プログラミングII

黒瀬　浩

kurose@neptune.kanazawa-it.ac.jp

OH: 講義の前後，Eメール問合せ，月3限21-405

居室　67・121

第1回復習　辞書，辞書のメソッド

```
d = dict(a=1, b=2, c=3, d=4)        # 変数dとキーdは別物
```
(d = {"a":1, "b":2, "c":3, "d":4} と書いても上と同じ)

キー一覧	d.keys() ⇨*1	['a', 'b', 'c', 'd']
値一覧	d.values() ⇨*1	[1, 2, 3, 4]
ペア一覧	d.items() ⇨*1	[('a',1), ('b',2), ('c',3), ('d',4)]
要素数	len(d)	

1つづつ処理する書き方

```
for k,v in d.items():
    print("key=", k, "val=", v)
# items()メソッドで得られたタプルの1個目が変数kに，2個目が変数vに入る
```

並び替え(1番上の変数dを定義して以下を確認せよ)

キー昇順	sorted(d.items())
キー降順	sorted(d.items(), reverse=True)
値昇順	sorted(d.items(), key=lambda x:x[1])
値降順*2	sorted(d.items(), key=lambda x:x[1], reverse=True)

*1: 正確には，list(d.keys()) のようにリスト化が必要

*2: lambdaはその場のみで有効な関数を定義する(後述)

第2回復習 map, reduce, filter　　無名関数lambda

map　全ての要素に同じ演算を行う
reduce 要素を順に演算し集約する(functoolsパッケージ)
filter　条件に合うものを残す

上記は，第1引数に演算の無名関数を，第2引数に対象を指定する
いずれも，for文やlist()などを使わないと値が解らない(遅延評価)

```
map( lambda x: x*x, range(1,11) )    # 数列をそれぞれ2乗
import functools as ft # recude はimportが必要
ft.reduce( lambda x,y: x+y, range(1,11) ) # 数列の総和
filter( lambda x: x%2==0, range(1,11) ) # 条件による抽出
```

map,reduce,filterはリスト内包や関数でも実装可能

```
[x*2 for x in range(11)]                    # 2倍の数列
sum( range(11) )                            # 総和
[x for x in range(11) if x%2==0]            # 偶数のみ残す
```

math.sin()は，引数は数値だがnumpy.sin()はリストを渡せる map機能あり

第3回復習

イテレータ(iterator)

- iter()関数　で1つづつ値を返すオブジェクトを作れる
- next(イテレータオブジェクト)　で1つづつ値を取り出せる
- 終わりまでいくとStopIterationエラーが返る
- forはイテレータを取り出す動作を行う

ジェネレータ(generator)

- 関数で定義するが，returnでなくyieldで値を返す
- 関数は終了しないので関数内の変数は保持される
- 関数でやることがなくなる（またはreturn)でジェネレータは消滅

range()関数　整数列生成

- 引数1個　0から終了の1つ前まで
- 引数2個　開始から終了の1つ前まで
- 引数3個　開始から終了の1つ前まで指定間隔ごと

可変長引数

- 引数の数が異なっても受けられる
- 受け取る変数に*をつける（*の付いていないものより後に指定)
- タプルで渡される

例外(exception)

プログラムはそれ以上動作できないエラーが出ると異常終了する

0除算	数学的に未定義
ファイル入出力	書けない，読めない，存在しないなど
文法エラー	解釈できない記法

OS(オペレーティングシステム)で例外が出ると強制終了

処理を続けたい場合がある

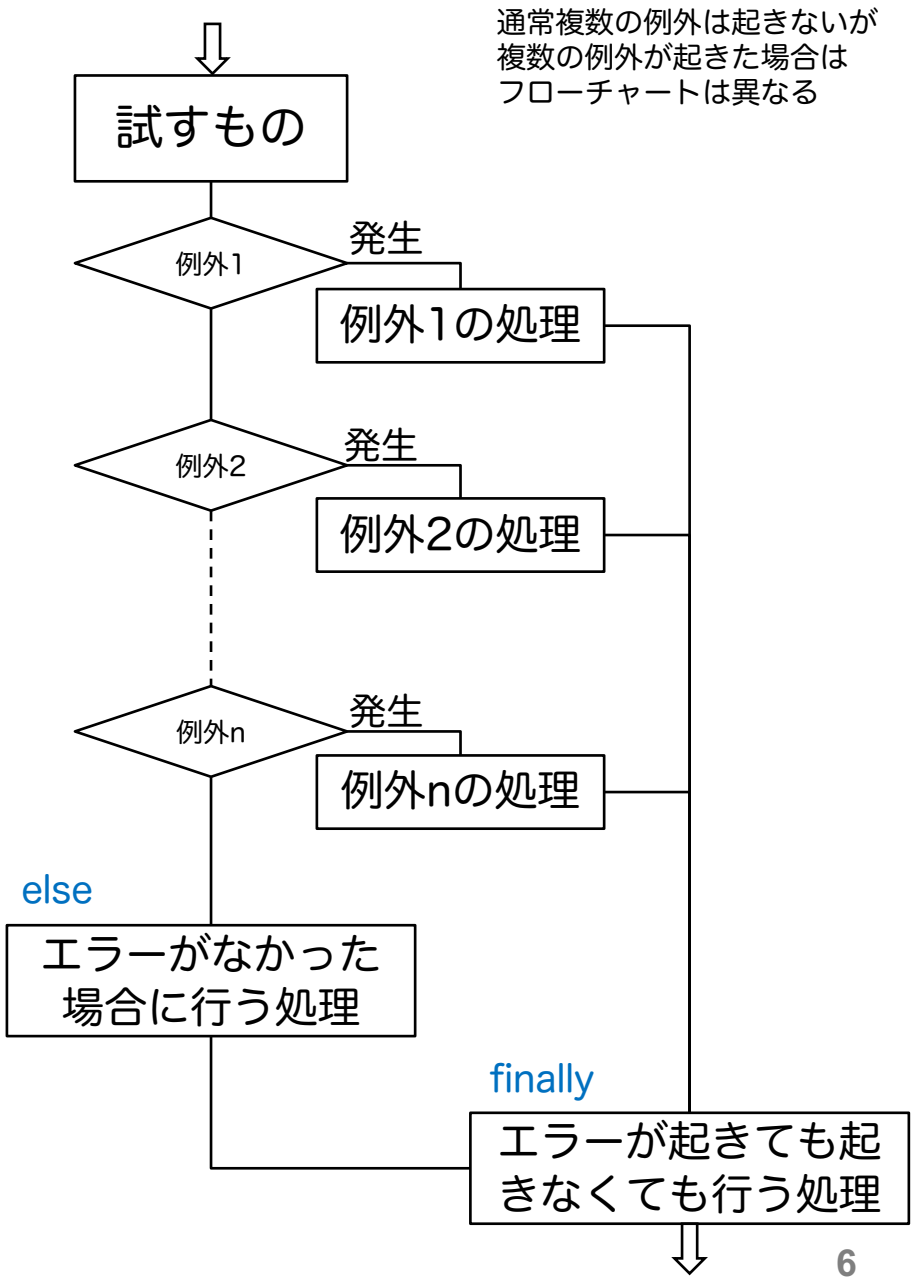
自分で例外を定義したい場合がある

```
try:
    試すもの
except:
    エラーが発生した場合の処理
```

try…exceptの構文

```
try:
        試すものを書く
except 例外1:
        例外1が発生した場合の処理
except 例外2:
        例外2が発生した場合の処理
        (中略)
except 例外n:
        例外nが発生した場合の処理
else:
        エラーがなかった場合実行する
finally:
        エラーが起きても起きなくても行う
```

青字はオプション



pythonの例外　詳細はdocs.python.jpの組み込み例外を参照

AssertionError	assert文失敗
AttributeError	属性の参照や代入失敗
TypeError	データ型に対応していない
EOFError	end-of-file (EOF) に達した
FloatingPointErro	浮動小数点演算失敗
GeneratorExit	ジェネレータ や コルーチン が閉じられた
ImportError	importできなかった
ModuleNotFoundError	import 文でモジュールが見つからない
IndexError	シーケンスの添字が範囲外
KeyError	マッピング (辞書) のキーが見つからなかった
KeyboardInterrup	ユーザが割り込みキー (Control-C) を押した
MemoryError	操作中にメモリが不足した
NameError	ローカルまたはグローバルの名前が見つからなかった
NotImplementedError	抽象メソッドが派生クラスでオーバライドされる
OSError	システム関数がシステム関連のエラーを返した
OverflowError	算術演算の結果が表現できない大きな値になった
RecursionError	最大再帰深度の超過を検出
ReferenceError	回収された後の参照対象オブジェクトの属性にアクセスした
RuntimeError	他のカテゴリに分類できないエラーが検出された
StopIteration	イテレータが生成するアイテムがこれ以上ない
以下略	

エラーの確認と例外を受ける処理

以下対話型で確認

```
3/0
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: division by zero
```

tryしないとエラーで終了

```
try:
    a=3/0
except ZeroDivisionError:
    print("ゼロで割った")
```

ゼロで割った

例外を指定すると発生した例外により動作する
大文字で始まる変数を推奨しない理由の１つは左

```
try:
    a=3/0
except:
    print("ゼロで割った")
```

ゼロで割った

例外を指定しないと何か例外が起きたら動作する
0除算でなくても例外処理が動く

例外を発生させる raise

例外処理の試験をする場合
python構文としては正しいが例外を追加したい場合

以下を実行し，aを負値にして再実行せよ

```
a = 10
try:
    if a<0:
        raise ValueError("negative value!")
except ValueError as e:
    print(e)
else:
    print("エラーは起きなかった")
```

最初はaが正なので例外が起きなず，elseブロックが実行される
aを負にすると例外が起き，as eのeにメッセージが渡される
ほかに，デバッグ情報を得るtraceback()などがある

range()互換のrange0()とforと同様な処理をwhileでかけ

range0（当然内部でrange()は使わない)

可変長引数で受ける．引数の数はlen()で得られる.

yieldで値を返す

開始の既定値(デフォルト)は0, 間隔の規定値は1

forと同様な処理を行う関数for0() (当然内部でfor文は使わない)

while Trueで無限ループする

range0()をジェネレータにして，next()で値を取り出す

try～exceptを使い，StopIteration例外が出たらbreakで抜ける

値をリストに貯めておき，printする

試験するrange0()の引数は4種類を試すこと

```
for0( range0(5) )          期待する出力  [0, 1, 2, 3, 4]
for0( range0(2,7) )              期待する出力  [2, 3, 4, 5, 6]
for0( range0(2,7,2) )            期待する出力  [2, 4, 6]
for0( range0(10,2,-2) )          期待する出力  [10, 8, 6, 4]
```

4つ目はチャレンジ (なるべく動かす努力をする)

レポート2

タイトル　　プログラミングII　レポート2 range,forの実装
タイトルを上部中央
クラス名列，氏名を上部右
前ページを実現するソースコードと実行結果のハードコピー（スクリーンショット)を貼る
A4 1枚に収める(両面可，横長配置可)

画像のフォントサイズの注意
レポート本文のフォント10〜12ポイントと同じ程度の字の大きさ
(大きすぎないこと)
画像の縦横比を変えない
見難い配色は避ける

期限：　第6回開始時
注意：　不明点は提出前に確認すること