1. Python で何ができるの？

1.1. Python って何だろう？

1.1.1. Python って何？

- 今から 20 年以上前につくられたプログラミング言語
- 人工知能（機械学習）やビックデータの解析などの研究によくつかわれる

1.1.2. Python の 3 つの特徴

特徴１：シンプルなプログラム
プログラムの「処理のまとまり」を「インデント（字下げ）」を使って書くのが特徴

```
print(xxx, yyy)
if(a > b):
  print(xxx, yyy) #インデント
  print(zzz)      #インデント
def function():
  for d in list:  #インデント
    print(d)      #インデント * 2
print(xxx, yyy)
```

特徴２：ライブラリが豊富
数学モジュールや通信モジュールなどの **標準モジュール** 以外にも、画像処理や機械学習などの **外部ライブラリ** といったものもある。

特徴３：試行錯誤がしやすい
Python はプログラムを書いたらすぐに実行できる **インタープリタ言語** 。
試行錯誤がやりやすい言語だから、新しいものを生み出す開発者にも向いている。

それに対し、C や Java などは **コンパイラ言語** といってプログラムを実行ファイルに変換するので手間がかかるが、その分実行速度がはやいという利点がある。

2. Python を触ってみよう

2.1. アイドルで始めよう！

> **IDLE（アイドル）**
>
> 手軽に Python を実行するためのアプリ。起動すればすぐに使えるので、
> Python の動作確認をしたり初心者が勉強したりするのに向いている。

命令を実行してみる

```
# 書式 -> print()
print("値")
print("値 1", "値 2")
```

```
>>> print(1 + 1)

# out -> 2
```

色々な演算子

記号	計算
`+`	足し算
`-`	引き算
`*`	掛け算
`/`	割り算
`//`	割り算（小数部分を切り捨て）
`%`	割り算のあまり

2.2. 文字も表示させてみよう

文字を表示するためには、 **文字列の両側を「'（シングルクォーテーション）」か「"（ダブルクォーテーション）」で囲んで記述する。**

```
>>> print("Hello!")

# out -> Hello!
```

文字列と数値の組み合わせ

```
>>> print("答えは", 10+ 20)

# out -> 答えは 30
```

おみくじプログラム

`omikuji.py`

```
import random
kuji = ["大吉", "中吉", "小吉"]
print(random.choice(kuji))

# out -> 大吉 or 中吉 or 小吉
```

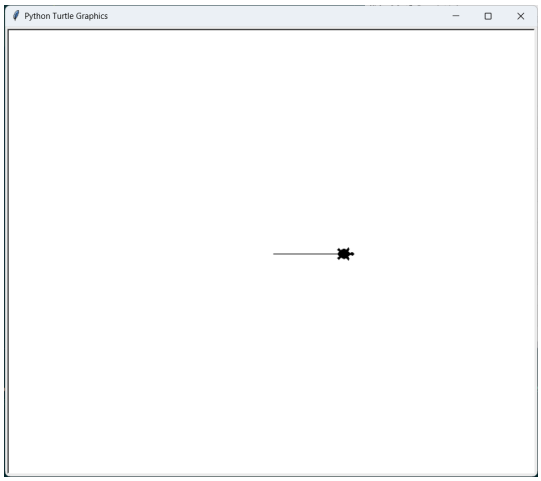
BMI 値計算プログラム

`bmi.py`

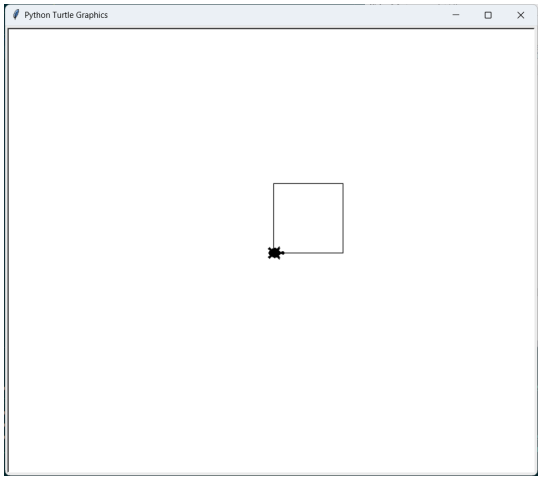
```
h = float(input("身長何 cm ですか?")) / 100.0
w = float(input("体重何 kg ですか?"))
bmi = w / (h * h)
print("あなたの BMI1 値は、", bmi, "です。")

# out -> 身長何 cm ですか?  171
#        体重何 kg ですか?  64
#        あなたの BMI 値は、21.887076365377382 です。
```

2.3. カメでお絵描きしてみよう

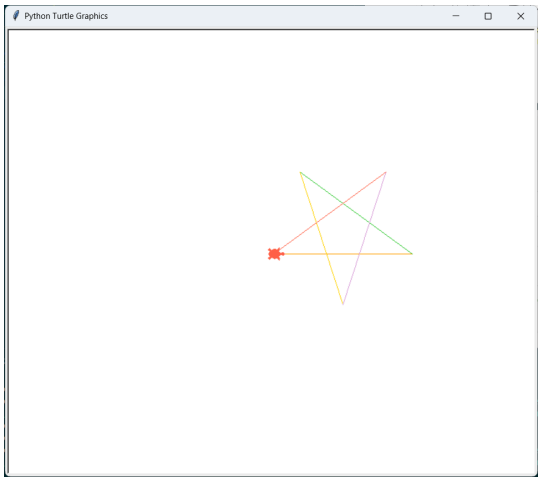
```
from turtle import * #turtle モジュールをインポート/読み込む
shape("turtle")      # カメの形にする
forward(100)         #100px 前進
done()               # 終了
```




```
from turtle import *     #turtle モジュールをインポート/読み込む
shape("turtle")          # カメの形にする
for i in range(4):       #4 回繰り返す
    forward(100)         #100px 前進
    left(90)             #90 度左に回転
done()
```



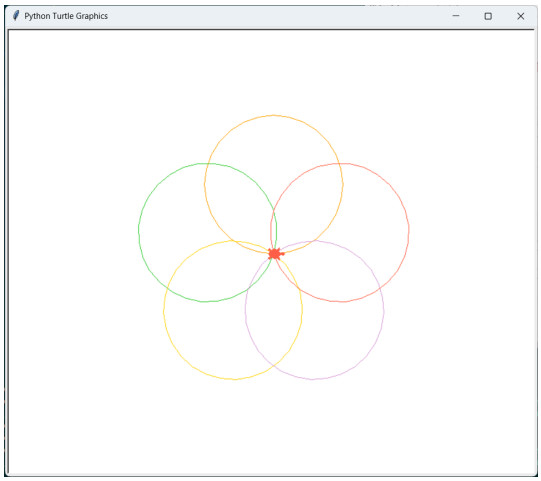

```
from turtle import *                        #turtle モジュールをインポート/読み込む
shape("turtle")                                  # カメの形にする
col = ["orange", "limegreen", "gold", "plum", "tomato"]   # 色のリスト
for i in range(5):                                  #5 回繰り返す
    color(col[i])                                # 色を変える
    forward(200)                               #200px 前進
    left(144)                                  #144 度左に回転
done()                                  # 終了
```



i について

`range(5) -> i` は、0 から 4 までの 5 つの数字を生成する。

```
from turtle import *
shape("turtle")
col = ["orange", "limegreen", "gold", "plum", "tomato"]
for i in range(5):
    color(col[i])
    circle(100)
    left(72)
done()
```



3. プログラムって何ができるの？

.

- プログラムは、どれだけ難しそうでも「順次」「分岐」「反復」の三つの組み合わせでできている。

3.1. データは入れ物に入れて使う

分類	データ型	説明
数値（整数型）	int	個数や順番に使う
数値（浮動小数点型）	float	小数
文字列型	str	文字（列）を扱うときに使う
ブール型	bool	True（真）か False（偽）かの二択の時に使う

3.2. 文字列の操作を覚えよう

3.2.1. 文字数を調べる

メソッド	説明
`len()`	文字列の長さを取得
`count()`	指定した文字列の出現回数を取得
`find()`	指定した文字列の位置を取得
`replace()`	指定した文字列を置換
`split()`	指定した文字列で分割

3.2.2. 文字列の一部を取りだす

メソッド	説明
`[n]`	n 番目の文字を取得

メソッド	説明
`[n:m]`	n 番目から m 番目までの文字を取得
`[n:]`	n 番目から最後までの文字を取得
`[:m]`	最初から m 番目までの文字を取得

こ	ん	に	ち	は	。	私	は	パ	イ	ソ	ン	で	す	。
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
−15	−14	−13	−12	−11	−10	−9	−8	−7	−6	−5	−4	−3	−2	−1

```
s = "こんにちは。私はパイソンです。"
print(s[0])   # こ
print(s[6:12]) # 私はパイソン
print(s[-3:])  # です。
```

3.2.3. 文字列の途中で改行するには？

```
print("こ\nん\nに\nち\nは\n。")
```

`\n`を使うことで、文字列の途中で改行することができる。

3.3. データ型を変換する

3.3.1. データ型を変換する

```
a = "100"
print(a + 100) # エラー
print(int(a) + 100) # 200
```

3.3.2. 変換できないときはエラーになる

```
a = "こんにちは"
print(int(a)) # エラー
```

書式：isinstance()

`isinstance()`は、変数が指定したデータ型かどうかを調べる関数。

```
a = "100"
print(isinstance(a, int)) # False
print(isinstance(a, str)) # True
```

3.4. たくさんのデータはリストに入れて使う

3.4.1. リストの作り方

```
a = [1, 2, 3, 4, 5]
print(a) # [1, 2, 3, 4, 5]
```

3.5. もしも～なら実行する

```
a = 10
if a > 5:
    print("a は 5 より大きい")
else:
    print("a は 5 以下")
```

比較演算子	説明
`==`	等しい
`!=`	等しくない
`>`	より大きい
`<`	より小さい
`>=`	以上
`<=`	以下

3.5.1. そうでないとき

```
a = 10
if a > 5:
    print("a は 5 より大きい")
```

```
elif a == 5:
    print("a は 5")
else:
    print("a は 5 以下")
```

3.6. 繰り返し処理をする

基本構造

```
for i in range(5):
    print(i)
```

```
a = [1, 2, 3, 4, 5]
for i in a:
    print(i)
```

3.6.1. リストのすべての要素について繰り返す

```
a = [1, 2, 3, 4, 5]
for i in a:
    print(i)
```