

# Object Oriented Programming

## Data Attributes

- Data attributes:
  - Private data
    - ◆ Starts with "\_\_"
    - ◆ E.g. `self.__name = name`
  - Prevents external functions from accessing
  - Only accessed by class methods
- Data hiding
  - Achieved by declaring data members to be private
  - Can only be accessed through public methods
- Class Methods:
  - Public functions
  - Includes parameter `self`
  - Accesses class data attributes
- Encapsulation
  - Combining methods and attributes as a single object type
- Inheritance
  - Ability to create new classes (subclasses) which possess ALL data attributes and methods of the existing parent class (superclass)
- Polymorphism
  - Ability of different classes to respond to same methods in different ways
  - Methods have the same name
  - Subclass method overrides superclass method
- Method overloading
  - Methods / functions of the same name
  - Different number of parameters / different types of parameters
- Reusability
  - Software components can be used in many different applications without having to modify code in the component
  - Reliability
    - ◆ Already tested and debugged
    - ◆ Developed by specialists
  - Saves time in software development
  - Decreases maintenance

- Advantages:
  - Ability to hide data
  - Reusability
  - Methods can be changed without affecting how they are used

## Class Methods:

- Initialising class:

```
class ABC:
    def __init__(self, data1, data2):
        self.data1 = data1
        self.data2 = data2

    def display(self):
        print(self.data1)
        print(self.data2)
```

- Subclass inheritance & polymorphism:

```
class DEF(ABC):
    def __init__(self, data1, data2, data3):

        # initialise superclass
        super().__init__(data1, data2)

        self.data3 = data3

    # polymorphism (same name as function in
    superclass)
    # overrides function superclass
    def display(self):
        print(self.data3)

    # output: data3
```

## Inheritance Diagram:

