復習

プログラミング1

(第2回) Pythonインタプリタとスクリプトの体験1，ペア・プログラミングの導入

1. Chapter 1 の補足1
    1. Calculations and Remembers
    2. Computational thinking
2. Chapter 2 -- 2.1.2までの補足
    1. Glossaries, 用語集1, 2, 3
    2. Reserved words, 予約語
3. 文字列結合の例
4. スクリプトの利用
    1. スクリプトとは？
    2. スクリプトを書いて動かしてみよう
    3. スクリプト vs.インタプリタ
5. 変数名・ファイル名の命名規則
6. マニュアルの参照
7. 演習
8. 宿題

欲しい出力を得るためのレシピを考える必要がある。レシピ≒アルゴリズム。

レシピを記述するための道具（基本的な型・算術演算子・比較演算子・論理演算子）を使えるようになろう。

基本演算とstr.format書式を読めるようになろう。

インタプリタ実行とファイル実行を使い分けよう。

慣習を守ることで「他人が読みやすいコード(readable code)」になる。

help()やオンラインマニュアルを活用しよう。

講義ページ: http://ie.u-ryukyu.ac.jp/~tnal/2018/prog1/

プログラミング1

(第3回) インタプリタとスクリプトの体験2: 文字列とif文, 関数の利用

1. Chapter 2.2, 2.3, 4.1.1の補足
    - 2.2 Branching Programs (条件分岐)
    - 2.3 Strings and Input (文字列と入力)
    - 4.1.1 Function Definitions (関数定義)
    - Reserved words, 予約語
2. ペアプロ演習
3. 宿題

講義ページ: http://ie.u-ryukyu.ac.jp/~tnal/2018/prog1/

Chapter 2.2, 2.3, 4.1.1の補足

2.2 Branching Programs (条件分岐)
2.3 Strings and Input (文字列と入力)
4.1.1 Function Definitions (関数定義)
Reserved words, 予約語

Branching Programs (条件分岐)とは？

- これまでのプログラム
  - インタプリタやファイルに書かれた命令文を、上から下に向かって順序よく実行する。
  - これだけだと、以下のような処理を書けない。
    - ゲームアプリで、レビュー書いたユーザに対して特別報酬を与えたい。
    - Webサービスで、アカウントを作成してログインしたユーザ向けに専用ページを表示したい。

- 条件分岐の考え方
  1. ある条件を満足しているか否かを確認する。(条件判定し、True/Falseいずれなのかを確認する。
  2. 判定結果に基づき、True時の処理(True block)、False時の処理(False block)を分けて記載する。True blockだけでもok。

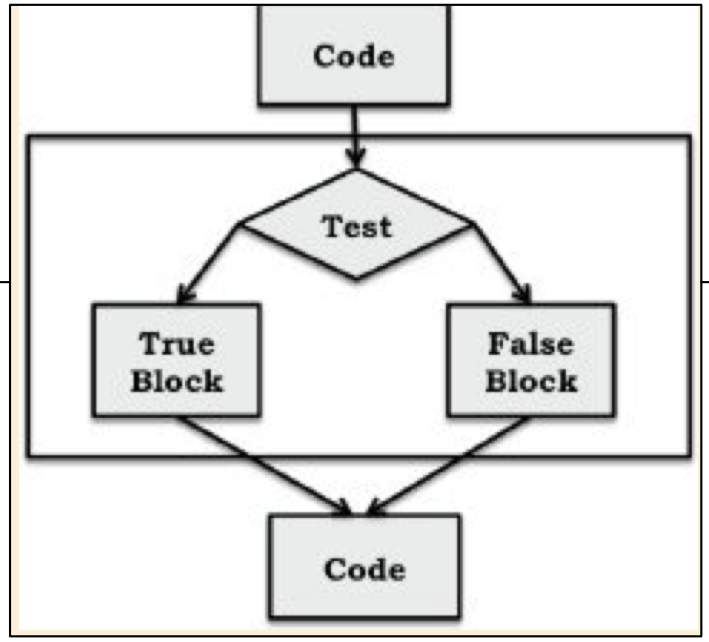

Fig. 2.3 Flow chart for conditional statement

条件分岐（if文, if-statement）の例

＊数字を入力したつもりでもstr型として受け取る。
＊int型として処理したいならint()関数を使って
cast(キャスト, 型変換)しよう。

input()
- 標準入力（≒キーボード）からの入力読み込み、str型を返す。
- 入力を促す文を引数で指定。

```
user_input = input("レポート1の出来は100点満点でどのぐらいですか？")
score = int(user_input)

if score >= 60:
        print('{0}点とはなかなか高評価だね！'.format(score))
        print('その調子で頑張ろう！')
else:
        print('分からないところは自分から相談しよう！')
```

condition (条件)
ブーリアン型となる判定。
If文の場合「if 条件:」と書く。

block(ブロック)
コードのまとまり。
True block

indent (インデント)
ブロックの範囲を示すため、半角スペース4つ、もしくはタブ1つで左端を揃えて列挙する。
CotEditorの場合はタブが1つ自動で挿入される。

2.3 Strings（文字列の補足）

多分時間ないので次回やります！

プログラミング1

(第3回) インタプリタとスクリプトの体験2: 文字列とif文, 関数の利用

if文を使い、条件分岐できるようになろう。ブロックを指定するためのインデントを忘れずに。

1. Chapter 2.2, 2.3, 4.1.1の補足
   - 2.2 Branching Programs (条件分岐)
   - 2.3 Strings and Input (文字列と入力)
   - 4.1.1 Function Definitions (関数定義)
   - Reserved words, 予約語
2. ペアプロ演習
3. 宿題

講義ページ: http://ie.u-ryukyu.ac.jp/~tnal/2017/prog1/

プログラミング1

(第3回) インタプリタとスクリプトの体験2: 文字列とif文, 関数の利用

1. Chapter 2.2, 2.3, 4.1.1の補足
   - 2.2 Branching Programs (条件分岐)
   - 2.3 Strings and Input (文字列と入力)
   - 4.1.1 Function Definitions (関数定義)
   - Reserved words, 予約語
2. ペアプロ演習
3. 宿題

ユーザ入力を受け取るにはinput()。
型を変更するにはキャストしよう。
(文字列操作には結合・インデックス指定・スライス操作がある。)

講義ページ: http://ie.u-ryukyu.ac.jp/~tnal/2017/prog1/

Functions（関数）とは何か？

- 関数とは、レシピに名前をつけたもの。
- 料理なら、材料を用意して、レシピ通りに誰かが調理したら、料理が仕上がる。
- プログラムなら、必要な入力情報を用意し、関数を呼び出したら（＝コンピュータに実行させる）、想定した出力が得られる。

- 関数の書き方
  1. 実現したい事象について整理し、手順（レシピ）を検討する。
  2. そのレシピに名前をつける。（関数名）
  3. 必要な入力情報（inputs, arguments, 引数；ひきすう）を指定する。
  4. 手順をブロックとして記述する。
- 関数の呼び出し方（実行方法）
  - *関数名(引数)*

関数の例1（戻り値がないケース）

def funcion_name(parameters):
- def: 関数宣言
- funciton_name: 関数名
- parameters: パラメータ
  * argments(引数)とも呼ぶ。
    料理と異なり、不要なら引数なしでもOK。

Tips: 関数名の命名規則
lower_with_under()

```
# 敵に遭遇した際のメッセージを出力する関数
def encount_enemy(name, number):
    print('{0}{1}匹に遭遇した。'.format(name,number))
    print('勇者は逃げ出した。')

encount_enemy('スライム', 2) #関数実行の例（必要な入力を与え、関数を呼び出す）
encount_enemy('ドラキー', 1)
```

indent

block(ブロック)
コードのまとまり。

関数の例2（戻り値があるケース）

```
# 第1引数と第2引数を比較し、大きい方を返す関数。
def max(value1, value2):
    if value1 > value2:
        return value1
    else:
        return value2

# 関数実行の例1。出力されない。
max(10, 5)

# 関数実行の例2。
# step 1:「=」を使い、右辺を評価。
# step 2: 関数呼び出しにより実行され、return文の結果が評価結果になる。
# step 3: 評価結果が、左辺の変数resultに紐付けられる。
result = max(10, 5)
print(result)
```

return文
(1) return文に辿り着いたら、その関数の実行をここで終える。（それ以降のコードは実行しない）
(2) 関数の呼び出し元にreturn指定したオブジェクトを返す。

関数定義から見たblock

プログラミング1

(第3回) インタプリタとスクリプトの体験2: 文字列とif文, 関数の利用

1. Chapter 2.2, 2.3, 4.1.1の補足
   - 2.2 Branching Programs (条件分岐)
   - 2.3 Strings and Input (文字列と入力)
   - 4.1.1 Function Definitions (関数定義)
   - Reserved words, 予約語
2. ペアプロ演習
3. 宿題

オリジナルの**関数**(≒レシピ)
を定義できるようになろう。

講義ページ: http://ie.u-ryukyu.ac.jp/~tnal/2017/prog1/

Reserved words, 予約語

https://goo.gl/4TclUz

- 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	

If文で使う「elif」は次回！

演習

前回の続き: 初めてのペア・プログラミング

プログラミング1

(第3回) インタプリタとスクリプトの体験2: 文字列とif文, 関数の利用

1. Chapter 2.2, 2.3, 4.1.1の補足
   - 2.2 Branching Programs (条件分岐)
   - 2.3 Strings and Input (文字列と入力)
   - 4.1.1 Function Definitions (関数定義)
   - Reserved words, 予約語
2. ペアプロ演習
3. 宿題

if文を使い、条件分岐できるようになろう。ブロックを指定するためのインデントを忘れずに。

ユーザ入力を受け取るにはinput()。
文字列操作には結合・インデックス指定・スライス操作がある。
型を変更するにはキャストしよう。

オリジナルの関数(≒レシピ)を定義できるようになろう。

講義ページ: http://ie.u-ryukyu.ac.jp/~tnal/2017/prog1/

宿題

- 復習: 適宜(これまでの内容)
  - 課題レポート2「3の倍数でアホになろう」 ＊講義ページ参照。
- 予習: 教科書読み
  - 3章
    - 3.1 Exhaustive Enumeration
    - 3.2 For Loops
- 復習・予習(オススメ): paiza, progate

参考文献

- 教科書: Introduction to Computation and Programming Using Python, Revised And Expanded Edition
- Reserved words, https://goo.gl/4TclUz