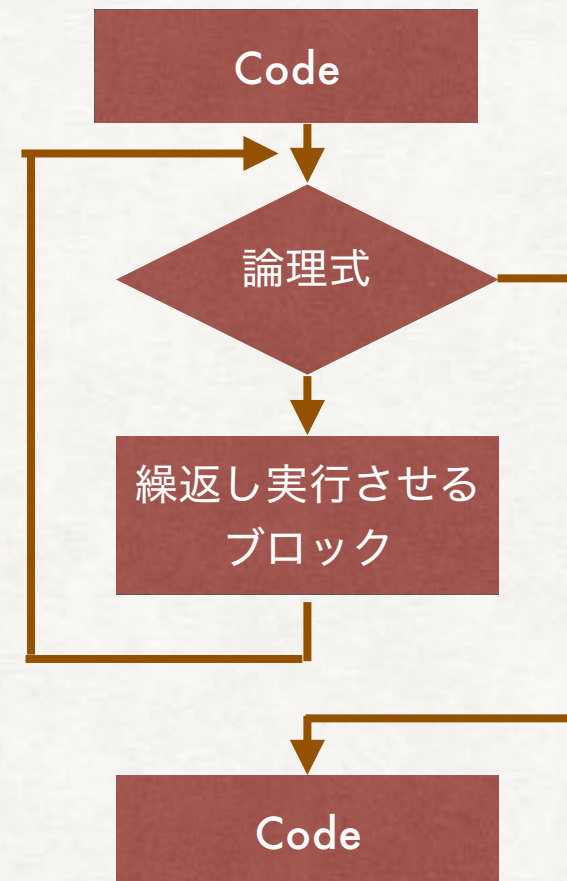
```
int(文字列など)
```

繰返し

Python勉強会@HACHINOHE

- while

```
while (論理式):  
    繰返し実行させるブロック
```



整数の自乗を難しい方法で求める例

donec quis nunc

- xの2乗を計算するプログラムの例
 - 掛け算を使わなくても、xの2乗は、xをx回足せば計算できる

```
# -*- coding: utf-8 -*-  
# 整数の自乗を難しい方法で求める  
x = 3 # 自乗する数  
ans = 0 # 結果  
itersLeft = x # 残りの繰返し回数  
while (itersLeft != 0): # itersLeftが0になるまで繰返す  
    ans = ans + x # ansをx増やす  
    itersLeft = itersLeft - 1 # itersLeftを1減らす  
printf str(x) + '*' + str(x) + ' = ' + str(ans)
```


指練習: 2.4節

Python勉強会@HACHINOHE

```
# -*- coding: utf-8 -*-
itersLeft = 10 # 繰り返す回数
max_odd = 0

while (itersLeft > 0):
    # 値を読み込んで整数に変換してtmpに代入
    tmp = int(raw_input('正の整数を入力してください:'))
    if tmp % 2 == 1 and max_odd < tmp:
        max_odd = tmp
    itersLeft = itersLeft - 1

if max_odd != 0:
    print '最大の奇数は' + str(max_odd) + 'でした'
else:
    print '奇数は入力されませんでした'
```