pf-10. クラス定義，オブジェクト生成，メソッド，属性

（Python 入門）

URL: https://www.kkaneko.jp/pro/pf/index.html

金子邦彦

① ソフトウエア設計の理解

② 抽象化

オブジェクトとメソッド

- **オブジェクト**：コンピュータでの**操作や処理の対象となるもの**

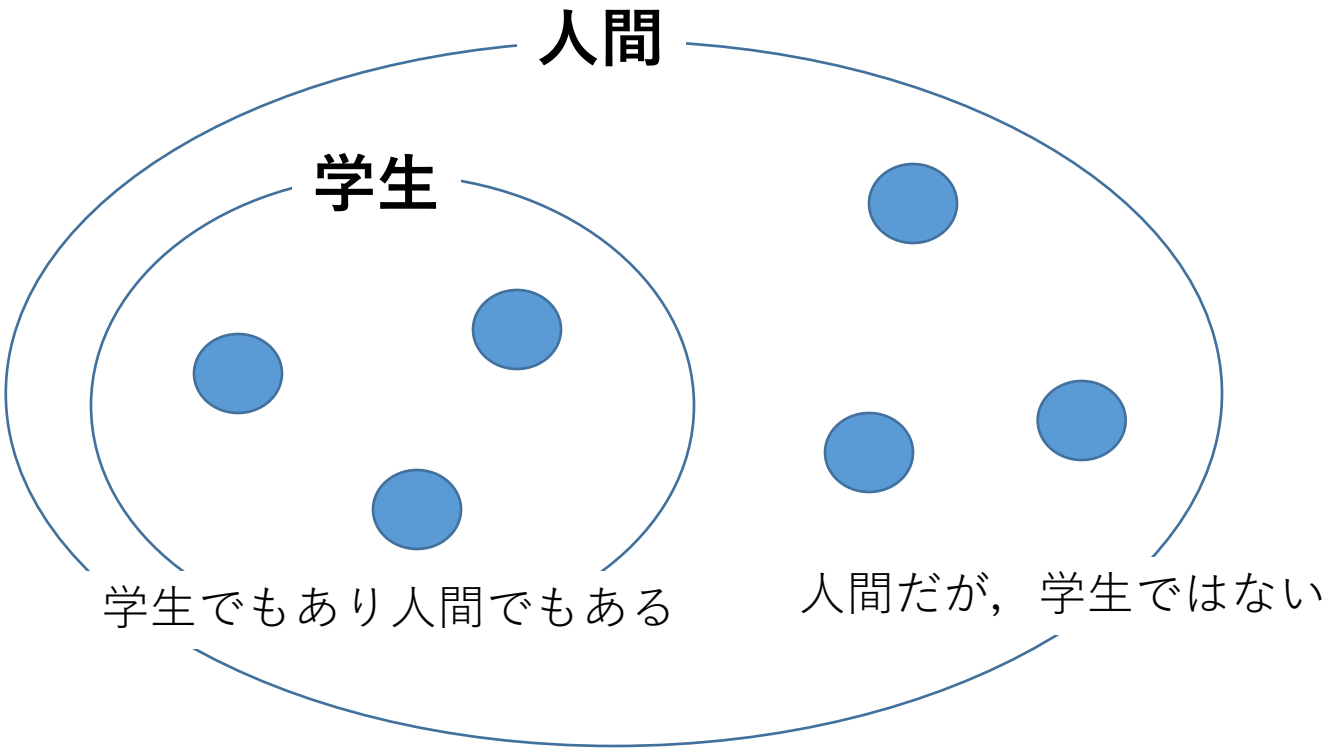
hero.moveDown()	**hero　オブジェクト** **moveDown()　メソッド** 間を「.」で**区切って**いる

- **メソッド: オブジェクト**に属する機能や操作．オブジェクトがもつ能力に相当する
- **引数：メソッド**が行う操作の詳細に関する情報，**メソッド**呼び出しのときに、引数を指定できる

hero.attack("fence", 36, 26)

クラスとオブジェクト

クラスは，**同じ種類のオブジェクトの集まり**と考えることができる

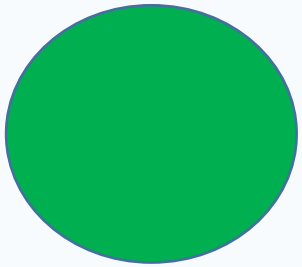


同じクラスの2つのオブジェクト

オブジェクト

半径 1，場所（8, 10）
色 blue

オブジェクト



半径 3，場所（2, 4）
色 green

クラス Ball

- **オブジェクト**は
  半径，場所，色などの属性
  を持つことができる

クラス定義の例

```
class Ball:
    def __init__(self, x, y, r, color):
        self.x = x
        self.y = y
        self.r = r
        self.color = color
    def printout(self):
        print(self.x, self.y, self.r, self.color)
```

クラス名: Ball

属性: x, y, r, color

メソッド: __init__, printout

オブジェクト生成の際に，メソッド __init__ が自動で実行される

クラス定義の例

```
class Ball:
    def __init__(self, x, y, r, color):
        self.x = x
        self.y = y
        self.r = r
        self.color = color
    def printout(self):
        print(self.x, self.y, self.r, self.color)
```

クラス名: Ball

メソッド: __init__

メソッド: printout

クラス名: Ball

属性: x, y, r, color

メソッド: __init__, printout

__init__ は，オブジェクトが生成される際に自動的に呼び出されるメソッド

クラスの利用

```
class Ball:
    def __init__(self, x, y, r, color):
        self.x = x
        self.y = y
        self.r = r
        self.color = color
    def printout(self):
        print(self.x, self.y, self.r, self.color)
```

- **オブジェクト生成**

Ball クラスのオブジェクト生成

- **属性アクセス**

属性 x, y, r, color の値の取得や変更

- **メソッド呼び出し**

メソッド printout の呼び出し

Ball クラスのオブジェクト生成

```
class Ball:
    def __init__(self, x, y, r, color):
        self.x = x
        self.y = y
        self.r = r
        self.color = color
    def printout(self):
        print(self.x, self.y, self.r, self.color)
```

- Python では，「**クラス名()**」の形式で**オブジェクト生成**を行う

```
a = Ball(8, 10, 1, "blue")
b = Ball(2, 4, 3, "green")
```

- オブジェクト生成の際に，メソッド __init__ が自動で実行される

Ball クラスの属性アクセス

```
class Ball:
    def __init__(self, x, y, r, color):
        self.x = x
        self.y = y
        self.r = r
        self.color = color
    def printout(self):
        print(self.x, self.y, self.r, self.color)
```

- Python では，「**オブジェクト . 属性名**」の形式で**属性にアクセス**する

```
print(a.x, a.y, a.r, a.color)
```

- 属性値の変更は「**オブジェクト . 属性名** = 値や式」の形式で行う

```
a.x = 18
a.y = 20
```

Ball クラスのメソッド呼び出し

```
class Ball:
    def __init__(self, x, y, r, color):
        self.x = x
        self.y = y
        self.r = r
        self.color = color
    def printout(self):
        print(self.x, self.y, self.r, self.color)
```

- Python では，「**オブジェクト . メソッド名 ()**」の形式で**メソッドを呼び出す**

```
a.printout()
```

メソッド定義内での属性アクセス，メソッド呼び出し

- **メソッド定義内**の属性アクセスは，「**self . 属性名**」の形式
- **メソッド定義内**のメソッド呼び出しは，「**self . メソッド名 ()** 」の形式

```
class Ball:
    def __init__(self, x, y, r, color):
        self.x = x
        self.y = y
        self.r = r
        self.color = color
    def printout(self):
        print(self.x, self.y, self.r, self.color)
```

trinket

- **Trinket** は**オンライン**の **Python、HTML** 等の**学習サイト**
- 有料の機能と無料の機能がある
- **自分が作成した Python プログラムを公開し、他の人に実行してもらうことが可能**（そのとき、書き替えて実行も可能）

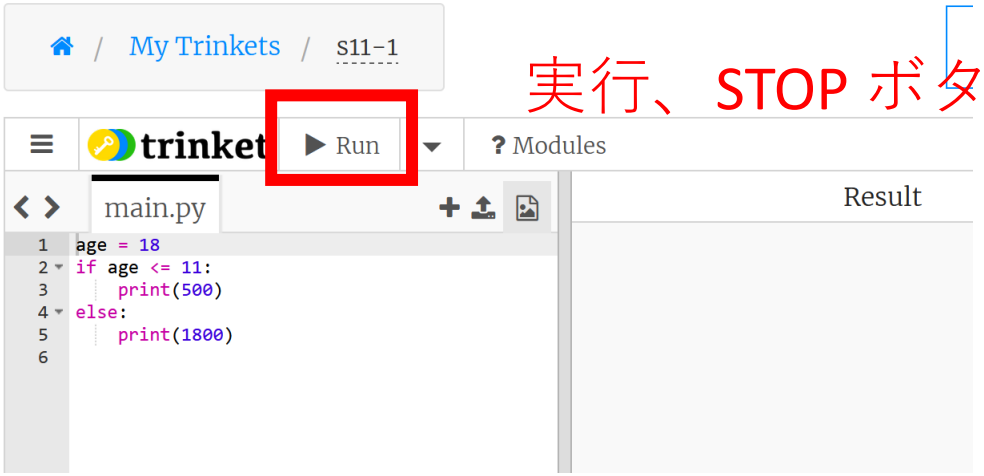
- **Python の標準機能**を登載、その他、次のモジュールやパッケージがインストール済み

math, matplotlib.pyplot, numpy, operator, processing, pygal, random, re, string, time, turtle, urllib.request

trinket でのプログラム実行

- **trinket は Python, HTML などのプログラムを書き実行できる**サイト
- https://trinket.io/python/0fd59392c8

のように、違うプログラムには違う URL が割り当てられる



- 実行が開始しないときは、「**実行ボタン**」で**実行**
- ソースコードを**書き替えて再度実行**することも可能

演習

資料：16

【トピックス】

- **クラス定義**
- **class**
- **オブジェクト生成**
- **メソッド呼び出し**

① trinket の次のページを開く

https://trinket.io/python/6fdf17af8f

② 実行結果が，次のように表示されることを確認

このプログラムは，**オブジェクト a, b を生成**する．そして，**メソッド printout を呼び出し**て，属性値を表示させる

```
class Ball:
    def __init__(self, x, y, r, color):
        self.x = x
        self.y = y
        self.r = r
        self.color = color
    def printout(self):
        print(self.x, self.y, self.r, self.color)

a = Ball(8, 10, 1, "blue")
b = Ball(2, 4, 3, "green")

a.printout()
b.printout()
```

```
Powered by trinket
(8, 10, 1, 'blue')
(2, 4, 3, 'green')
```

演習

資料：18

【トピックス】

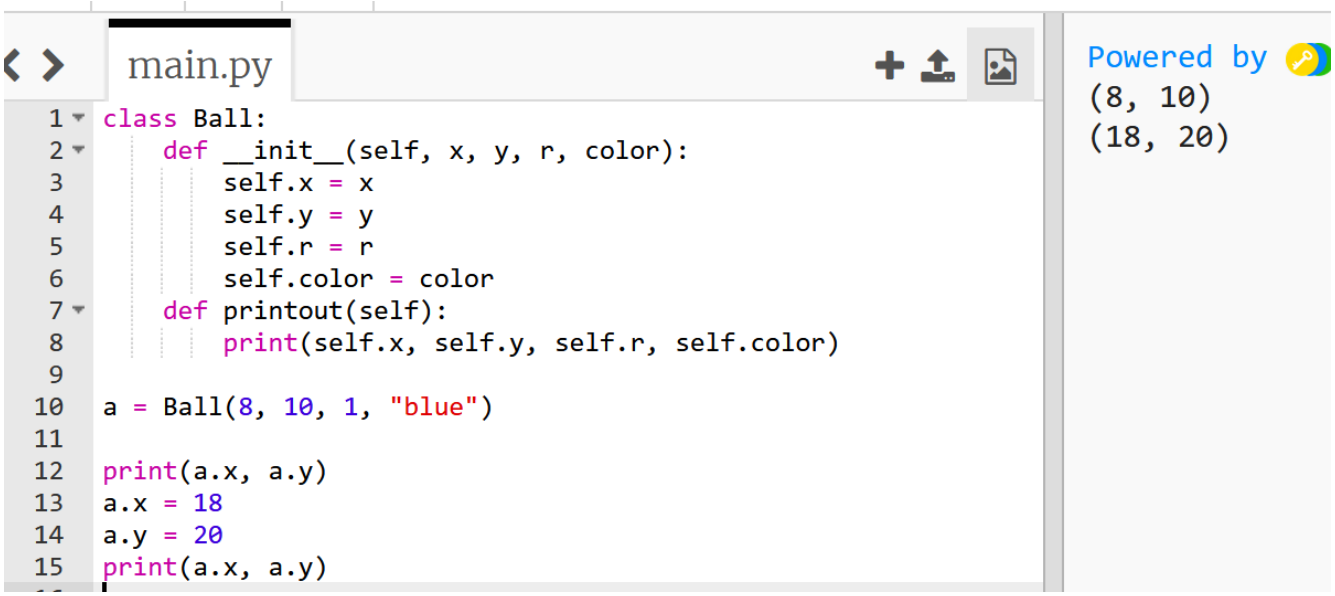
- **クラス定義**
- **class**
- **オブジェクト生成**
- **属性アクセス**

① trinket の次のページを開く

https://trinket.io/python/2fd4a420de

② 実行結果が，次のように表示されることを確認

このプログラムは，**オブジェクト a を生成**する．そして，**属性 x, y にアクセス**する



① ソフトウエア設計の理解

クラス、**メソッド**、**属性**を使用することで、**プログラムは読みやすく**なり、**将来必要となる変更も行いやすく**なります。更には、これらの要素を適切に使用することで、全体的なソフトウェアアーキテクチャと設計への理解を深め、**効率的で再利用可能なコードを作成**することができます。

② 抽象化

同じ種類のオブジェクトをクラスにまとめるという**抽象化**を行うことで、**現実世界の概念**を**プログラムに取り入れる**ことが容易になるとともに、**プログラムが読みやすく再利用しやすく**なります。また、抽象化により、システム全体の柔軟性を向上させることができます。

全体まとめ

- **クラス**は**同じ種類のオブジェクトの集まり**
- **属性**は**オブジェクトの状態**を表す
- **メソッド**は**オブジェクトに属する機能や操作.**
- 次の**クラス定義**により、「Ball」クラスの**オブジェクトの生成**、「color」や「x」などの**属性アクセス**、情報を表示する「printout」**メソッドへのアクセス**ができるようになる

```
class Ball:
    def __init__(self, x, y, r, color):
        self.x = x
        self.y = y
        self.r = r
        self.color = color
    def printout(self):
        print(self.x, self.y, self.r, self.color)
```

- 次のプログラムに**より x = 8, y = 10, r = 1, color = "blue" のボールが生成**され，**表示**が行われる

```
a = Ball(8, 10, 1, "blue")
a.printout()
```