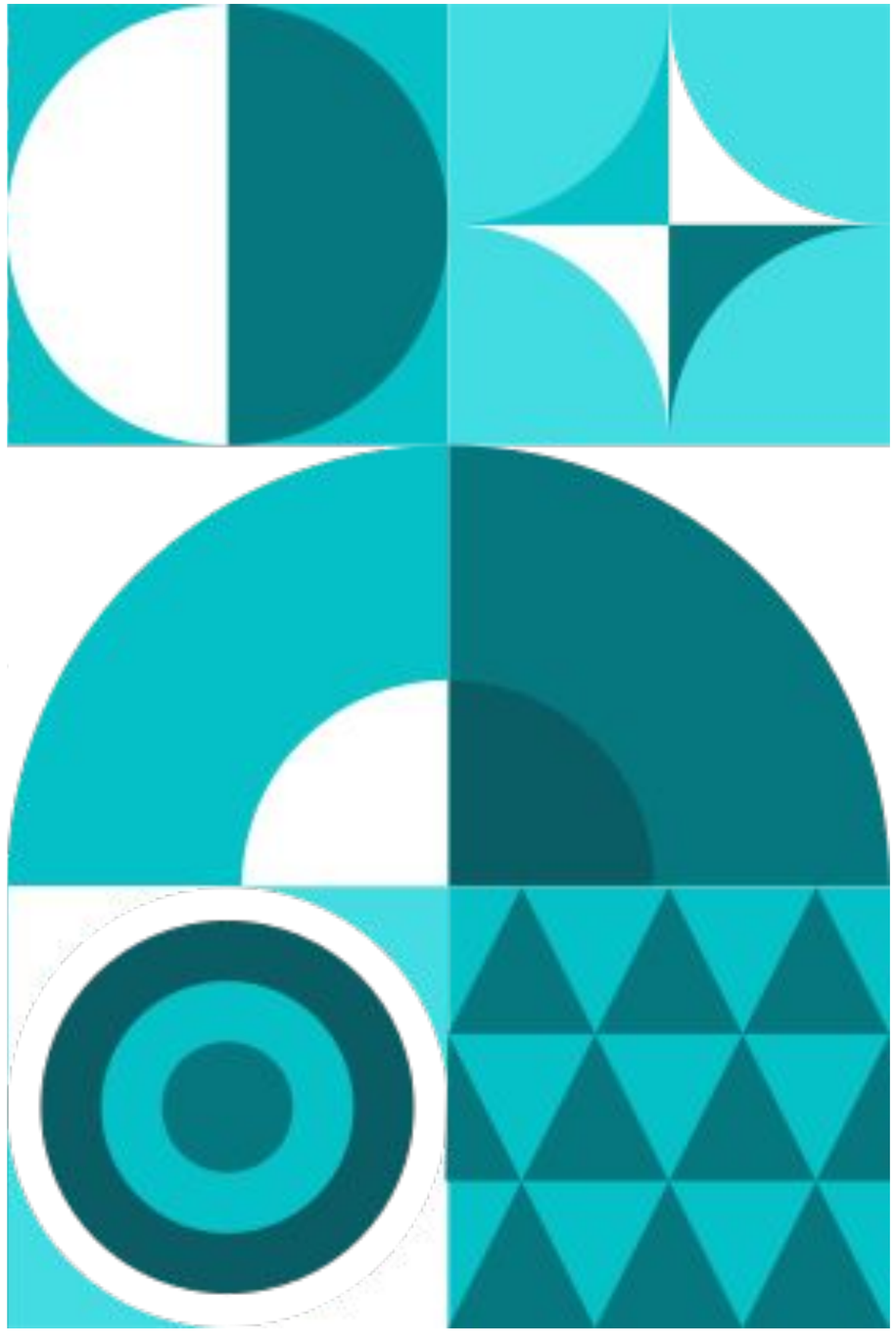


codingstudio

DAY 6

INTRO TO PROGRAMMING

BY: EVAN AURELRIUS

codingstudio

Timeline

1. OBJECT AND CLASS
2. OOP
3. FILE PROCESSING

codingstudio

OBJECT AND CLASS

codingstudio

Python Class

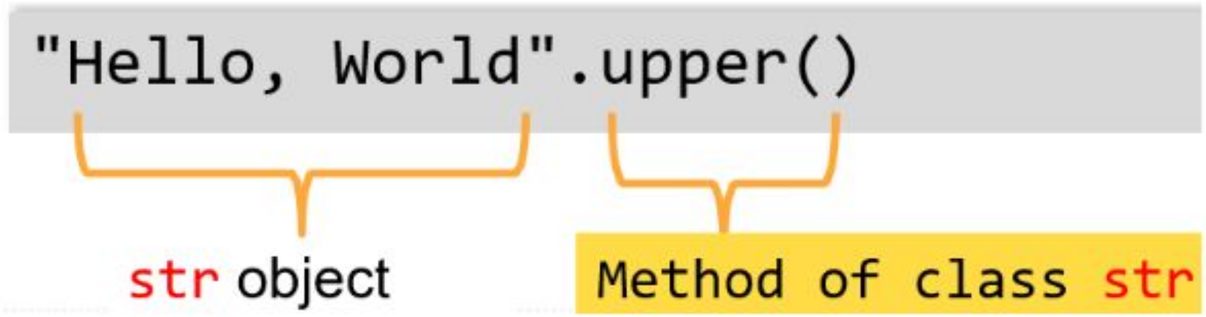
Class adalah blueprint atau template untuk membuat objek. Class mendefinisikan atribut (data) dan metode (fungsi) yang dimiliki oleh objek.

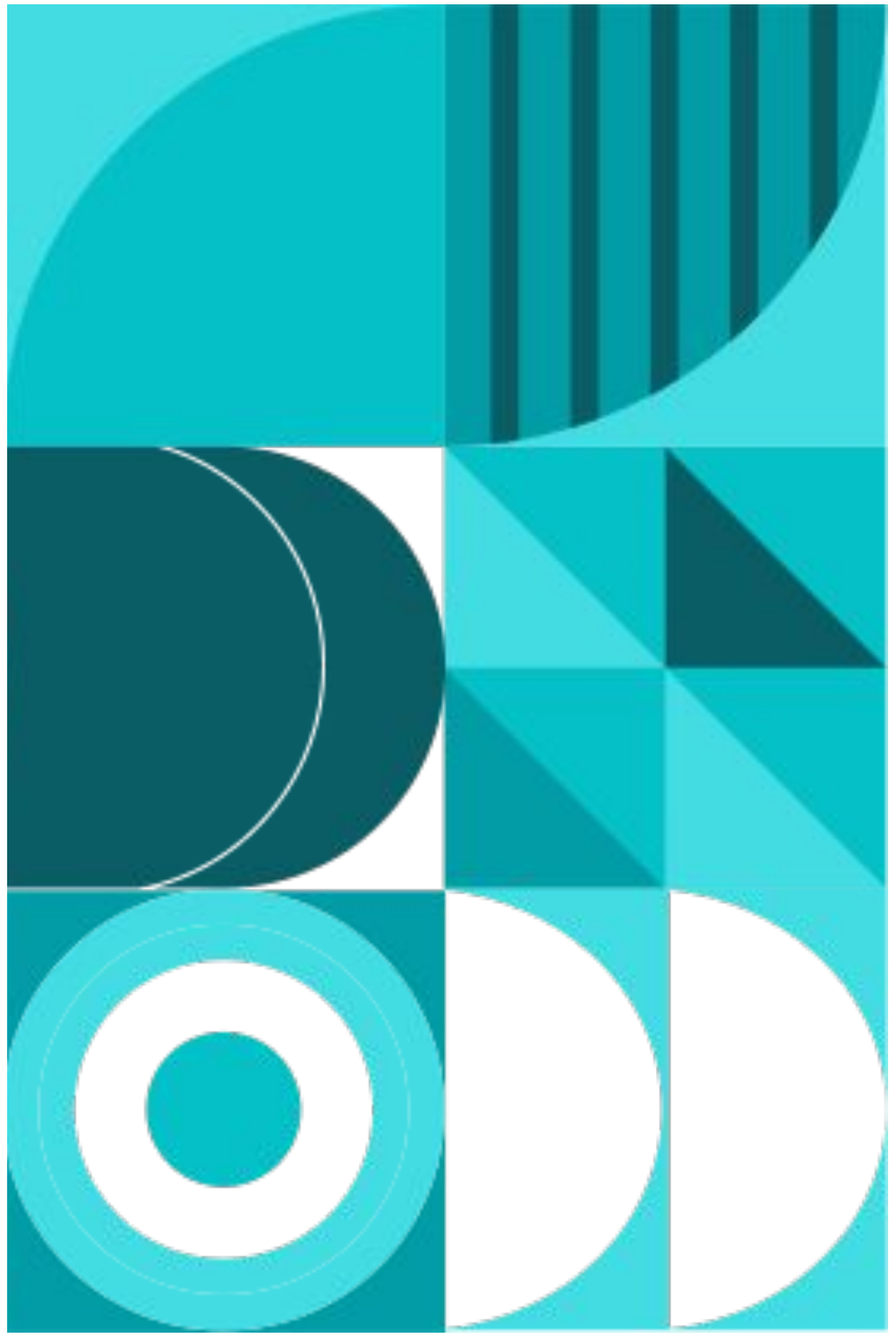
Dengan kata lain, class adalah sebuah cetak biru (blueprint) yang mendeskripsikan bagaimana suatu objek akan berperilaku dan apa yang bisa dilakukan oleh objek tersebut.

codingstudio

PYTHON CLASS

Contoh:
Class str
mendeskripsikan apa
yang dapat objek string
lakukan





codingstudio

PYTHON CLASS

Di bawah ini adalah contoh penggunaan method class yang illegal

```
["Hello", "World"].upper()
```

Alasannya, karena method .upper() tidak dimiliki oleh class list. Method .upper() hanya tersedia untuk class str.

Contoh method yang berlaku pada class list:

```
["Hello", "World"].pop()
```

codingstudio

PYTHON OBJECT

Object adalah instansiasi dari class.

Dengan menggunakan class sebagai blueprint, kita bisa membuat berbagai objek yang memiliki data dan perilaku yang ditentukan oleh class tersebut.

Setiap objek adalah entitas independen yang memiliki karakteristik unik berdasarkan class yang menjadi template-nya.

codingstudio

Python Object

- Class mendefinisikan properti dan prilaku dari Object.
- Object adalah **instance** dari Class
- Kita dapat membuat banyak instances dari sebuah Class
- Membuat instance dari sebuah Class disebut instantiation
- Class menjelaskan apa instance tersebut ketahui atau lakukan

codingstudio

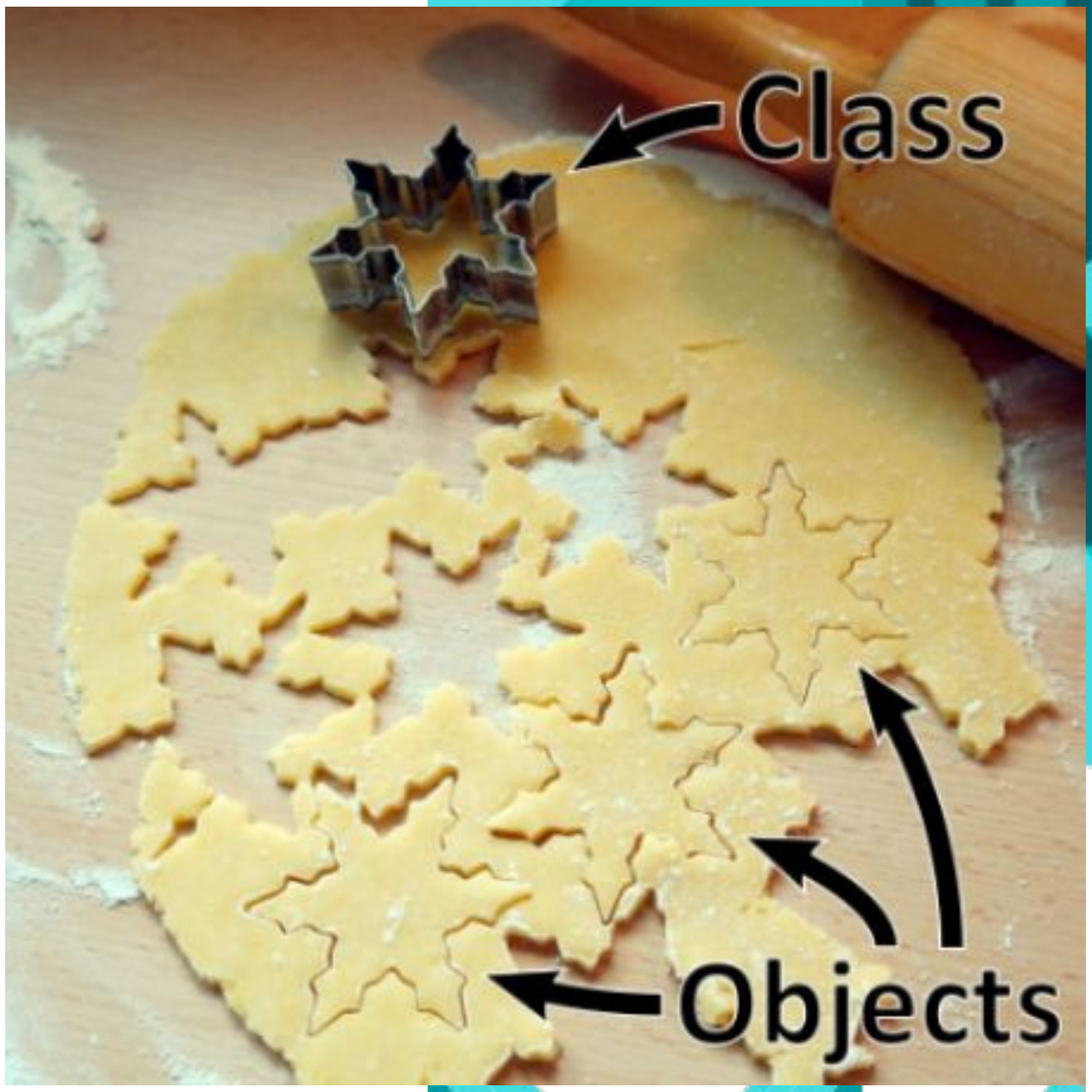
Python Object

- “abc” adalah Object dari Class str
- 2024 adalah Object dari Class int
- [1,2,3] adalah Object dari Class list

- Setiap Object memiliki atribut, contohnya Object Mahasiswa memiliki atribut seperti: Nama, NIM, dll.
- Setiap Object akan merespon pada methods. (ex: memanggil())

codingstudio

CLASS VS OBJECT

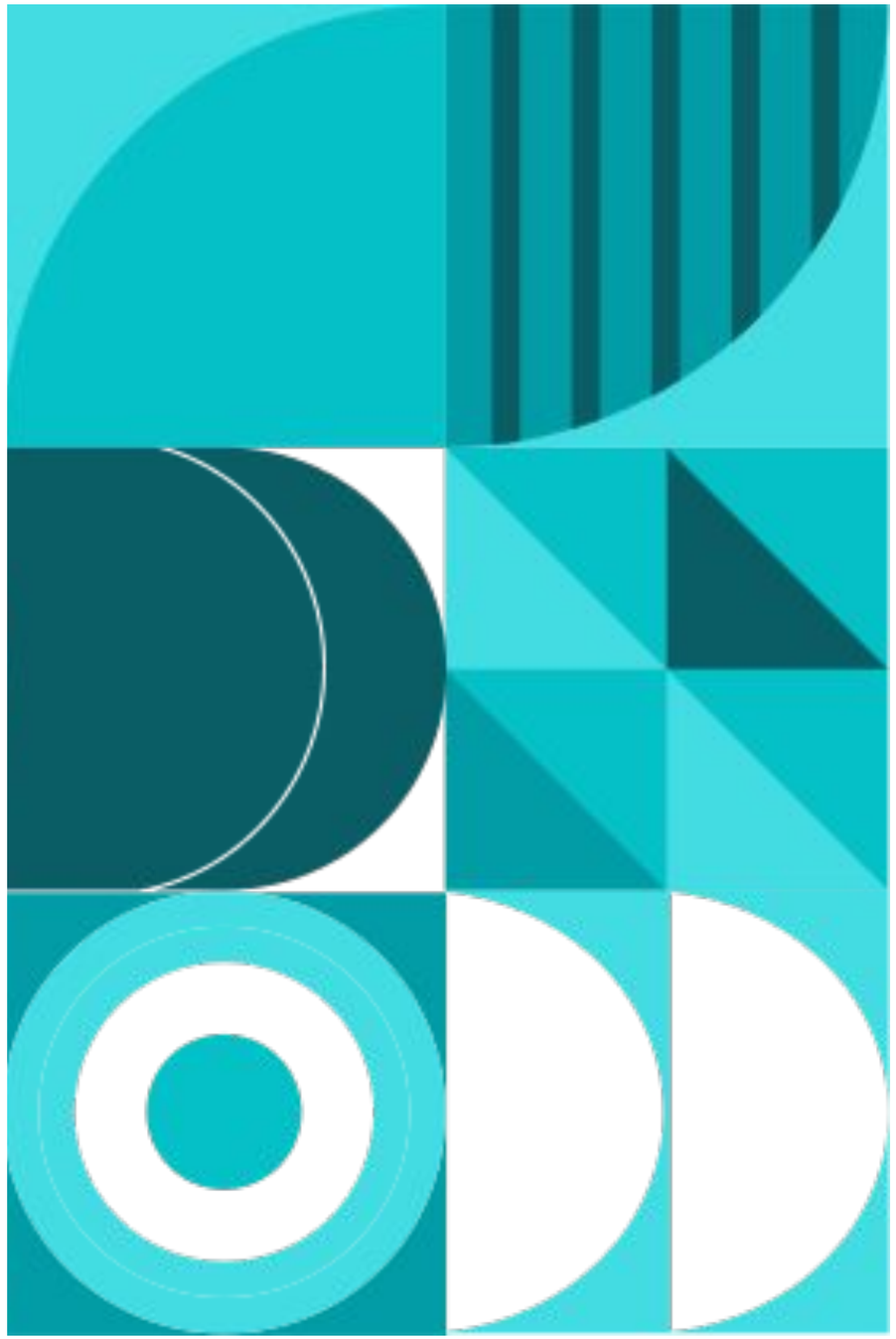


codingstudio

STANDAR PENAMAAN CLASS

- Setiap kata dimulai dengan huruf kapital.
- (ex: Car, UniversityStudent)

- T**anpa** garis underscore (ex: University_Student)



MENDEFINISIKAN CLASS

codingstudio

Class memiliki bentuk umum:

```
class <class-name> (<superclass>, ...):
    <variable and method definitions>
```

- Method terlihat mirip dengan Function, meletakkan Function di dalam Class akan membuatnya menjadi sebuah Method.
- Parameter pertama dari sebuah Method selalu self, yang berarti diri sendiri atau objek yang sedang melakukan Method.

Contoh Pembuatan Class dan Object

codingstudio

```
import math
class Circle:
    def __init__(self, radius = 1):
        self._radius = radius

    def __str__(self):
        return "Circle with radius {}".format(self._radius)

    def getPerimeter(self):
        return 2 * self._radius * math.pi

    def getArea(self):
        return math.pi * (self._radius ** 2)

    def setRadius(self, radius):
        self._radius = radius
```

```
>>> myCircle = Circle()
>>> print(myCircle)
Circle with radius 1
>>> myCircle.getPerimeter()
6.283185307179586
>>> myCircle.getArea()
3.141592653589793
>>> myCircle.setRadius(5)
>>> print(myCircle)
Circle with radius 5
```

codingstudio

OOP

codingstudio

PRINSIP OOP

Encapsulation

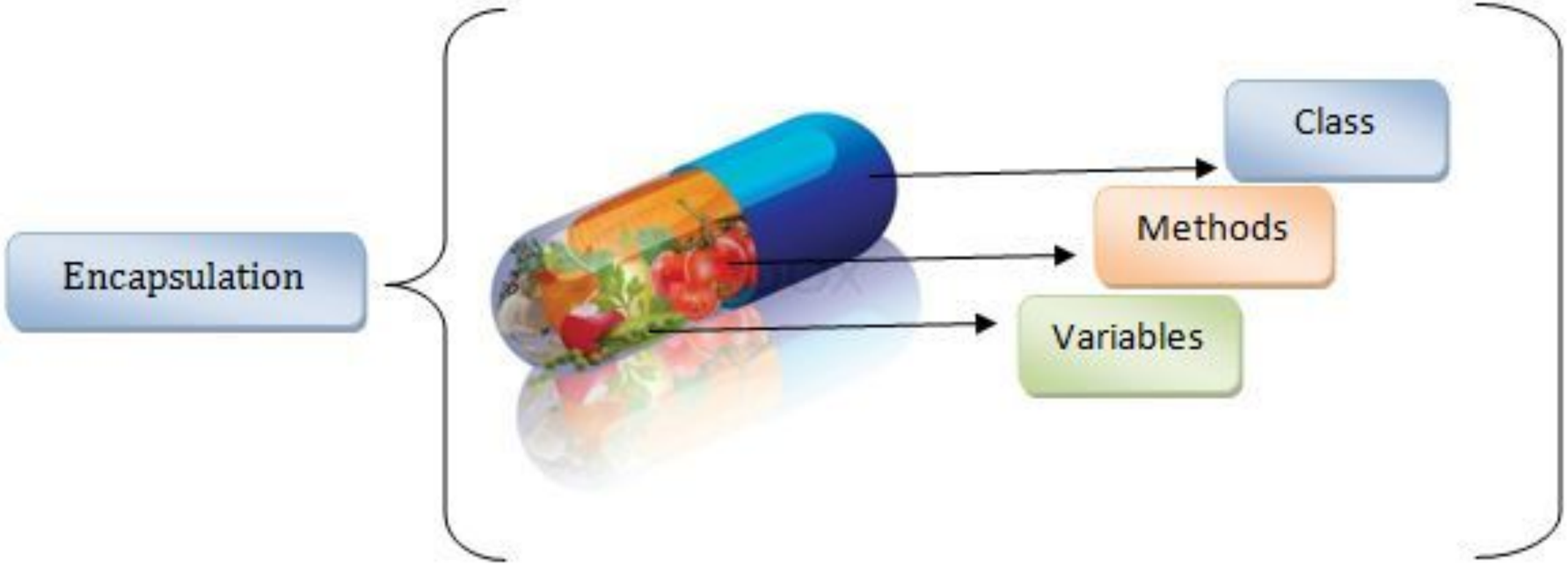
hiding implementation details to make the program clearer and more easily modified

Modularity

making multiple modules first and then linking and combining them

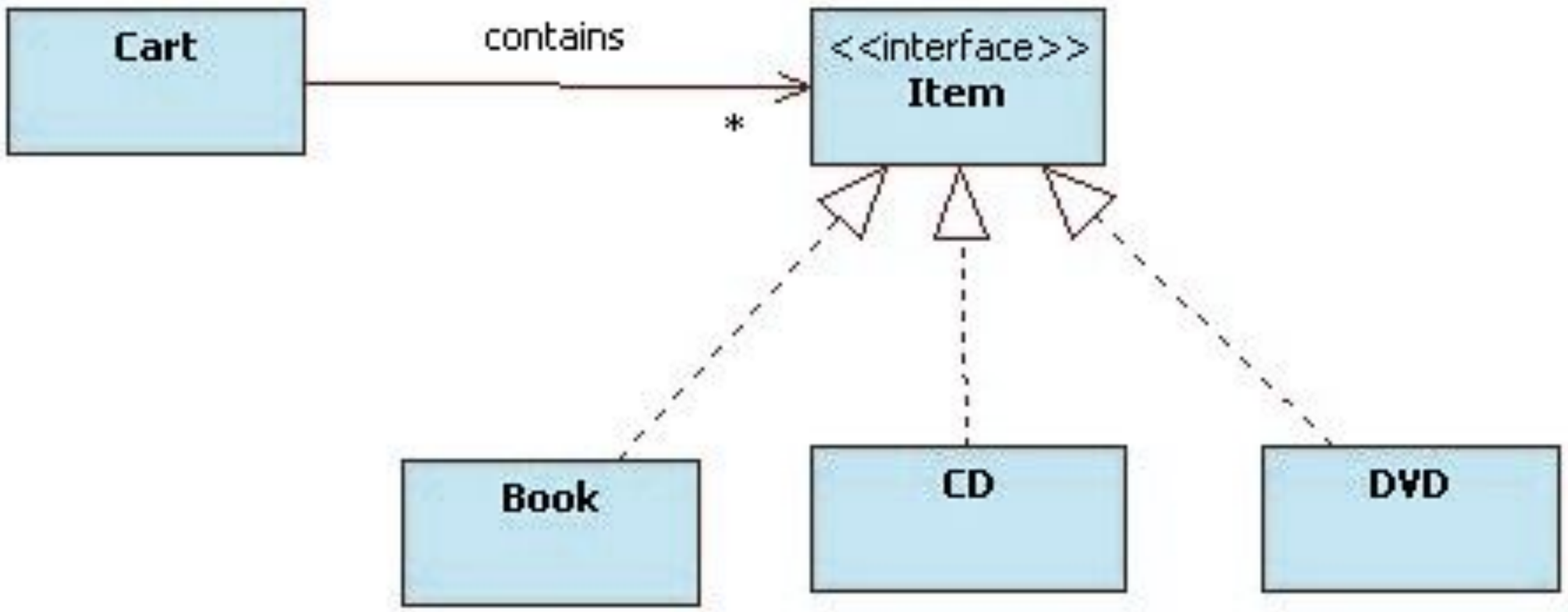
codingstudio

ENCAPSULATION



codingstudio

Modularity



PRINSIP OOP

codingstudio

Inheritance

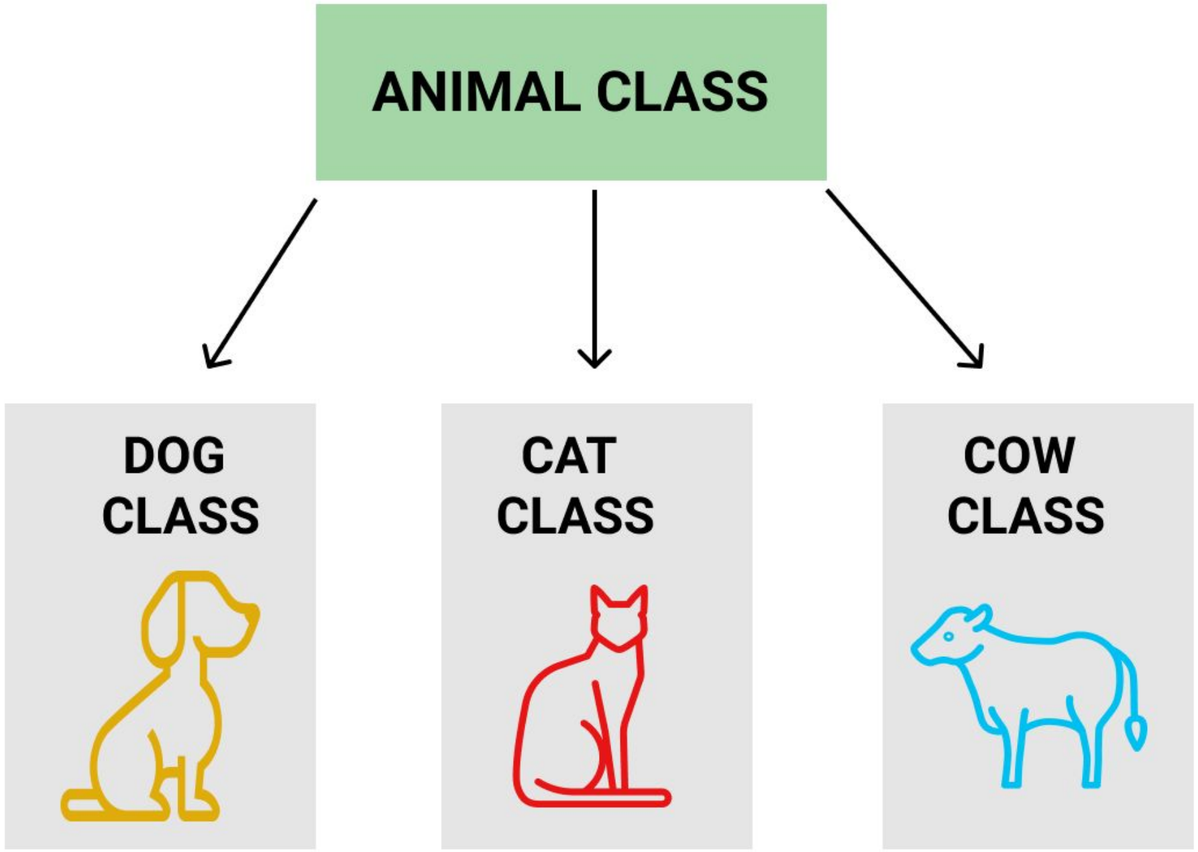
The ability to derive a new class from one or more existing classes. Inherited variables and methods of the original (parent) class are available in the new (child) class as if they were declared locally.

Polymorphism

An object-oriented technique by which a reference that is used to invoke a method can result in different methods being invoked at different times, based on the type of the actual object referred.

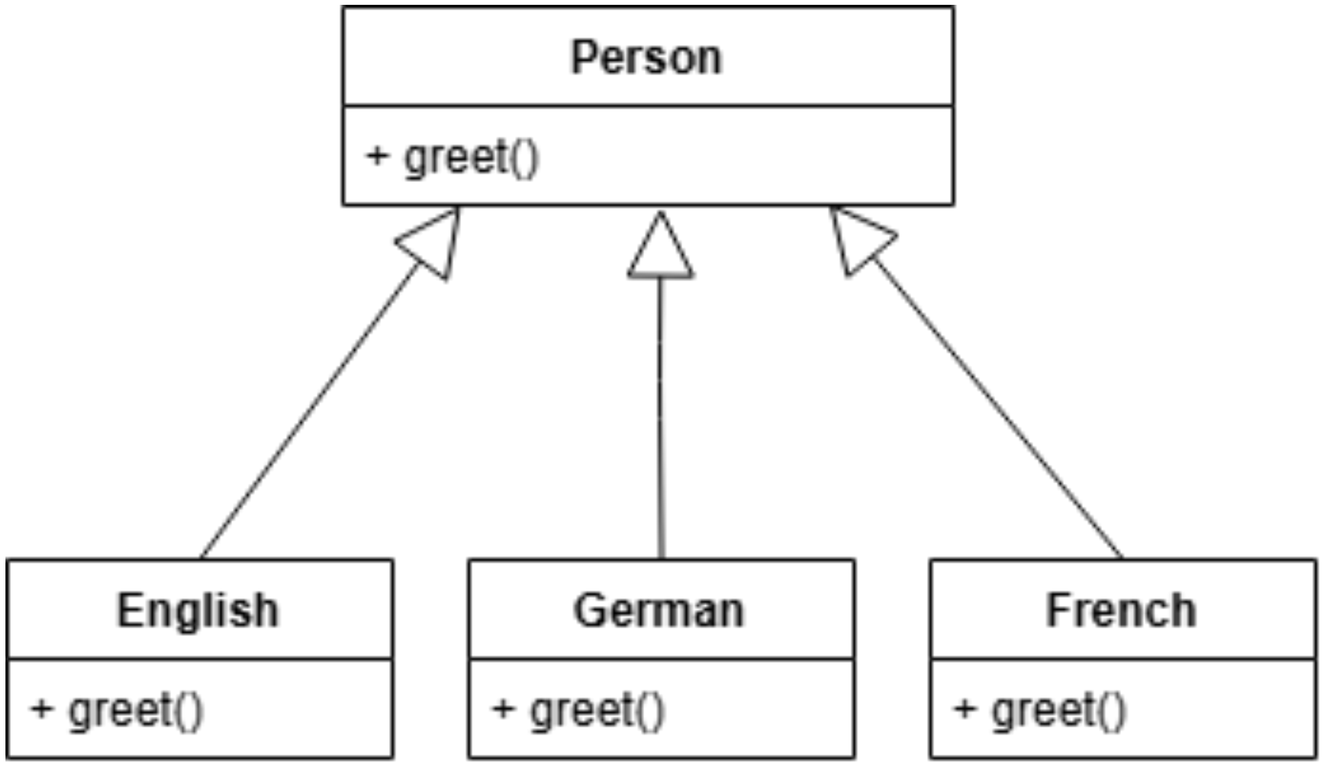
codingstudio

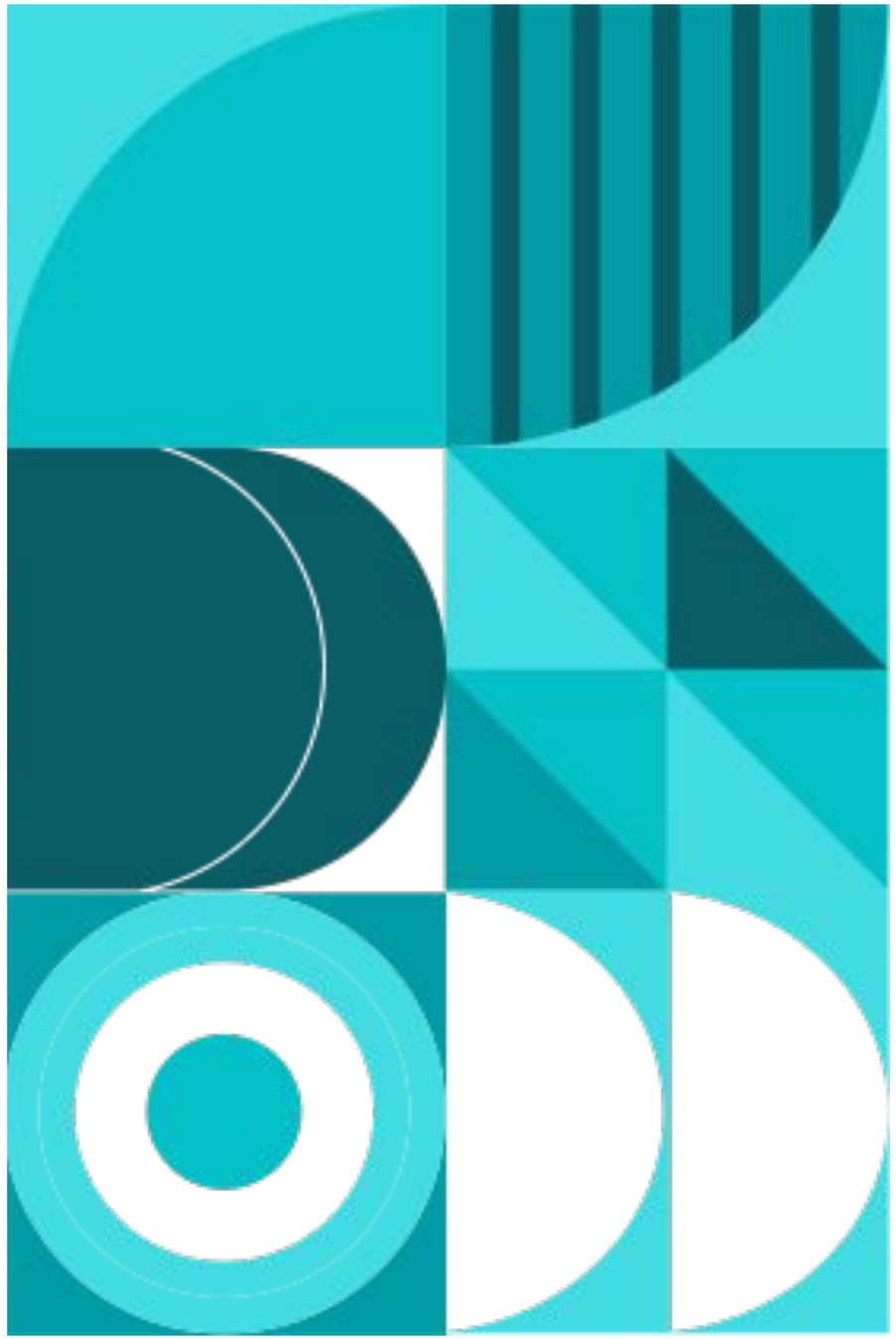
Inheritance



codingstudio

POLYMORPHISM





codingstudio

Constructor

- Constructor dipanggil menggunakan nama Class dan diikuti dengan pemanggilan fungsi dengan menambahkan ().

- Contoh:
  - k = Karyawan()

- Dengan menggunakan Constructor, kamu dapat menentukan untuk melakukan hal apa ketika sebuah Object dibuat.

codingstudio

Constructor

- Cara membuat Constructor:

```
def __init__(self, nama, NIK):
 self.nama = nama
 self.NIK = NIK
```

- Tanpa membuat Constructor, default Constructor seperti di atas akan dibuat secara otomatis.

codingstudio

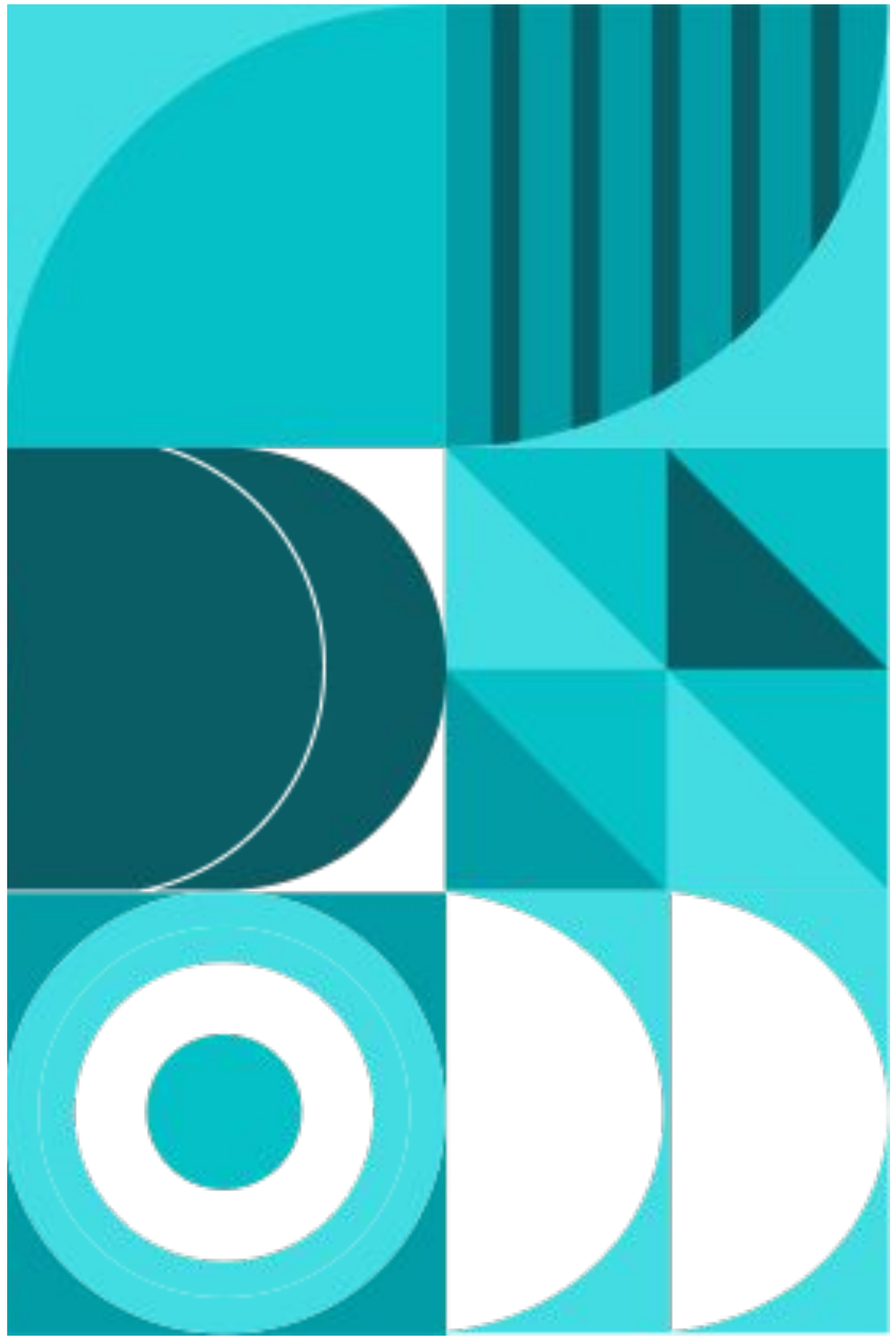
ATRIBUT CLASS

- **Kita dapat mengakses Atribut dari Class dengan menggunakan dot (.) contoh: siswa.nama**
- **Atribut adalah bagian dari Object, setiap Object dapat memiliki nilai Atribut yang berbeda-beda**

Misal:

print(siswa.nama)

Kode tersebut akan mencetak nama siswa tersebut.



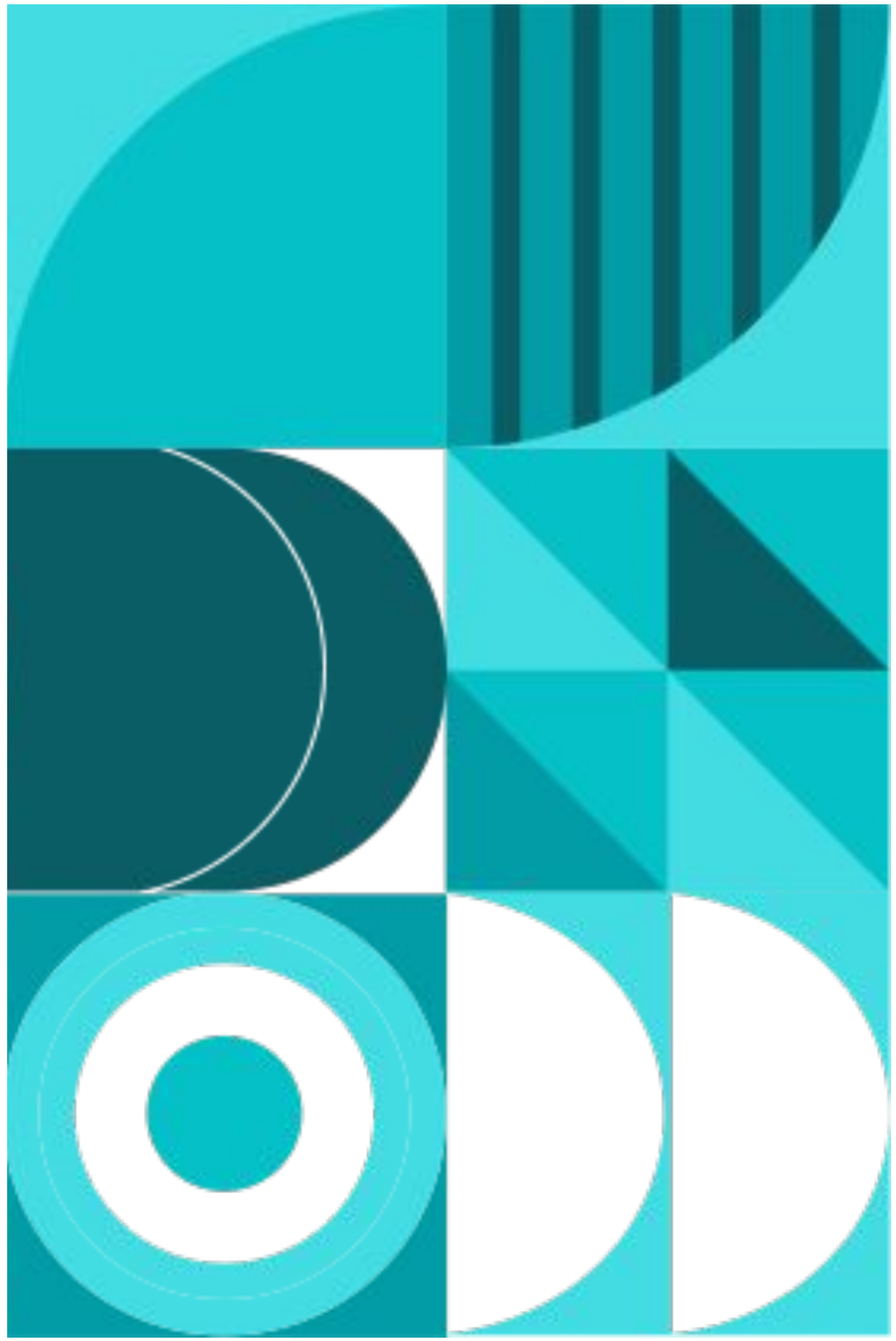
codingstudio

ATRIBUT CLASS

- Cara mengatur nilai dari Atribut:

siswa.nama = “Evan”

Kode tersebut akan mengatur Atribut nama pada Object siswa dengan nilai “Evan”



```
class Mobil:
    roda=4

    def __init__(self, merk=None, seri=None, warna = None):
        self.merk   = merk
        self.seri   = seri
        self.warna  = warna

    def __str__(self):
        return "merk: " + self.merk + " seri: " + self.seri + "warna: " +
        self.warna

m1= Mobil("Toyota", "Avanza", "hitam")
m2 = Mobil("Honda", "Jazz", "kuning")

#cetak class attribute
print(m1.roda)
print(m2.roda)

#cetak instance attribute
print(m1.merk)
print(m2.merk)
```

Class Attribute

Instance Attribute

codingstudio

CLASS METHOD

Method pada Class adalah Function yang terikat pada Class tersebut dan memiliki reference ke Object.

codingstudio

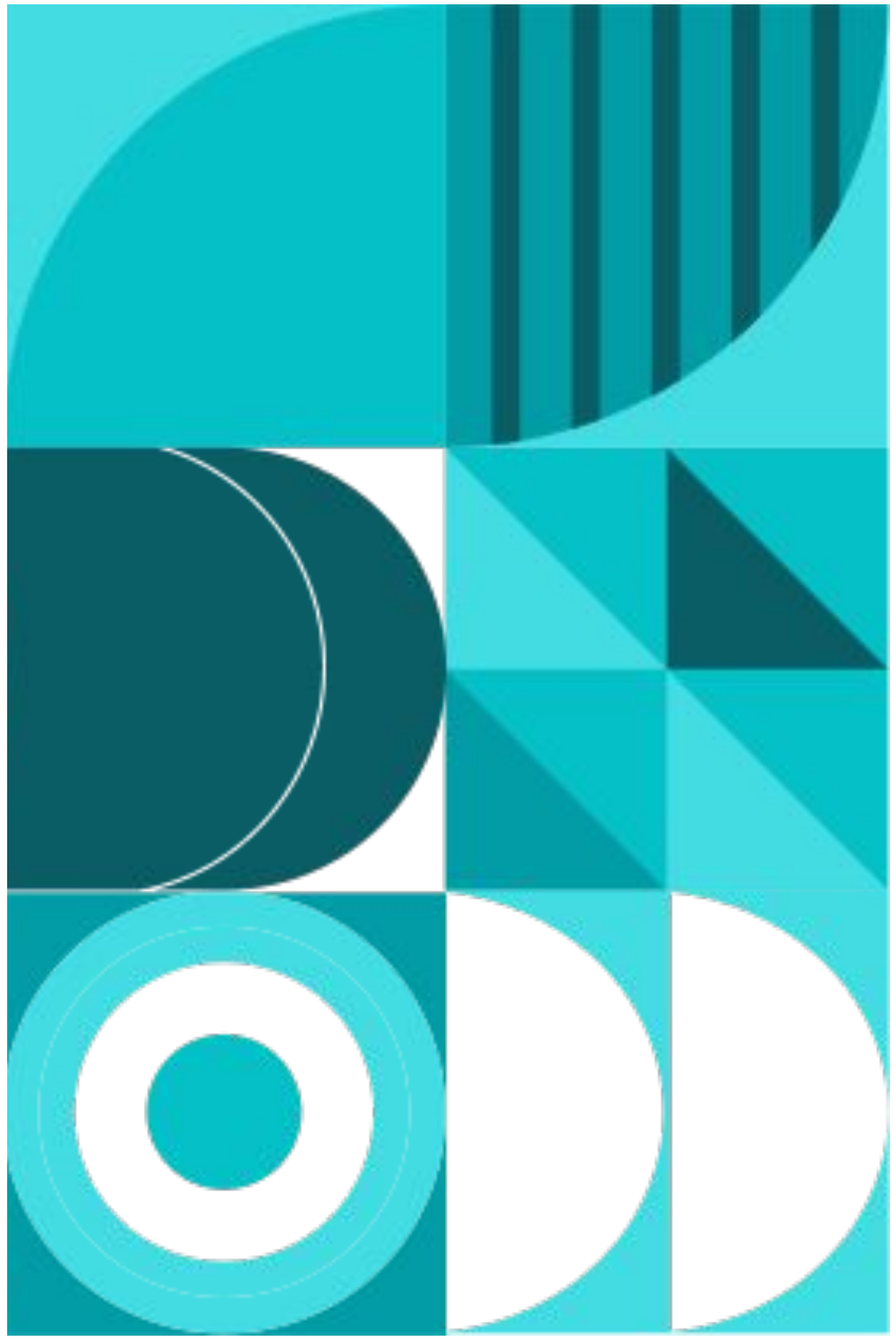
METHOD VS FUNCTION

function:

```
do_something(param1)
```

method:

```
an_object.do_something(param1)
```



codingstudio

FILE PROCESSING

codingstudio

File Processing

- File Processing

File processing adalah kemampuan untuk membaca, menulis, dan mengelola file di dalam program Python. Python menyediakan berbagai fungsi dan metode untuk melakukan operasi file, seperti membuka file, membaca isi file, menulis ke file, dan menutup file.

- Membuka File

Untuk membuka file, kita dapat menggunakan fungsi open(). Fungsi ini memiliki dua parameter utama, yaitu nama file dan mode file.

FILE PROCESSING - READ

codingstudio

Mode file yang umum digunakan adalah:

- "r": Membuka file dalam mode baca (default)
- "w": Membuka file dalam mode tulis (menghapus isi file jika sudah ada)
- "a": Membuka file dalam mode tambah (menambahkan data ke akhir file)
- "r+": Membuka file dalam mode baca-tulis

Membaca Isi File

Setelah file dibuka, kita dapat membaca isinya menggunakan beberapa metode, seperti:

- read(): Membaca seluruh isi file
- readline(): Membaca satu baris dari file
- readlines(): Membaca semua baris dari file dan mengembalikannya sebagai daftar

codingstudio

```
file = open("filename.txt", "r")
content = file.read()
print(content)
file.close()
```

FILE PROCESSING - WRITE

codingstudio

- Untuk menulis ke file, kita dapat menggunakan metode write(). Jika file tidak ada, maka file akan dibuat.
- Setelah selesai menggunakan file, sebaiknya file ditutup menggunakan metode close(). Ini untuk membebaskan sumber daya yang digunakan oleh file.

```
file = open("filename.txt", "w")
file.write("Ini adalah teks yang ditulis ke file.")
file.close()
```

THANK YOU

codingstudio