10　キーボードからのデータの取得

キーボードから入力した文字や数値に応じて，プログラムの結果を変化させる事を考えます。キー入力を取得するには raw_input() と input() という関数を使います。前者は文字列を取得するときに使い，後者は数値を取得するときに使います[*4]。次のファイルを作って実行してみましょう：

- ファイル名：**input1.py**

```
1 # -*- coding: utf-8 -*-
2
3 namae = raw_input('あなたは誰ですか？：')
4 print 'こんにちは', namae, 'さん'
```

上で作ったファイルを実行すると，次のような表示が出ます：

```
1 user@debian:~/Desktop/dataproc1$ python input1.py
2 あなたはだれですか？：
```

そこで，信州花子と入力して Enter キーを押せば

```
1 こんにちは 信州花子 さん
```

と出力されます。上のプログラムは次の手順で実行されました：

1.「あなたはだれですか？：」と画面に表示
2. 変数 namae を作成。キーボード入力を待つ。
3. 入力された文字を変数 namae に代入
4.「こんにちは・・・」を画面に表示

ここで重要な点は，raw_input() で入力された文字列は変数 namae に格納されることです。

入力するものが数値である場合には input() 関数を使います。キー入力を受け付けて，入力した数値の 2 乗を出力するプログラムを作ってみます。

- ファイル名：**input2.py**

```
1 # -*- coding: utf-8 -*-
2
3 num = input('数値を入力してください：')
4 print '入力された数値の２乗は', num**2, 'です'
```

これを実行すると次のようになります：

```
1 user@debian:~/Desktop/dataproc1$ python input2.py
2 数値を入力してください：123
3 入力された数値の２乗は 15129 です
```

上の二つを合わせて次のような BMI を計算するプログラムを作ってみましょう：

- ファイル名：**bmi1.py**

```
1 # -*- coding: utf-8 -*-
2
3 namae = raw_input('あなたの名前を入力してください：')
4 shintyo = input('あなたの身長は何センチですか？：')
```

[*4] Python3 では raw_input() は廃止されて，input() に置き換わりました。

```
5 weight  = input('あなたの体重は何キログラムですか？：')
6 bmi = 10000*weight/(shintyo**2)
7 print namae, 'さんのＢＭＩは', bmi, 'です'
```

プログラムを実行して自分の BMI を計算してみましょう。

11　論理型と比較演算子

数学では，正しい命題は真，間違った命題は偽であるといいます。Python でも真偽値というものがあり，条件が正しいかどうかで真・偽の値が決まります。これらは条件に応じて処理を変化させるときに使います。

Python では真を `True`，偽を `False` で表します。これらは予約語であり特別な意味を持ちます。文法に注意しながら，インタラクティブシェルでの例を見てください：

```
1 >>> a = 3      #aに3を代入
2 >>> a == 3     #aは3だろうか？
3 True           #a==3は正しい！
4 >>> a == 2     #aは2だろうか？
5 False          #a==2は正しくない
6 >>> a > 2
7 True
8 >>> 3 != 2     #3は2に等しくないだろうか？
9 True           #3と2は異なるので，上の条件は真
```

この例のように，二つの数値を比較して真か偽かの条件を調べる事ができます。上の例で『`==`，`>`，`!=`』という記号を用いました。これらはデータを比較するときに用いるので**比較演算子**と呼ばれています。数値に対して使える使える比較演算子には次があります：

数値に対して使える比較演算子

`==`　`!=`　`<`　`>`　`<=`　`>=`

これらは，それぞれ伝統的な数学記号 $=$，$\neq$，$<$，$>$，$\leq$，$\geq$ に対応しています。

不等号と等号の順番は英語の”less than or equal” の順番と同じだと憶えましょう。『`!=`』は”not equal” と憶えます。

`True` と `False` には論理演算 `and`，`or`，`not` を行うことが出来ます。

```
1 >>> a = True      # aをTrueとする
2 >>> b = False     # bをFalseとする
3 >>> a and b      # aかつbは？
4 False
5 >>> a or b       # aまたはbは？
6 True
7 >>> not a        # aの否定は？
8 False
```

データが与えられたときに，それらの関係に対して真偽値を得ることが出来ます。

リストや集合などコレクションに対して使える比較演算子もあります：

リストや集合などに対して使える比較演算子

`==`　`!=`　`in`

`==`と`!=` の説明は不要でしょう。`in` は次のように用います：

```
>>> a = [3,6,5,1]        #aをリスト[3,6,5,1]とする
>>> 5 in a               #5はaの中にいるだろうか？
True
>>> 7 in a               #7はaの中にいるだろうか？
False
```

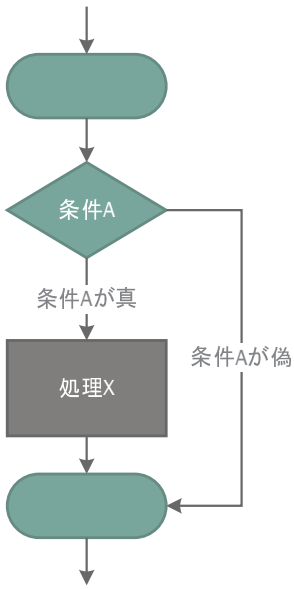
次のようにして，型を調べる type と組み合わせて使うことも出来ます：

```
>>> a = [3,5,6,1]
>>> type(a)
<type 'list'>
>>> type(a) == list
True
>>> type(a) == tuple
False
```

12　条件分岐

12.1　if 文

条件 (True か False) に応じて，処理を変化させるときに if 文という決められた文法を用います。if 文の手順は次のチャートの通りです：



これを Python では次の文法で書きます：

Python での if 文の構造

```
if 条件A:
    [ 処理X ]
    [ 処理X ]
    [ 処理X ]
```

- 条件 A が True であれば処理 X が行われ，False であれば処理 X は行われません。
- if の行の行末のコロン『：』を忘れずに！
- 処理 X 数行にわたる場合は，上のようにインデントを揃える。

ここで処理 X の前のインデント（字下げ）が文法上重要な役割を果たします。処理 X が数行にわたるときに，どこまでが処理 X であるかはインデントによって判断されます。インデントは単なるスペースで見えないのですが，インデントが揃った部分が，処理されるプログラムの塊（ブロック）であると判断されます：

if 文を使った例題をやってみましょう。次のアルゴリズムを Python プログラムにします。

(1) 整数 a を入力させる。
(2) a が偶数なら，『a is even』とプリントしてから，a を半分にする。
(3) a が奇数なら何もしない。

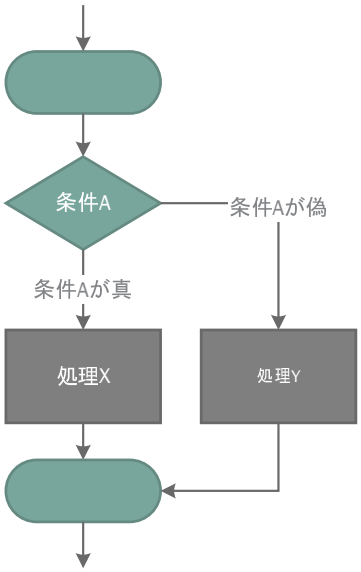
- ファイル名：**if1.py**

```
1 # -*- coding: utf-8 -*-
2 a = input('Input integer a: ')
3
4 if a%2 == 0:          # もしaが偶数なら
5     print 'a is even'
6     a = a/2           # aを半分にする。行頭のインデントは重要！！！
7
8 print a               # aの値をプリント
```

上のプログラムの 4〜6 行目が if 文です。実行していくつかの数値を入れて動作を試してみましょう。

12.2 if-else 文

次に，少し複雑な条件分岐を考えます。条件が真のときには処理 X，偽のときには別の処理 Y がしたいとしましょう。



もちろん，これは上で説明した if 文だけで書くことができます。つまり

```
if 条件A:        # 条件Aが真なら
    処理X        # 処理Xを行い，偽なら行わない。
if not 条件A:    # 条件Aが偽なら
    処理Y        # Yを行い，そうでなければ行わない。
```

と書くだけです。しかし，次に説明する if-else 文を使って，より明示的に書くこともできます。

Python での if-else 文の構造

```
if 条件A:
    処理X
else:
    処理Y
```

- 条件 A が `True` であれば，処理 X が行い，`False` であれば処理 Y を行う。
- ここでも処理 X, 処理 Y はインデントによって判断されます。

`if-else` 文を使った例題をやってみましょう。次のアルゴリズムを考えます。

(1) 整数 `a` を入力させる。
(2) もし `a` が偶数ならば，`a` を半分にする。
(3) もし `a` が奇数ならば，`a` を `3*a+1` にする
(4) `a` の値を表示する。

これをプログラムで書いてみましょう。

- ファイル名：**if2.py**

```
# -*- coding: utf-8 -*-
a = input('a?:')

if a%2 == 0:     # もしaが偶数なら
    a = a/2      #aを半分にする
else:            # そうでなければ
```

```
7     a = 3*a + 1   #aを3a+1にする
8
9  print a          #aの値をプリント
```

このプログラムを実行していくつかの数値を入れて動作を試してみましょう。

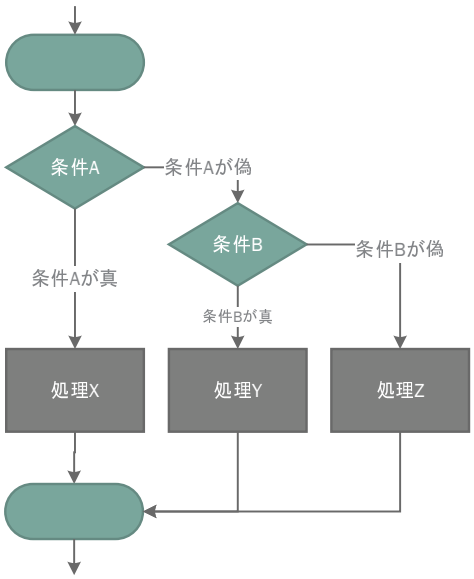
12.3 if-elif-else 文

さらに条件分岐を記述するためには if-elif-else 文を使います。elif は else if の略です。

if-elif-else 文の構造

```
1  if 条件A:
2      処理X
3  elif 条件B:
4      処理Y
5  else:
6      処理Z
```

- 条件 A が真の場合に処理 X を行います。
- 条件 A が偽で条件 B が True のときに，処理 Y を行います。
- 条件 A も条件 B も成り立たないときに，処理 Z を行います。
- さらに条件 C，条件 D，・・・と条件が続く場合は `elif` で処理します。



さらに `elif` を追加して，`if-elif-elif-elif-else` のように条件を増やすことができます。

12.4 複雑な if 文の例

`if-elif-else` 文を使って，与えた西暦が閏年かどうかを判定するプログラムを作りたい。グレゴリウス暦では閏年は次のように決められている：

- 西暦年が 4 で割り切れる年は閏年とする。
- ただし西暦年が 4 で割り切れる年でも，100 で割り切れる年は閏年としない。

- ただし西暦年が 4 で割り切れ，100 でも割り切れる年でも 400 で割り切れる年は閏年とする。

上の閏年の定め方の文章には，『ただし』が多くあって，論理構造がわかりにくく，そのままではプログラムにはしづらい。そこで閏年の条件を次のように書き直します：

1. 西暦年が 400 で割り切れるならば，それは閏年である。
2. 上以外のとき，西暦年は 100 で割り切れるならば，それは閏年ではない。
3. 上以外のとき，西暦年が 4 で割り切れるならば，それは閏年である。
4. 上のどれにも当てはまらないとき，その年は閏年ではない。

これを実現する Python のプログラムは次のようになります。

- ファイル名：**uruuQ.py**

```
# -*- coding: utf-8 -*-
year = input('西暦を入力:')    # 入力した数値を変数yearに代入する

if year % 400 == 0:      # もしyearを400で割った余りが0なら
    print year, 'は閏年です。'
elif year % 100 == 0:    # そうでないとき，もしyearが100で割り切れたら
    print year, 'は閏年ではありません。'
elif year % 4 == 0:      # そうでないとき，もしyearが4で割り切れたら
    print year, 'は閏年です。'
else:                    # 上の全ての条件に当てはまらないとき
    print year, 'は閏年ではありません。'
```

13　練習問題

13.1　問題：BMI 計算プログラム

名前，身長，体重を入力させ，そこから BMI を計算・表示し，その値に応じてやせ・肥満度の結果を出すプログラムを作りたい。プログラムは次の手順を行うものとする。

(1) 名前を入力させ，変数名 `namae` に代入
(2) 身長・体重を入力させ，それぞれ `shintyo`, `weight` に代入
(3) bmi を表示
(4) もし bmi<18.5 なら，「やせ気味です」と表示
(5) そうでないとき，bmi<25.0 なら，「ふつうです」と表示
(6) そうでないとき，bmi<30 なら，「太り気味です」と表示
(7) うえのどちらでもないとき「太りすぎです」と表示

上の手順が実行されるように，つぎのプログラムの『*****』の部分を推測してプログラムを完成させなさい。

- ファイル名：**bmi2.py**

```
# -*- coding: utf-8 -*-

namae = raw_input('あなたの名前を入力してください：')
shintyo = input('あなたの身長は何センチですか？：')
weight  = input('あなたの体重は何キログラムですか？：')

bmi = 10000.0*weight/(shintyo**2)
```

```
8  bmi = round(bmi,1)     #bmiの値を小数点以下2桁目を四捨五入する
9
10 print namae, 'さんのＢＭＩは', bmi, 'で',
11
12 if **********:
13     ***** 'やせ気味です。'
14 elif **********:
15     ***** 'ふつうです。'
16 ***************:
17     ***** '太り気味です。'
18 ****:
19     ***** '太りすぎです。'
```