演習や課題についての補足

- 設問文の意図がわからない場合
  - どんどん聞いてください。(文章自体が不適切なケース(出題側の問題)があります)
- 出力結果を読もう
  - 演習レポートに「エラーが出たまま次に進んでいる」ケースがたまに見つかります。
- 報告書
  - 学籍番号、氏名を書こう。
- 変数名
  - 命名規則を参考に、適切な変数名を付けよう。
- 個別コメント
  - Google共有フォルダ「exercise-check-0506.pdf」参照。

プログラミング1

(第4回) 関数の利用2、ループ処理（while文）

1. 振り返り（ミニクイズ）
2. Chapter 4.1.1の補足2
    1. 関数とローカル変数
3. Chapter 3.1の補足
    1. Iteration, looping (反復処理)
    2. ループ処理の例、実行例
    3. 3種類の処理流れ制御
4. 演習・課題への取り組み方、レポートの書き方のデモ
5. 演習
    1. 演習1～4: 初めてのレポート
    2. 演習5: if文, 関数の利用
    3. 演習6: while文
6. 宿題

講義ページ: http://ie.u-ryukyu.ac.jp/~tnal/2016/prog1/

ミニクイズの振り返り

- [教科書] 2.1.2 変数と代入
- 変数名・ファイル名の命名規則
- 条件分岐

Chapter 4.1.1, 3.1の補足

4.1.1 Function Definitions

3.1

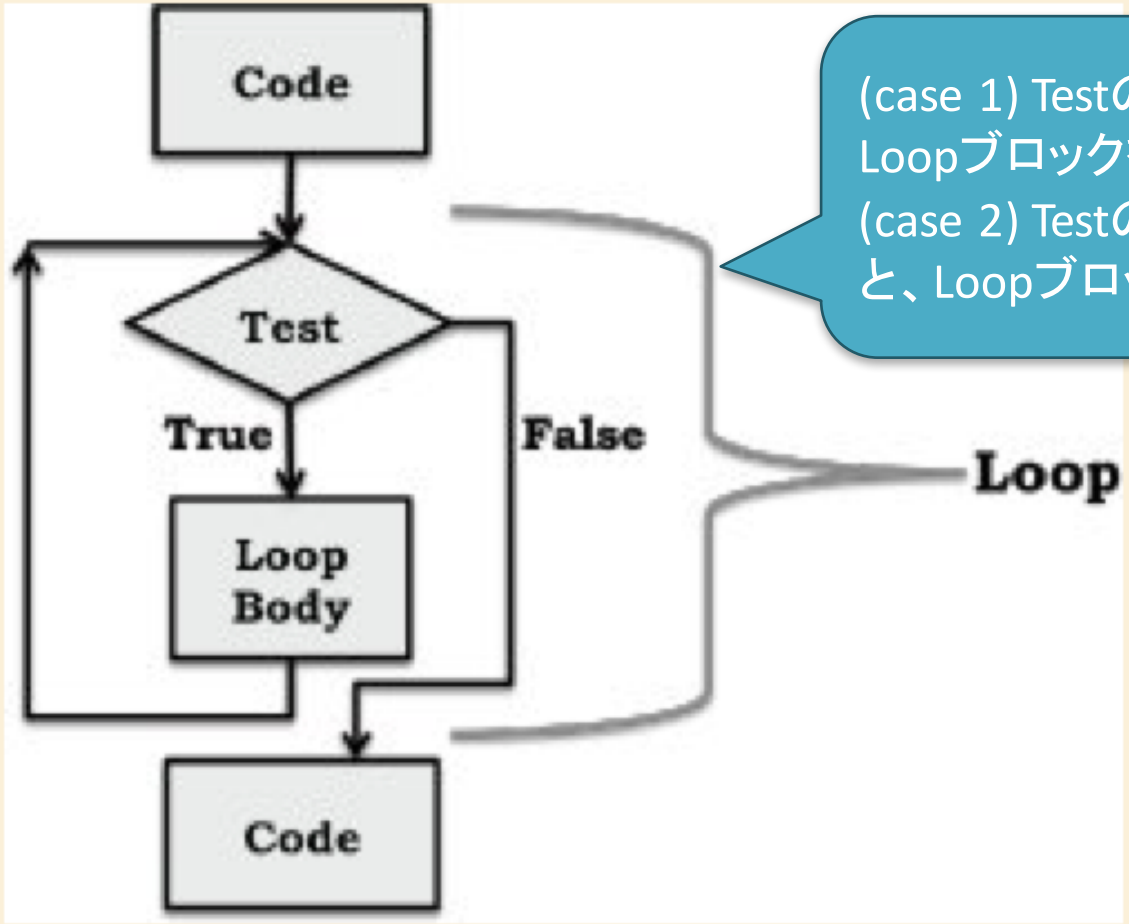
4.1.1 Function Definitions (関数定義)

・一連の手続きにaddという名前を付けた。
・addという関数は2つの引数を必要とする。
・x, y, a, b は関数内での呼び名（ローカル変数）。引数に名前をつけただけ。

```
def add(x, y):
    answer = x + y
    return answer
```

```
def add(a, b):
    answer = a + b
    return answer
```

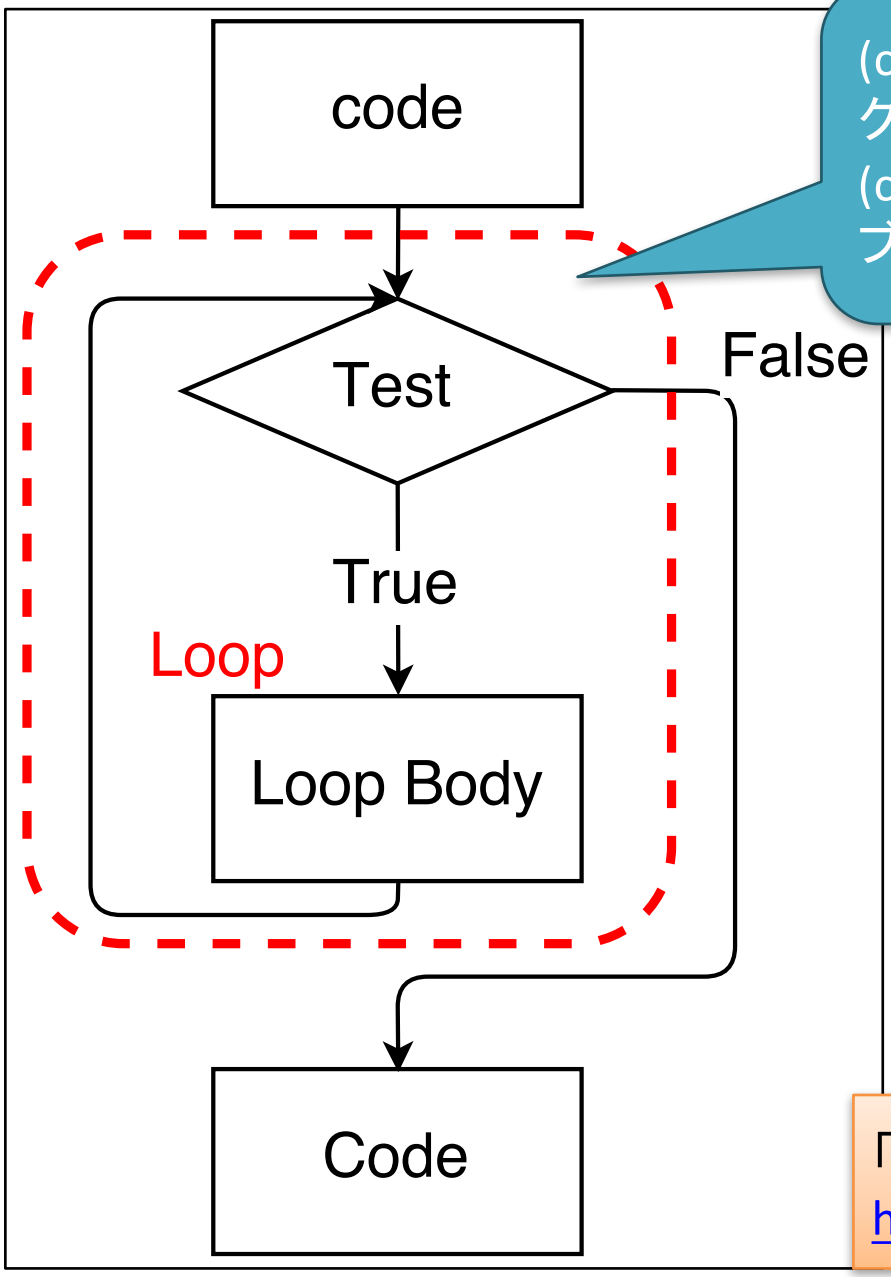
2.4 Iteration, looping (反復処理, 繰り返し処理)



(case 1) Testの結果がTrueの間、Loopブロックを繰り返す。
(case 2) Testの結果がFalseになると、Loopブロックを抜ける。

Figure 2.4 Flow chart for iteration

2.4 Iteration, looping (反復処理, 繰り返し処理)



(case 1) Testの結果がTrueの間、Loopブロックを繰り返す。
(case 2) Testの結果がFalseになると、Loopブロックを抜ける。

「draw.io」図描画webサービス
https://www.draw.io/

ループ処理の例 (2.4節の改良版)

http://ie.u-ryukyu.ac.jp/~tnal/2016/prog1/loop.py

```
# スライムのHPが0より大きい間タコ殴りにするゲーム

import random

def encount_enemy():
    hitpoint = random.randint(3,7)
    return hitpoint

enemy_hitpoint = encount_enemy()
print('スライムに遭遇した。（敵HP=', enemy_hitpoint, '）')

while (enemy_hitpoint > 0):
    attack = random.randint(2,4)
    enemy_hitpoint = enemy_hitpoint – attack
    print('あなたの攻撃！スライムに', attack, 'のダメージ！（敵
HP=', enemy_hitpoint, '）')

print('スライムを倒した！')
```

HP3～7を取る敵を用意する関数を定義。この時点では、まだ実行されない点に注意。

定義した関数を使って、敵を用意。

「条件がTrueの間」＝「スライムのHPが0より大きい間」、下記ブロックを繰り返す。

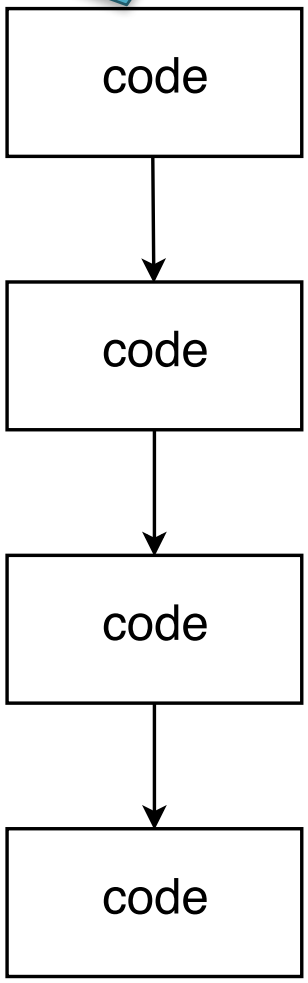
実行例 (ファイルのダウンロード+実行)

Usage: curl URL –o filename

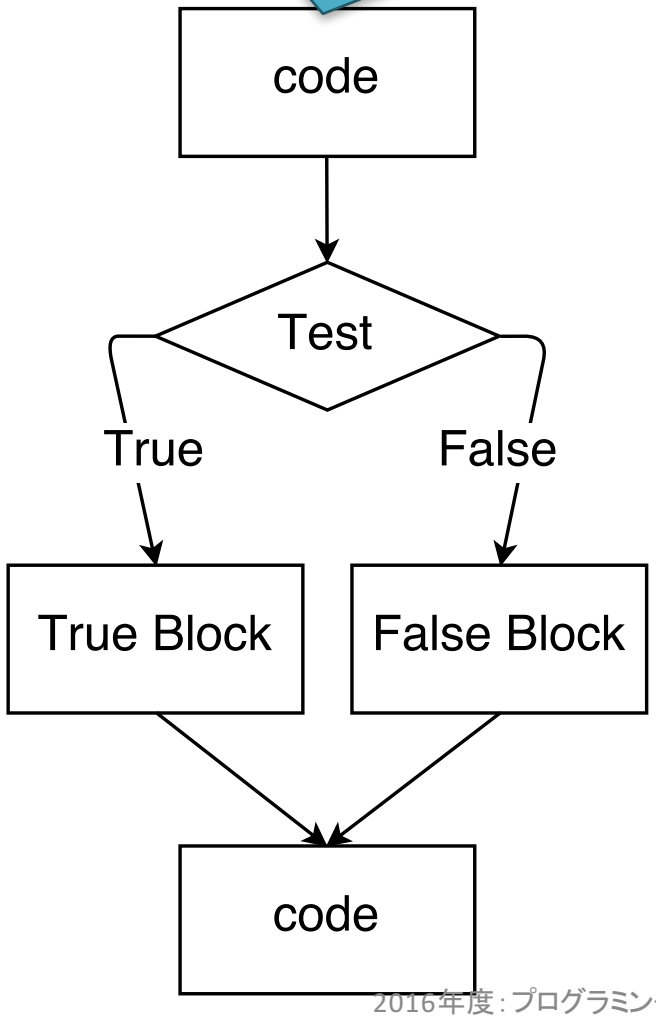
```
oct:tnal% curl http://ie.u-ryukyu.ac.jp/~tnal/2016/prog1/loop.py -o loop.py
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   473  100   473    0     0  37168      0 --:--:-- --:--:-- --:--:-- 39416

oct:tnal% python3 loop.py
スライムに遭遇した。（敵HP=5）
あなたの攻撃！スライムに2のダメージ！（敵HP=3）
あなたの攻撃！スライムに2のダメージ！（敵HP=1）
あなたの攻撃！スライムに4のダメージ！（敵HP=-3）
スライムを倒した！
```

3種類の流れ制御 (control flow)

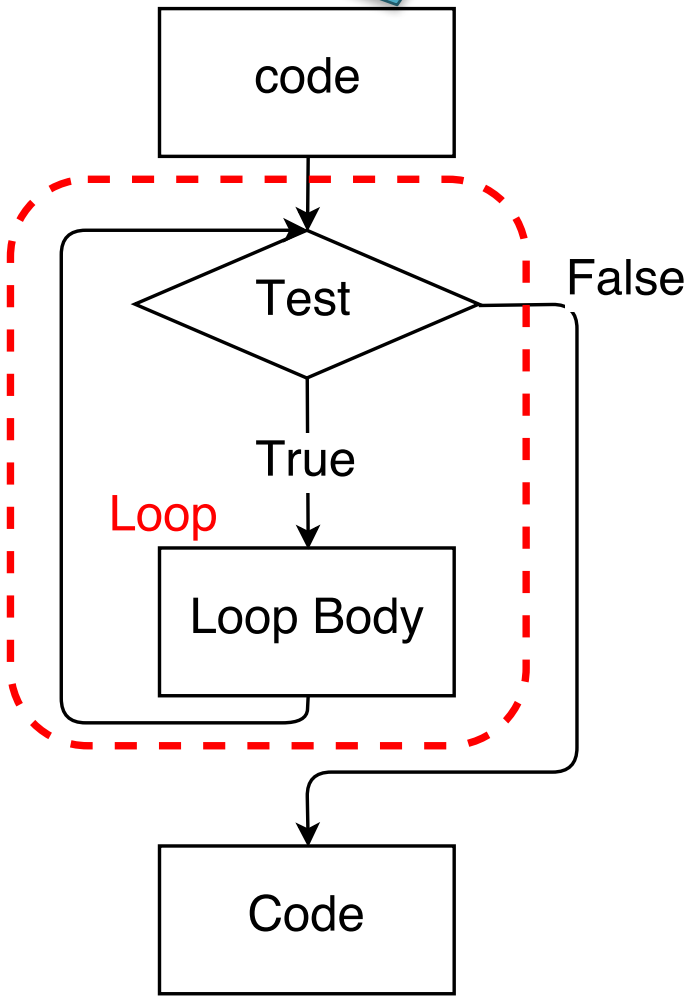
Sequence,
ordered statement
(逐次処理)



Selection,
conditional statement
(条件分岐)



Iteration, looping
(反復処理)



Reserved words, 予約語

https://goo.gl/rEzdAN

- 一覧（赤丸は今回出てきた予約語）

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

演習・課題への取り組み方、レポートの書き方のデモ

取り組み方の例

- 問題を分割する。
  - 分割して分かるところから手を付ける（土台を作る）
  - 分からないところは、更に分割できないか考える
    - それでも分からないなら、分割の仕方や、分割した問題の解き方を訪ねてみよう
    - 「分割の仕方」＝問題解決手段の一つ
- 個々のサブ問題を個別に解く。
  - これ以上分割できない≒最小の部品なら教科書・授業で習ってるはず
    - ->該当部分を復習
      - 該当部分が分からないなら、該当部分の探し方を訪ねてみよう。
      - 教科書・授業の復習が足りてないかも。
- それらの組み合わせ方を考える。

演習

演習1〜4: 初めてのレポート

演習5: if文, 関数の利用

演習6: while文の利用

補足1

- ペアプログラミングを始める前に
  - 記入漏れ
    - 「実施日」と「報告者」
  - 前回の復習確認
    - 「何をやったっけ？」
    - 「これはこうやれば良いんだっけ？」

補足2

- ペアプログラミングのやり方
  - 7ステップ
    - 作業を決める
    - 最初の目標を決める
    - パートナーを頼りにし、支えてやる
      - driver: 仕事を終わらせることに専念
      - observer: 横から観察し、疑問・改善・簡潔化など大局的な問題について考える
    - 喋る
      - 「一人で悩む」のは十秒程度に留める
      - 一緒に相談しながら考える練習
    - お互い何をやっているか把握する
      - 頻繁に同期をとる
    - 喜ぶ
    - 交代する

分業ではない

二人で2,3分考えても分からない場合には、手を上げて質問しよう

演習

- 演習1～演習4: 初めてのレポート
  - https://ie.u-ryukyu.ac.jp/~tnal/2016/prog1/week2-ex.html
  - レポートの作成手順は授業ページを参照
- 演習5: if文と関数
  - https://ie.u-ryukyu.ac.jp/~tnal/2016/prog1/week3-ex.html
- 演習6: while文
  - https://ie.u-ryukyu.ac.jp/~tnal/2016/prog1/week4-ex.html
- ペア・プログラミング
  - https://ie.u-ryukyu.ac.jp/~tnal/2016/prog1/week2-pair-programming.html
  - driver, observer (navigator)

宿題

- 復習: 適宜(これまでの内容)
- 予習: 教科書読み
  - 3章
    - 3.2 For Loops
    - (スキップ) 3.3 Approximate Solutions and Bisection Search
    - 3.4 A Few Words About Using Floats
- 復習・予習(オススメ): paiza
  - Python入門編1:プログラミングを学ぶ(全9回)
    - https://paiza.jp/works/python3/primer
  - プログラミングスキルチェック　＊レベル設定のある課題集
    - https://paiza.jp/challenges/info

参考文献

- 教科書: Introduction to Computation and Programming Using Python, Revised And Expanded Edition
- Python 3.5.1 documentation, https://docs.python.org/3.5/index.html
- Google Python Style Guide, https://google.github.io/styleguide/pyguide.html
- ペアプログラミングのやりかた, http://goo.gl/ZWtlW
- (paiza) Python入門編1:プログラミングを学ぶ (全9回), https://paiza.jp/works/python3/primer