POLYTECHNIQUE MONTRÉAL

TECHNOLOGICAL UNIVERSITY

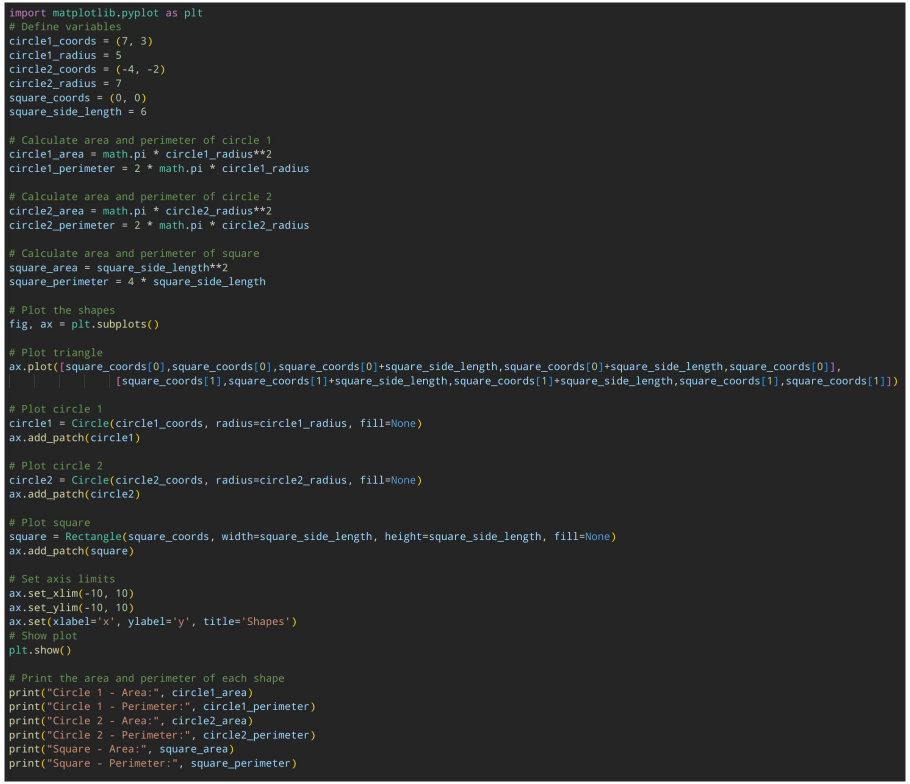
TP2

Rappel COO

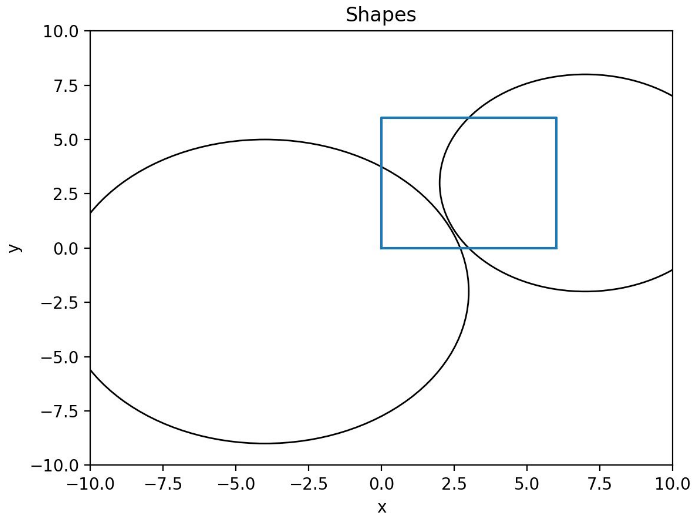
Code Orienté Objet

- **Modularité** : Plein de petit code simple plutôt qu’un unique long
- **Réutilisabilité** : Les classes et les objets peuvent être réutilisés.
- **Encapsulation** : Encapsulation de l'information et comportements -> +Stable +Sur
- **Abstraction** : Simplifie, tâche complexe -> Facile à utiliser
- **Héritage** : Réduire la redondance
- **Polymorphisme** : Même fonction pour différent objet -> flexible et plus facile à étendre.

Example

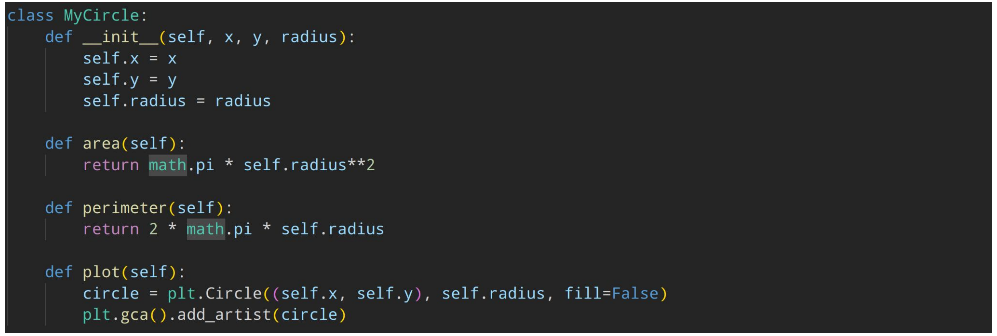


```
import matplotlib.pyplot as plt
# Define variables
circle1_coords = (7, 3)
circle1_radius = 5
circle2_coords = (-4, -2)
circle2_radius = 7
square_coords = (0, 0)
square_side_length = 6

# Calculate area and perimeter of circle 1
circle1_area = math.pi * circle1_radius**2
circle1_perimeter = 2 * math.pi * circle1_radius

# Calculate area and perimeter of circle 2
circle2_area = math.pi * circle2_radius**2
circle2_perimeter = 2 * math.pi * circle2_radius

# Calculate area and perimeter of square
square_area = square_side_length**2
square_perimeter = 4 * square_side_length

# Plot the shapes
fig, ax = plt.subplots()

# Plot triangle
ax.plot([square_coords[0],square_coords[0],square_coords[0]+square_side_length,square_coords[0]+square_side_length,square_coords[0]],
                 [square_coords[1],square_coords[1]+square_side_length,square_coords[1]+square_side_length,square_coords[1],square_coords[1]])

# Plot circle 1
circle1 = Circle(circle1_coords, radius=circle1_radius, fill=None)
ax.add_patch(circle1)

# Plot circle 2
circle2 = Circle(circle2_coords, radius=circle2_radius, fill=None)
ax.add_patch(circle2)

# Plot square
square = Rectangle(square_coords, width=square_side_length, height=square_side_length, fill=None)
ax.add_patch(square)

# Set axis limits
ax.set_xlim(-10, 10)
ax.set_ylim(-10, 10)
ax.set(xlabel='x', ylabel='y', title='Shapes')
# Show plot
plt.show()

# Print the area and perimeter of each shape
print("Circle 1 - Area:", circle1_area)
print("Circle 1 - Perimeter:", circle1_perimeter)
print("Circle 2 - Area:", circle2_area)
print("Circle 2 - Perimeter:", circle2_perimeter)
print("Square - Area:", square_area)
print("Square - Perimeter:", square_perimeter)
```

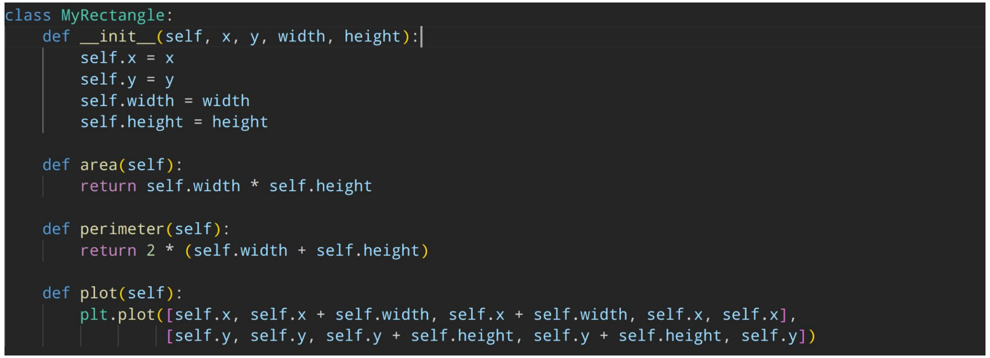


```
Circle 1 - Area: 78.53981633974483
Circle 1 - Perimeter: 31.41592653589793
Circle 2 - Area: 153.93804002589985
Circle 2 - Perimeter: 43.982297150257104
Square - Area: 36
Square - Perimeter: 24
```

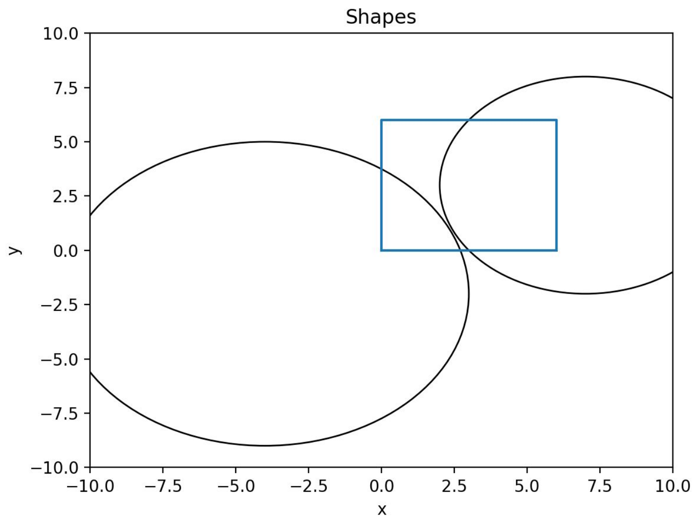
Example



```python
class MyCircle:
    def __init__(self, x, y, radius):
        self.x = x
        self.y = y
        self.radius = radius

    def area(self):
        return math.pi * self.radius**2

    def perimeter(self):
        return 2 * math.pi * self.radius

    def plot(self):
        circle = plt.Circle((self.x, self.y), self.radius, fill=False)
        plt.gca().add_artist(circle)
```



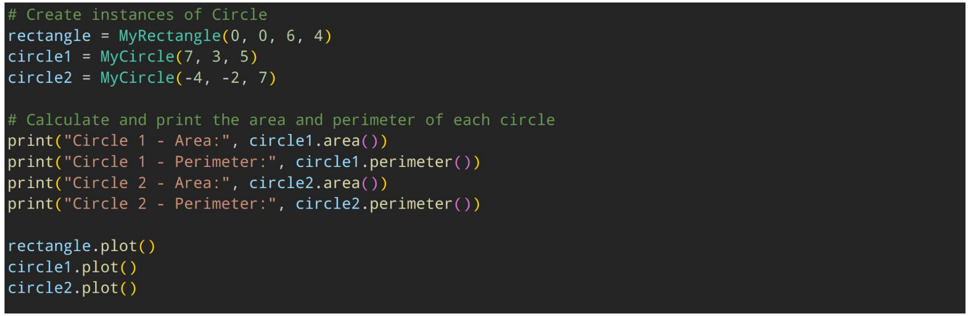
```python
class MyRectangle:
    def __init__(self, x, y, width, height):
        self.x = x
        self.y = y
        self.width = width
        self.height = height

    def area(self):
        return self.width * self.height

    def perimeter(self):
        return 2 * (self.width + self.height)

    def plot(self):
        plt.plot([self.x, self.x + self.width, self.x + self.width, self.x, self.x],
                 [self.y, self.y, self.y + self.height, self.y + self.height, self.y])
```



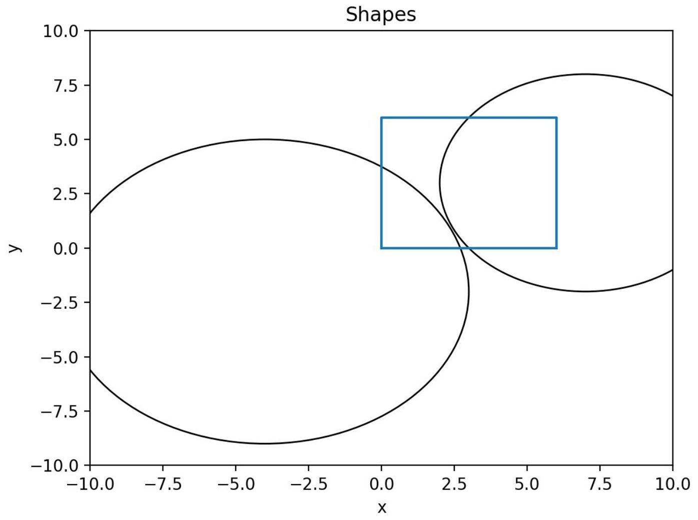
```
Circle 1 - Area: 78.53981633974483
Circle 1 - Perimeter: 31.41592653589793
Circle 2 - Area: 153.93804002589985
Circle 2 - Perimeter: 43.982297150257104
Square - Area: 36
Square - Perimeter: 24
```

POLYTECHNIQUE MONTRÉAL
TECHNOLOGICAL UNIVERSITY

Example



```
# Create instances of Circle
rectangle = MyRectangle(0, 0, 6, 4)
circle1 = MyCircle(7, 3, 5)
circle2 = MyCircle(-4, -2, 7)

# Calculate and print the area and perimeter of each circle
print("Circle 1 - Area:", circle1.area())
print("Circle 1 - Perimeter:", circle1.perimeter())
print("Circle 2 - Area:", circle2.area())
print("Circle 2 - Perimeter:", circle2.perimeter())

rectangle.plot()
circle1.plot()
circle2.plot()
```



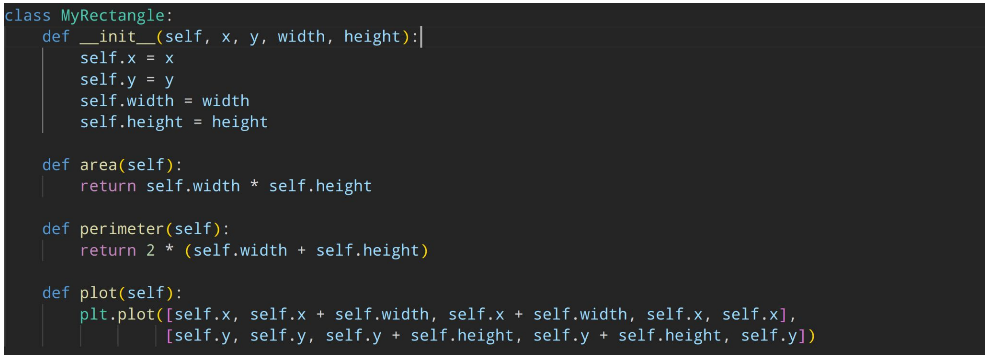
```
Circle 1 - Area: 78.53981633974483
Circle 1 - Perimeter: 31.41592653589793
Circle 2 - Area: 153.93804002589985
Circle 2 - Perimeter: 43.982297150257104
Square - Area: 36
Square - Perimeter: 24
```

Nomenclature

Classe :	Téléphone
Un objet	Mon Téléphone (Instance de classe)
Des attributs (caractéristique)	Marque / Modèle / % batterie / ….
Des méthodes (fonction)	Allumer / éteindre / …

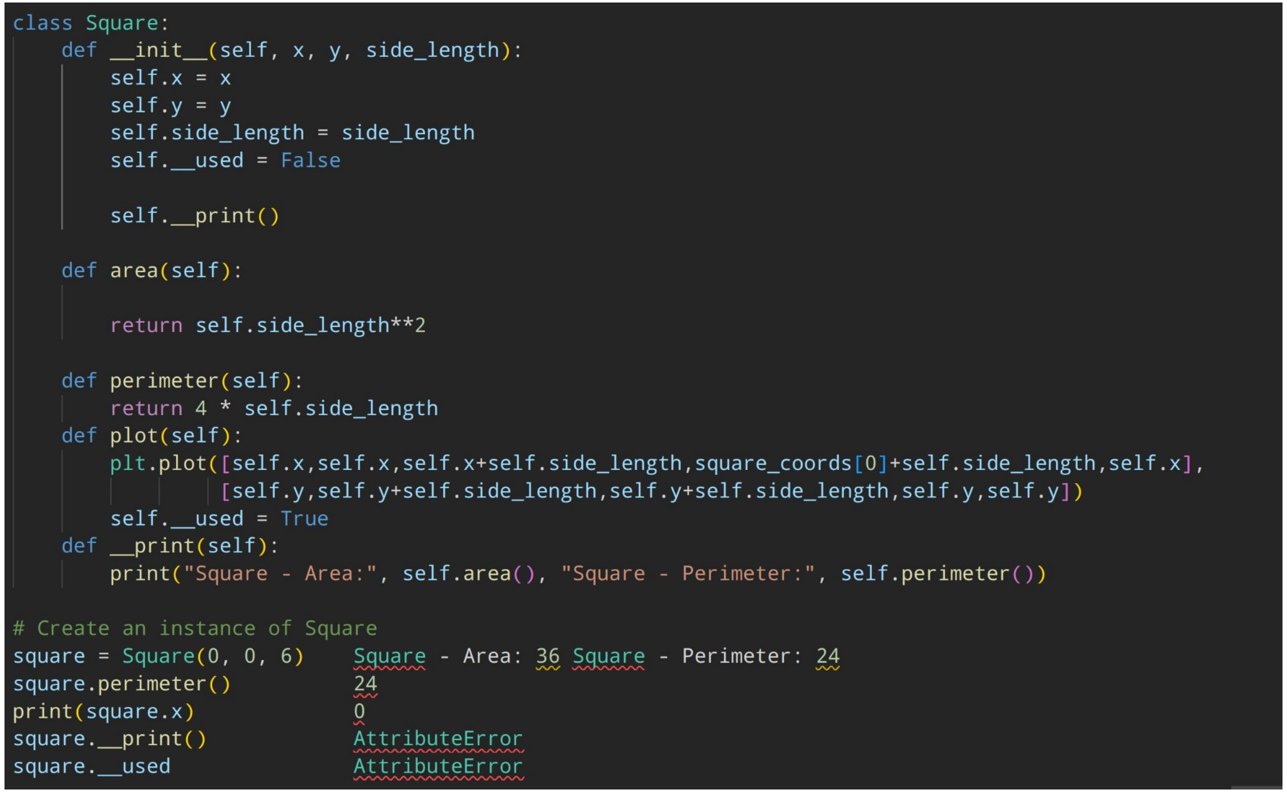
Constructeur

- Information nécessaire
- Crée une instance
- Rectangle = MyRectangle(0,0,5,6)



```python
class MyRectangle:
    def __init__(self, x, y, width, height):
        self.x = x
        self.y = y
        self.width = width
        self.height = height

    def area(self):
        return self.width * self.height

    def perimeter(self):
        return 2 * (self.width + self.height)

    def plot(self):
        plt.plot([self.x, self.x + self.width, self.x + self.width, self.x, self.x],
                 [self.y, self.y, self.y + self.height, self.y + self.height, self.y])
```

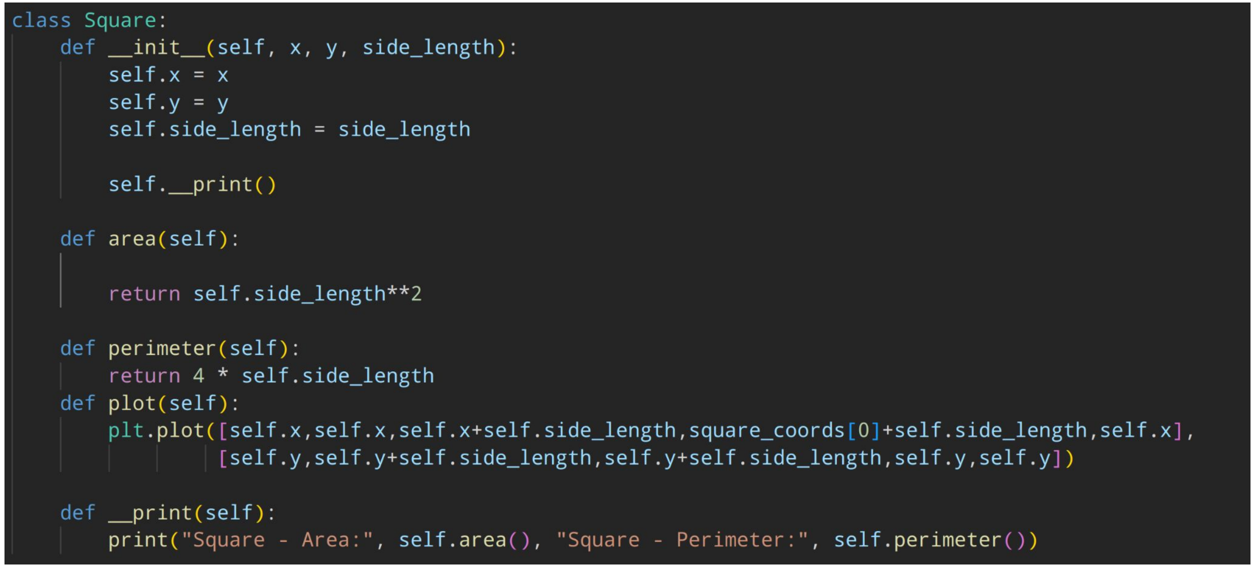
Privé VS Public



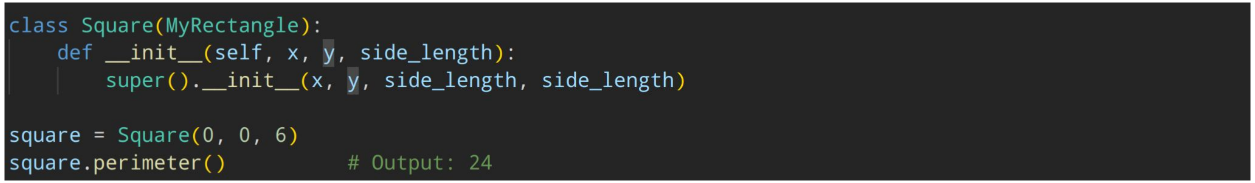
```
class Square:
    def __init__(self, x, y, side_length):
        self.x = x
        self.y = y
        self.side_length = side_length
        self.__used = False

        self.__print()

    def area(self):

        return self.side_length**2

    def perimeter(self):
        return 4 * self.side_length
    def plot(self):
        plt.plot([self.x,self.x,self.x+self.side_length,square_coords[0]+self.side_length,self.x],
                 [self.y,self.y+self.side_length,self.y+self.side_length,self.y,self.y])
        self.__used = True
    def __print(self):
        print("Square - Area:", self.area(), "Square - Perimeter:", self.perimeter())

# Create an instance of Square
square = Square(0, 0, 6)     Square - Area: 36 Square - Perimeter: 24
square.perimeter()           24
print(square.x)              0
square.__print()             AttributeError
square.__used                AttributeError
```

POLYTECHNIQUE
MONTRÉAL
TECHNOLOGICAL
UNIVERSITY

Héritage



```python
class Square:
    def __init__(self, x, y, side_length):
        self.x = x
        self.y = y
        self.side_length = side_length

        self.__print()

    def area(self):

        return self.side_length**2

    def perimeter(self):
        return 4 * self.side_length
    def plot(self):
        plt.plot([self.x,self.x,self.x+self.side_length,square_coords[0]+self.side_length,self.x],
                 [self.y,self.y+self.side_length,self.y+self.side_length,self.y,self.y])

    def __print(self):
        print("Square - Area:", self.area(), "Square - Perimeter:", self.perimeter())
```



```python
class Square(MyRectangle):
    def __init__(self, x, y, side_length):
        super().__init__(x, y, side_length, side_length)

square = Square(0, 0, 6)
square.perimeter()          # Output: 24
```

Exercice

Class counting de ultralytics

https://github.com/ultralytics/ultralytics/blob/main/ultralytics/solutions/object_counter.py