9. OOPS & Exception Handling

Table of Contents

|  |  |  |
| --- | --- | --- |
| Sl. No | Topics | Page No |
| 9.1 | **Understand Object Oriented Concepts** | **2** |
| 9.2 | **Usage of Regular Expression** |  |
| 9.3 | **Understand Exception Handling** |  |
| 9.4 | **Creating User Defined Exception** |  |

## 9.1 Understand Object Oriented Concepts

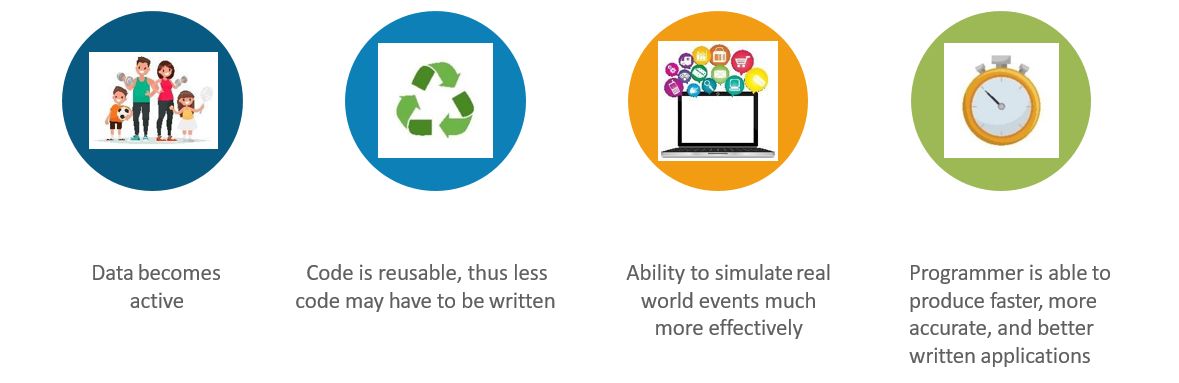
**What is the** **difference between Procedure Oriented Programming (POP) & Object Oriented Programming (OOP)?**

|  |  |
| --- | --- |
| **Procedure Oriented Programming** | **Object Oriented Programming** |
| In POP, program is divided into small parts called functions | In OOP, program is divided into parts called objects |
| POP follows Top Down approach | OOP follows Bottom Up approach |
| In POP, Data can move freely from function to function in the system | In OOP, objects can move and communicate with each other through member functions |
| Examples of POP: C, VB, FORTRAN and Pascal | Examples of OOP: C++, JAVA and Python |

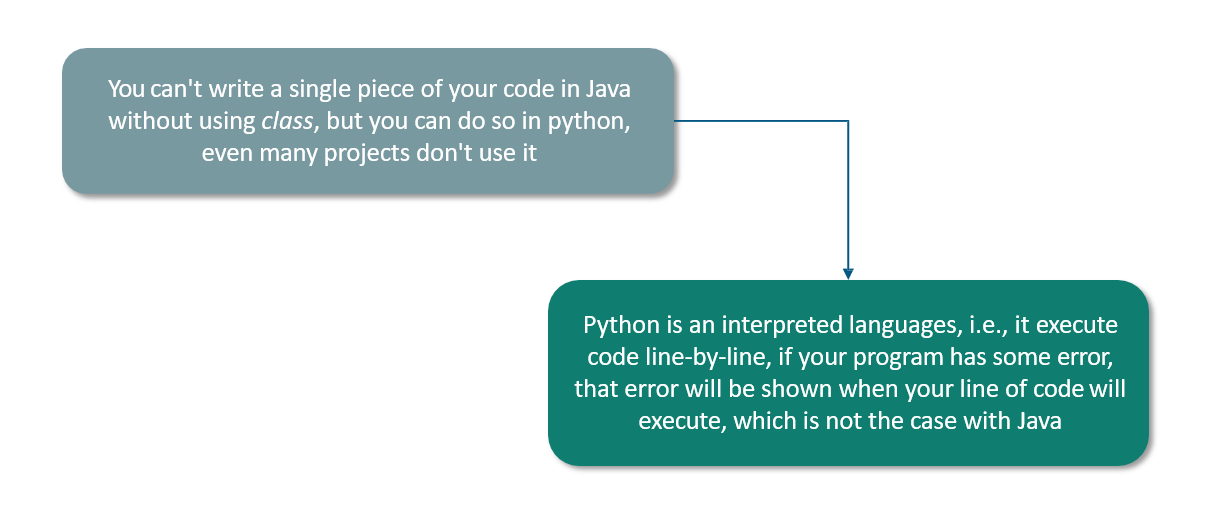
**Object Oriented Concepts**

**What are the Object Oriented Concepts?**

Python is an Object Oriented language, because of this creating and using classes and objects becomes easy.



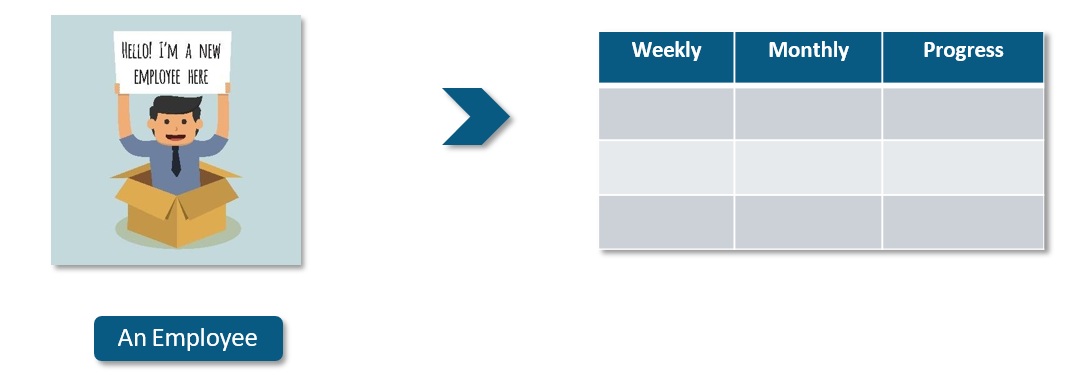
**Python OOP is different than other OOPS**



**Give a Use Case for Classes & Objects?**

**Use Case**

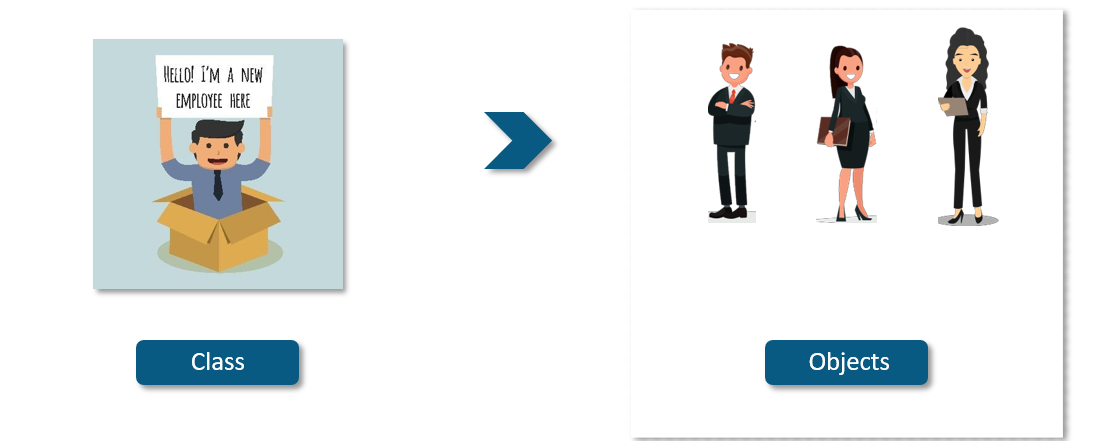
**Sri Soft** is a Multi-National Company. This Company wants to create employee information sheet, which will include name, employee ID, and their progress



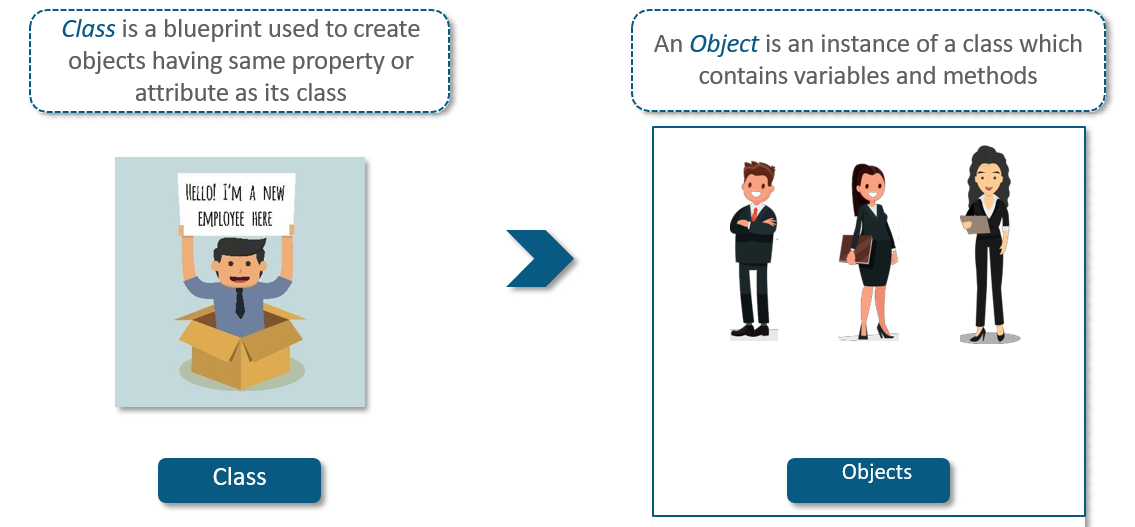
**Solution: Use Case**

However, it will be difficult to create a separate sheet for every employee. So they decided to create one class

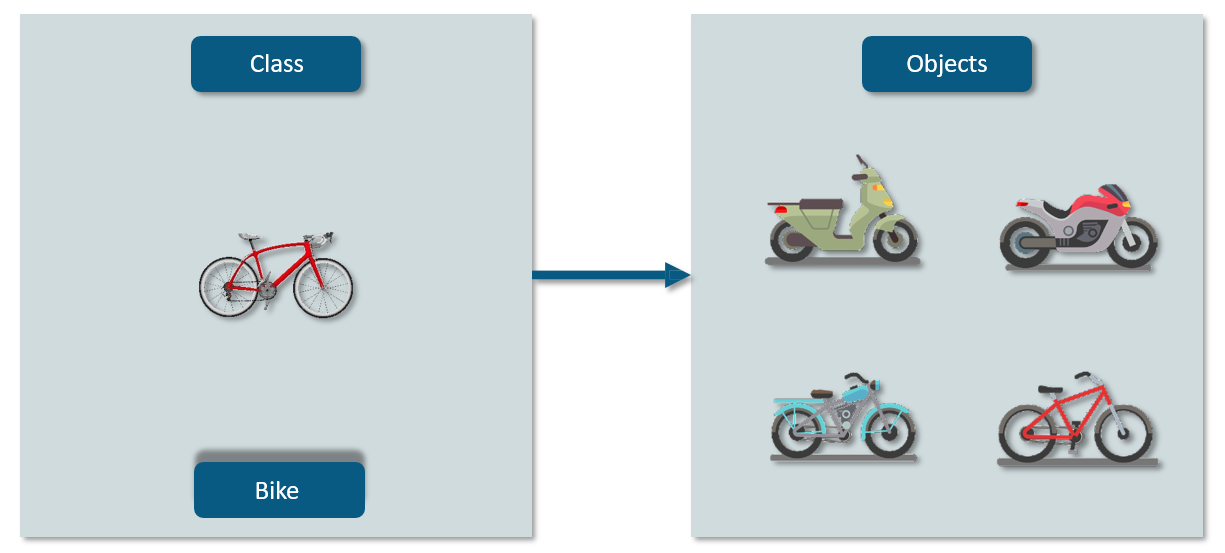
of Employee’s Information and create object of every employee and call class for new employee.



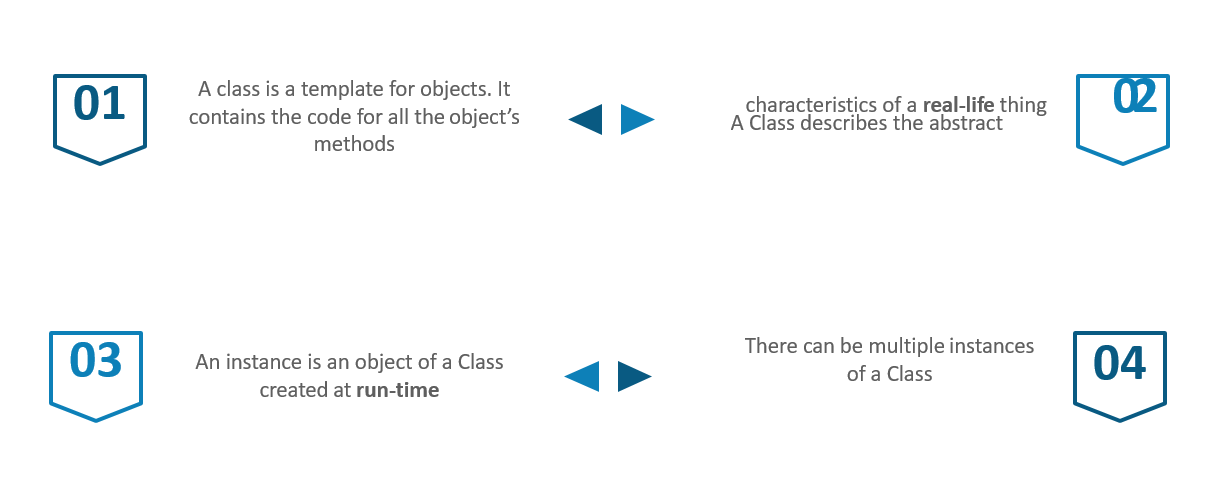
**What are Class and Objects? Give an example?**



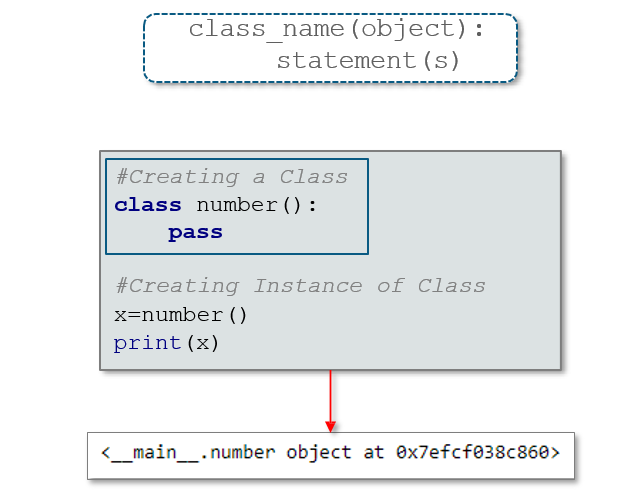
**Example:** **Class and Objects**



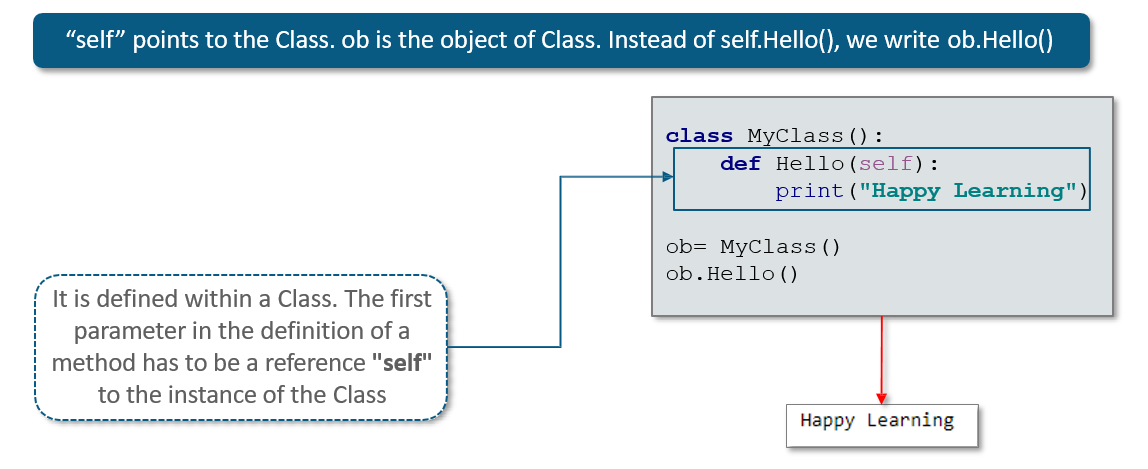
**What is the Relation between Classes and Objects?**



**How will you create a Class?**



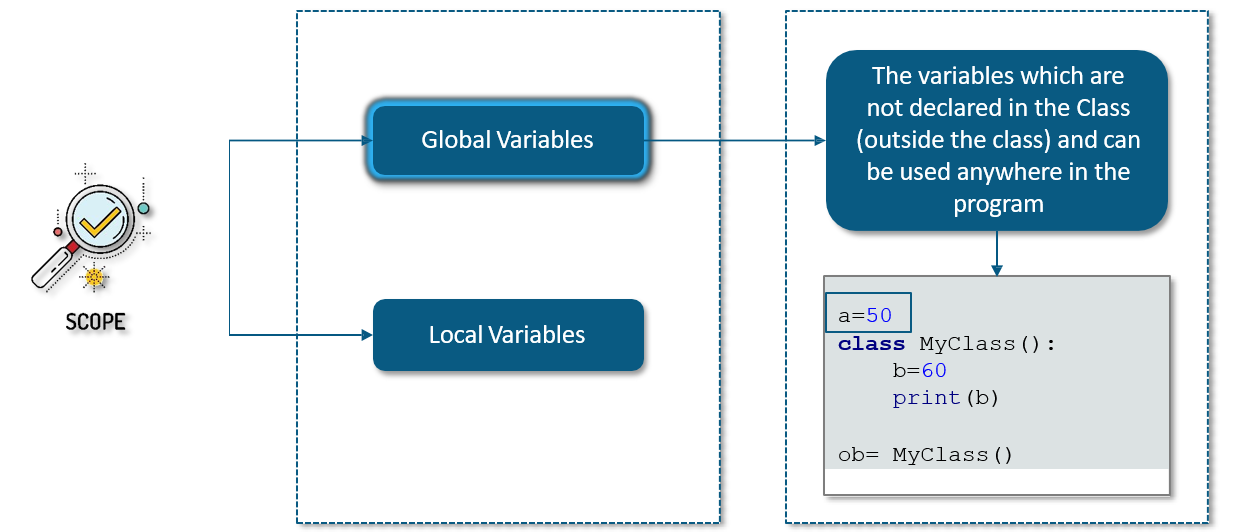
**How will you define a Method?**



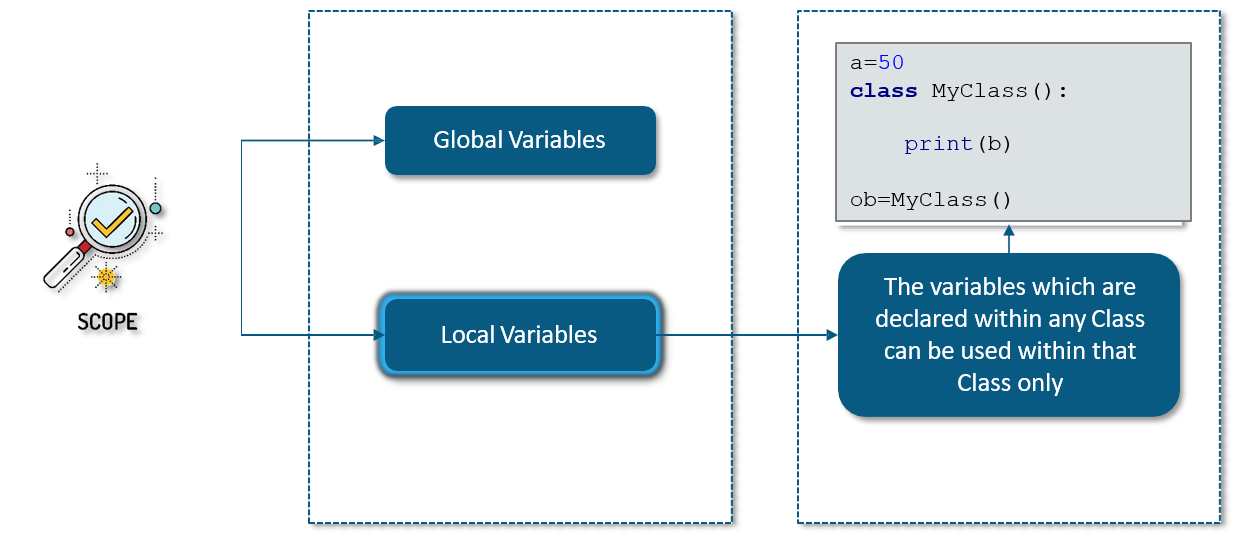
**What are the Scope of Variables in Python?**

The scope of variable should be either Global or Local.

**Global Variable**

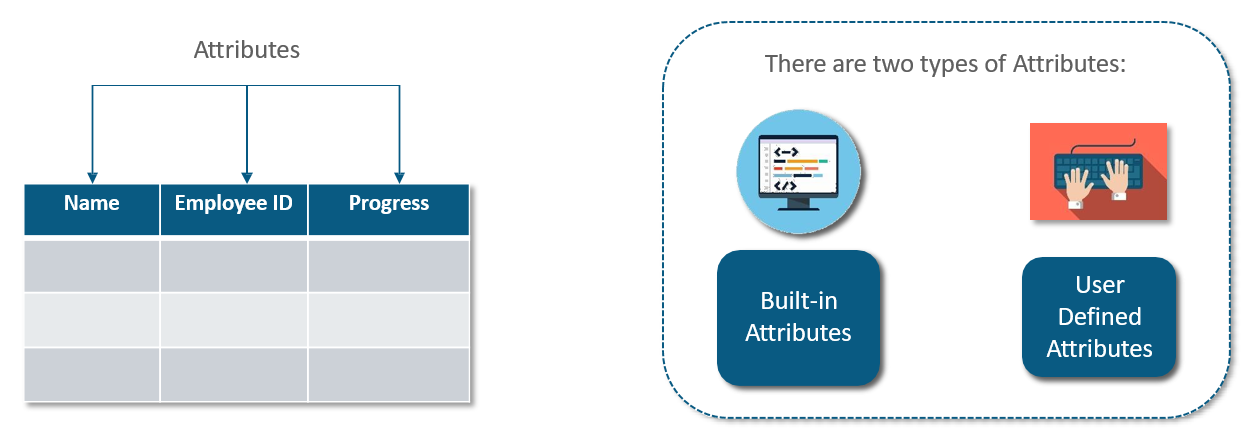
****

**Local Variable**



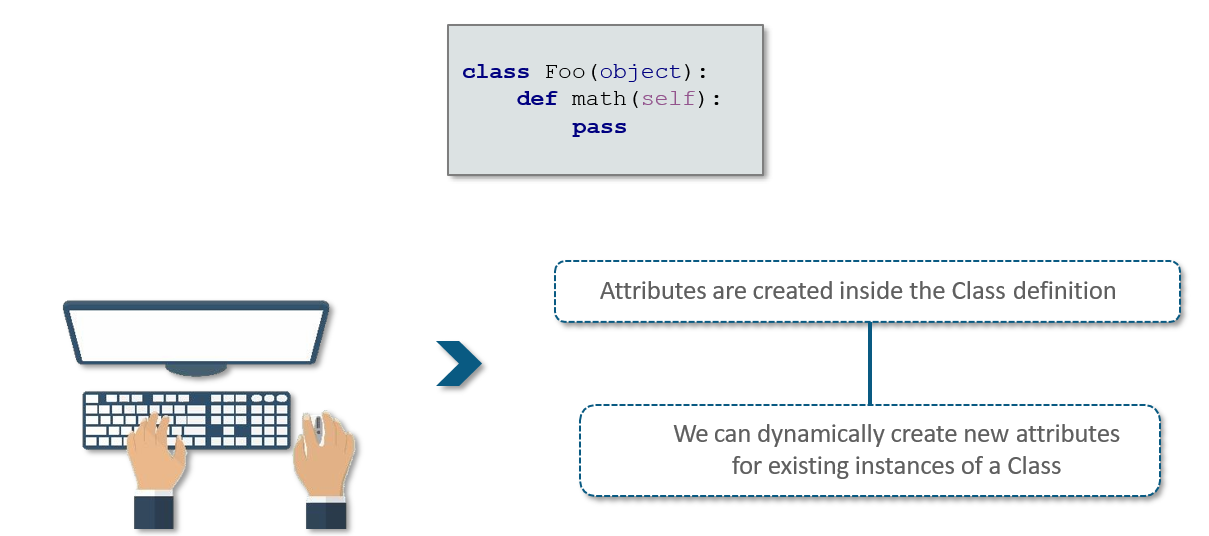
**What is Attributes?**

Class Attributes are attributes which are owned by the class itself. They will be shared by all the instances of the class.

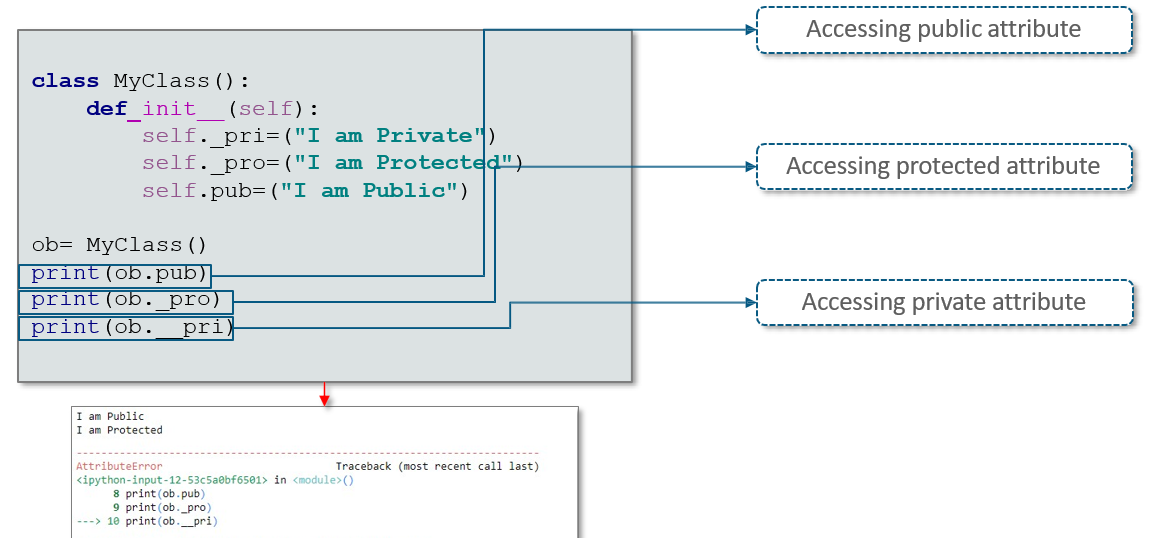
****

**What are the Built-in Class Attributes?**

**What are the attributes defined by users?**



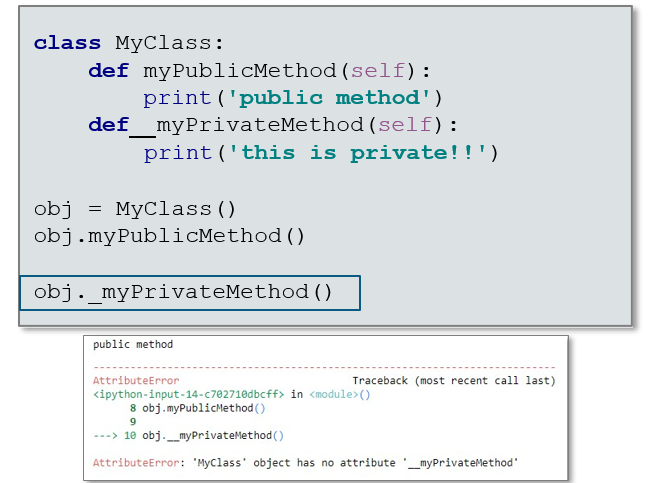
**Example: Public, Protected and Private Attributes**

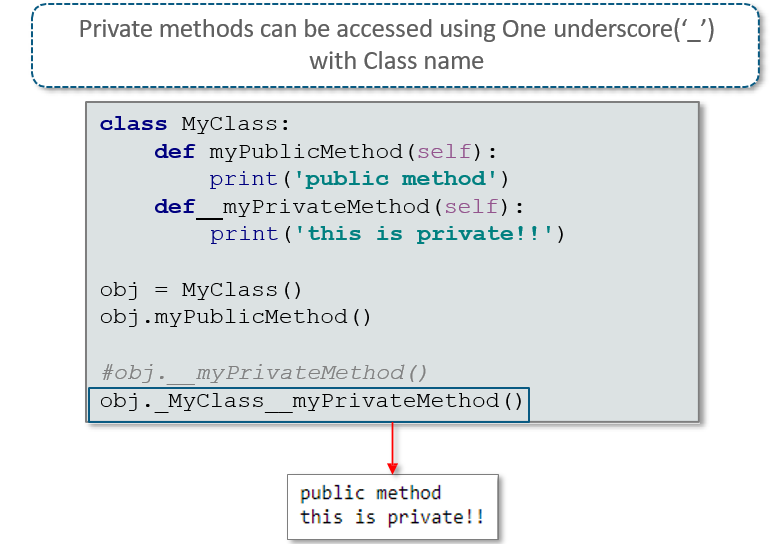


**What are Private Methods**?

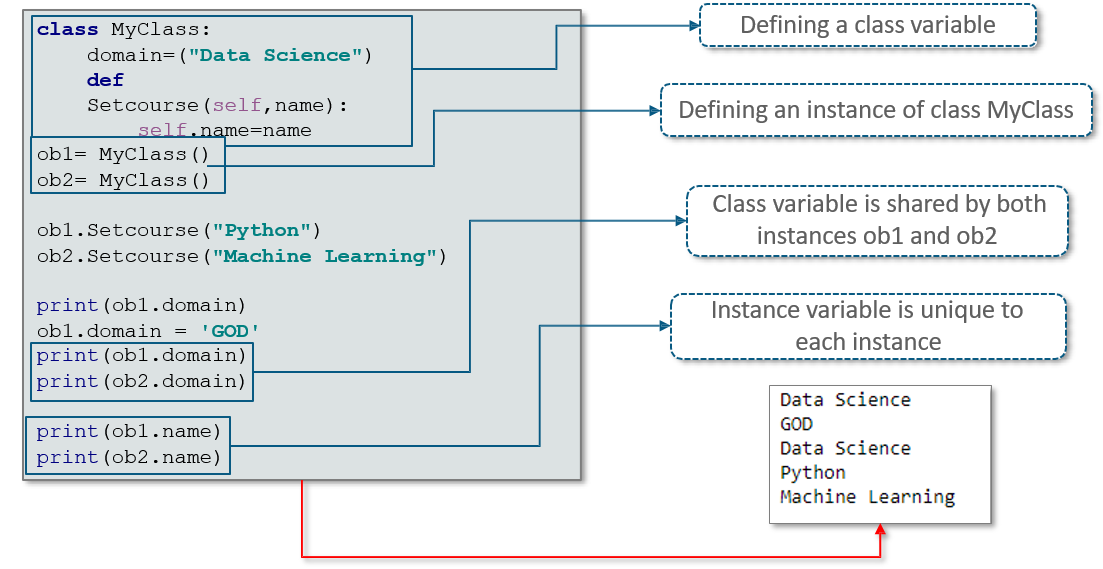
When the attributes of an object can only be accessed inside the Class, it is called a **Private Class**.

Python use **two underscores** to hide a Method. Two underscores can also be used to hide a Variable.

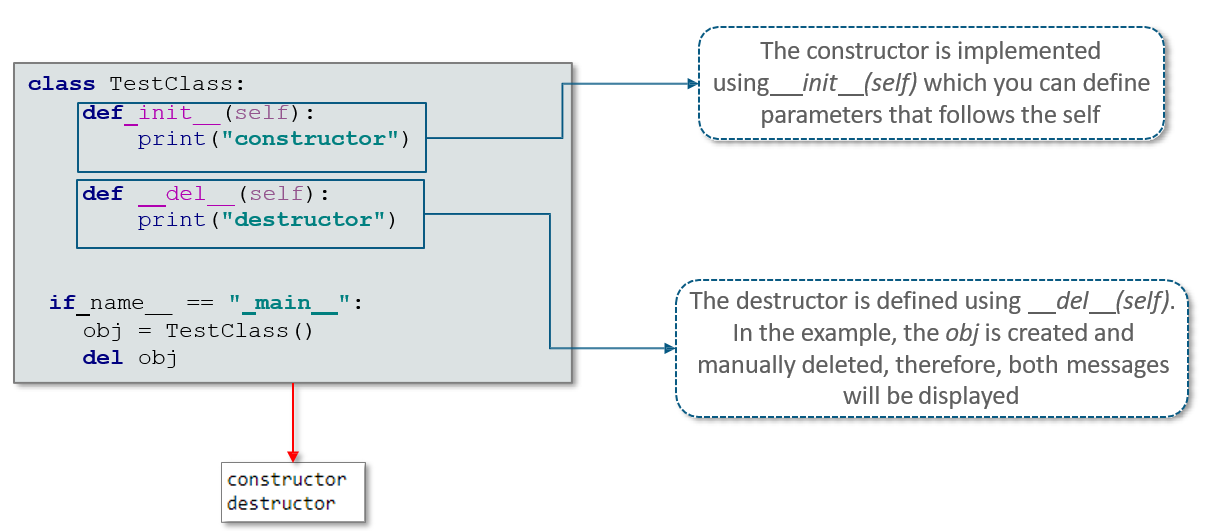




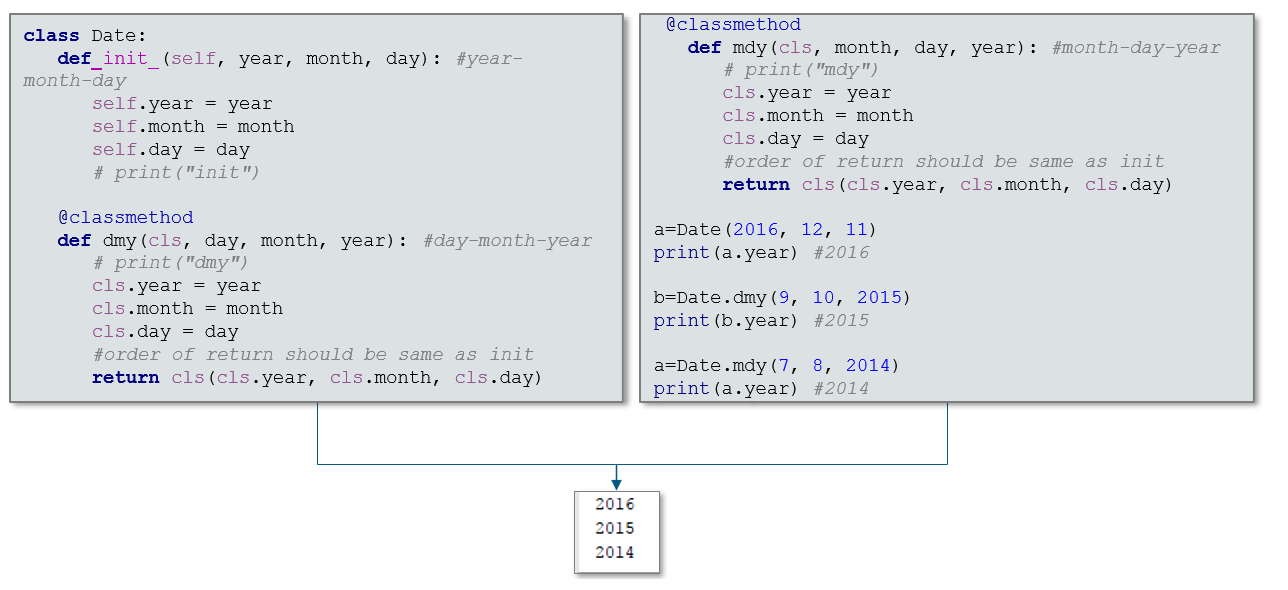
**What is the difference between Class Variable and Instance Variable**?



**What is Constructor and Destructor?**

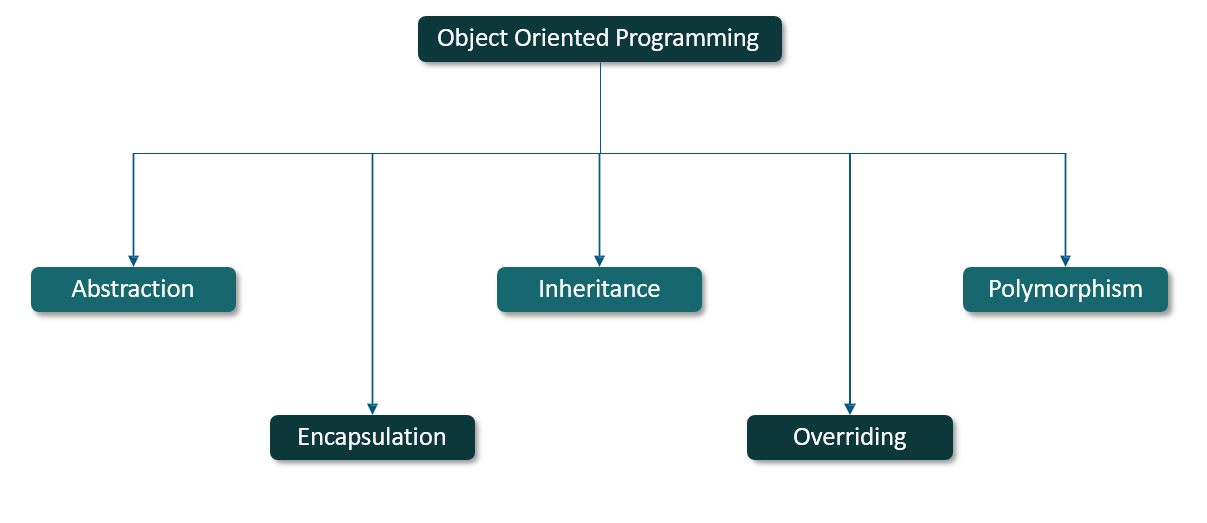


**Can we define Multiple Constructors inside a class?**



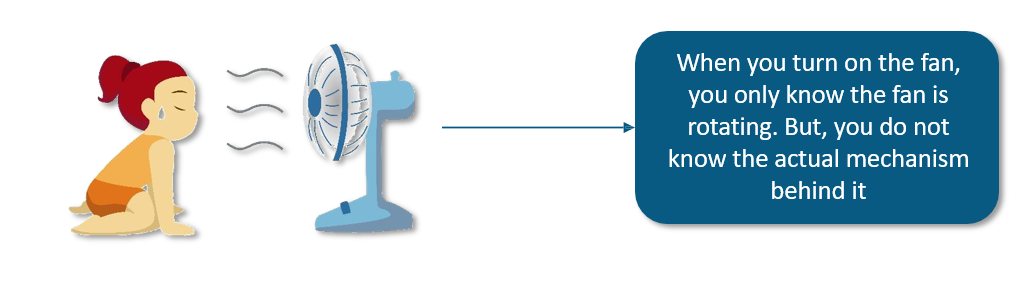
**Object Oriented Programming**

**What are the Key Concepts - Object Oriented Programming?**



**What is Abstraction?**

*Abstraction* is simplifying complex reality by modelling classes appropriate to the problem Class abstraction means to separate class implementation from the use of the class.



**What is Encapsulation?**

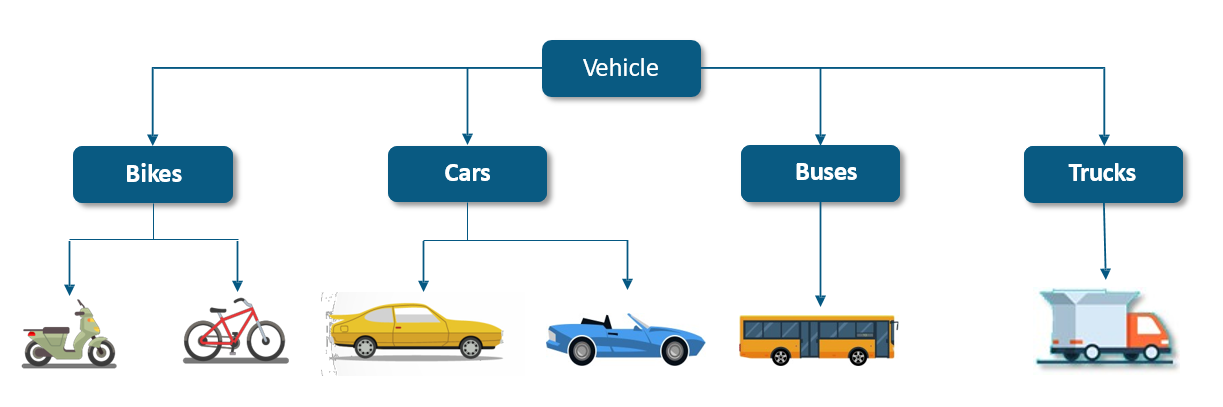
**Encapsulation:** Combining the code into a public **interface**, and a private **implementation** of that interface

It is a Mechanism for restricting the access to some of an objects components, which means that the internal representation of an object cannot be seen from outside of the objects definition.

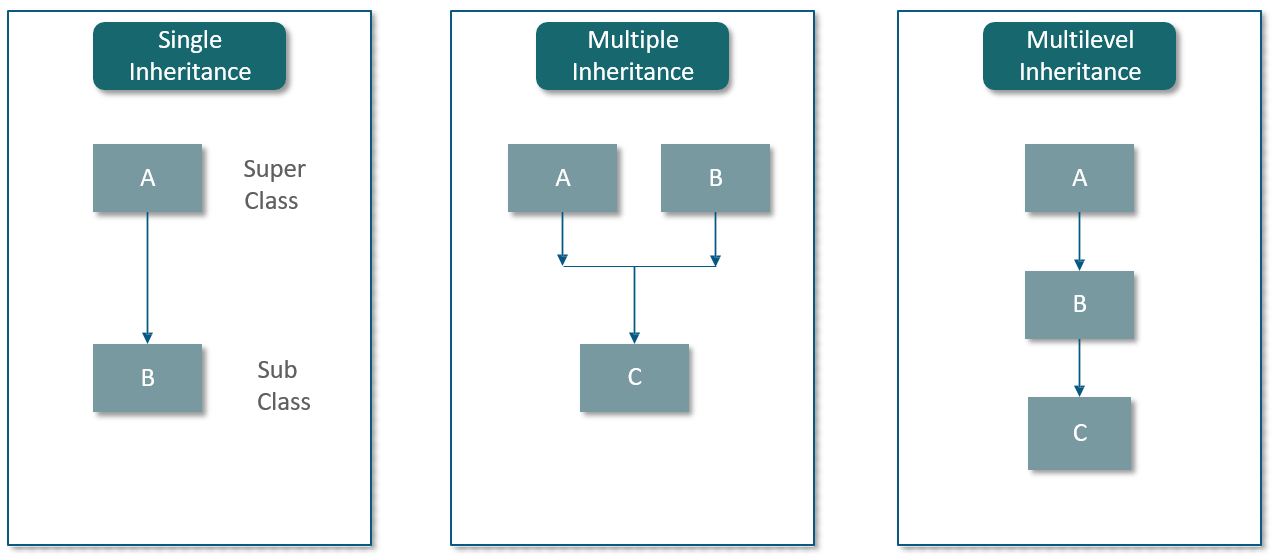
**What is Inheritance?**

Inheritance is the powerful feature of Object Oriented Programming.

It refers to deriving a class from the base class with little or no modification in it.

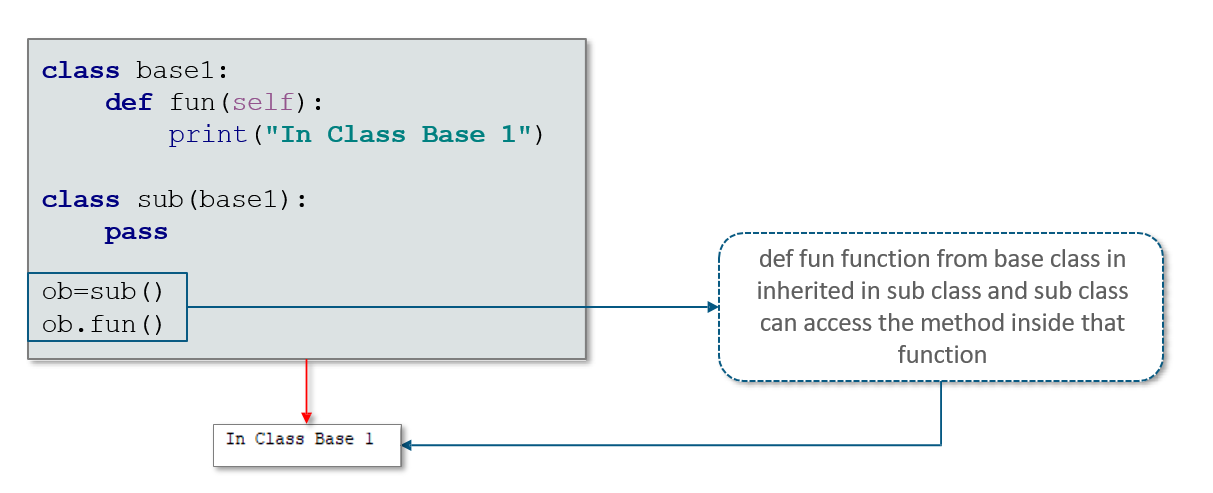


**What are the types of Inheritance?**

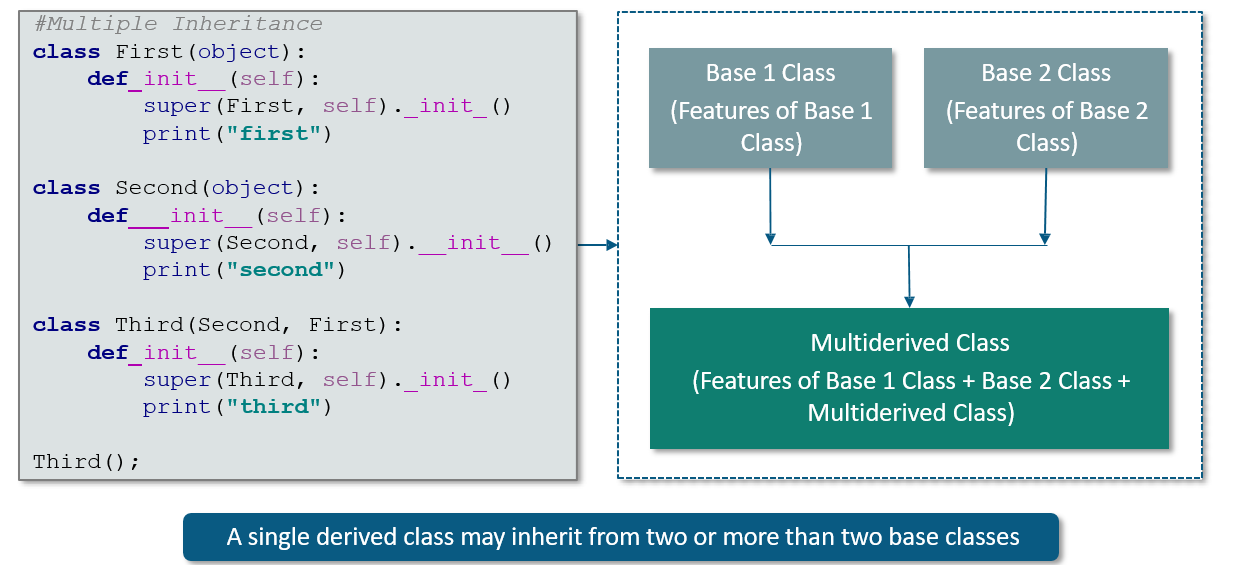


**Single Inheritance**

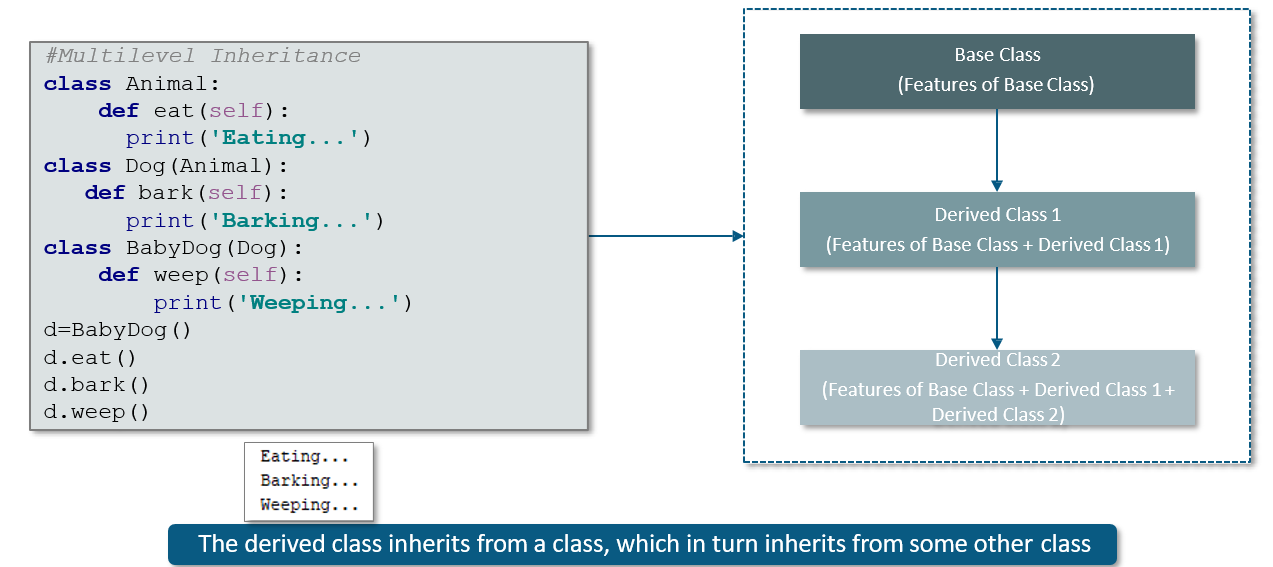
Important benefits of inheritance are code reuse and reduction in the complexity of a program.



**Multiple Inheritance**

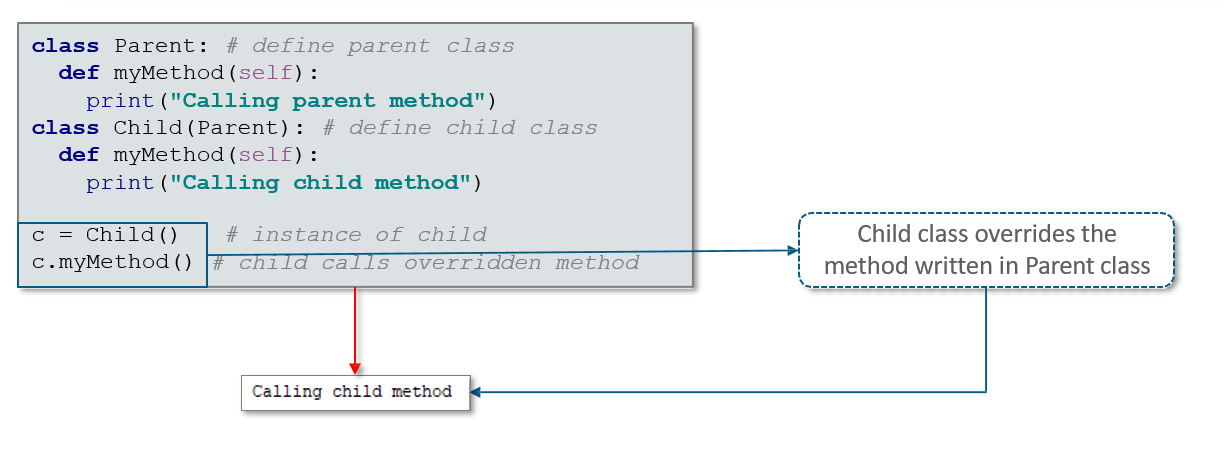


**Multilevel Inheritance**

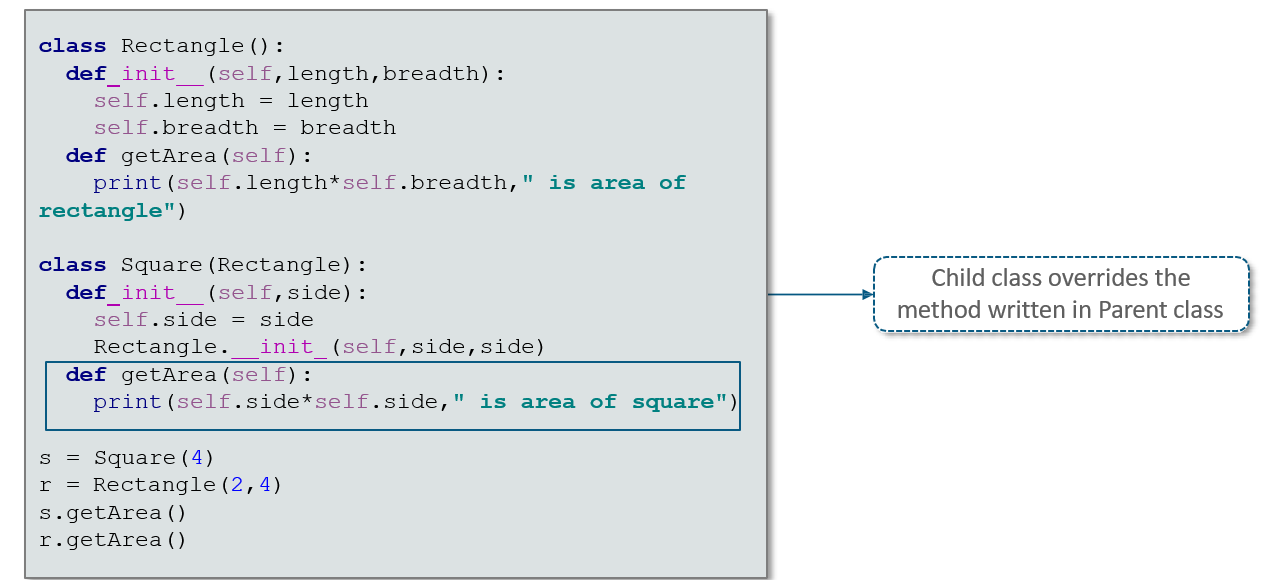


**How will you override a method?**

Parent class methods can always be overrided. One reason for overriding parent's methods is because you may want special or different functionality in your subclass.

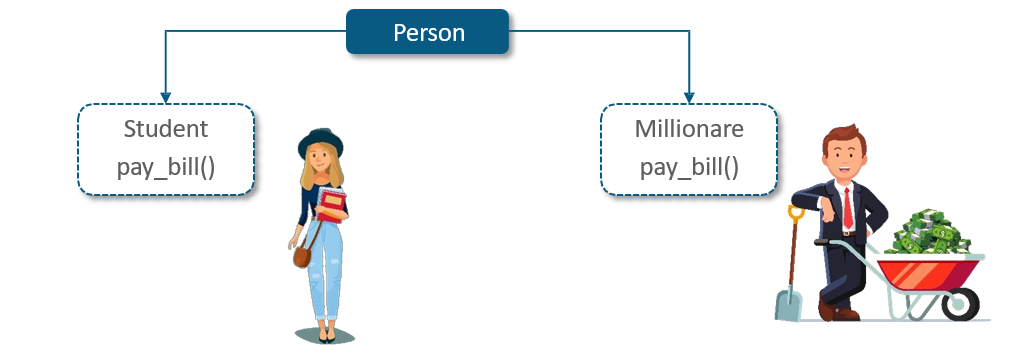


**Example: Class Overriding**

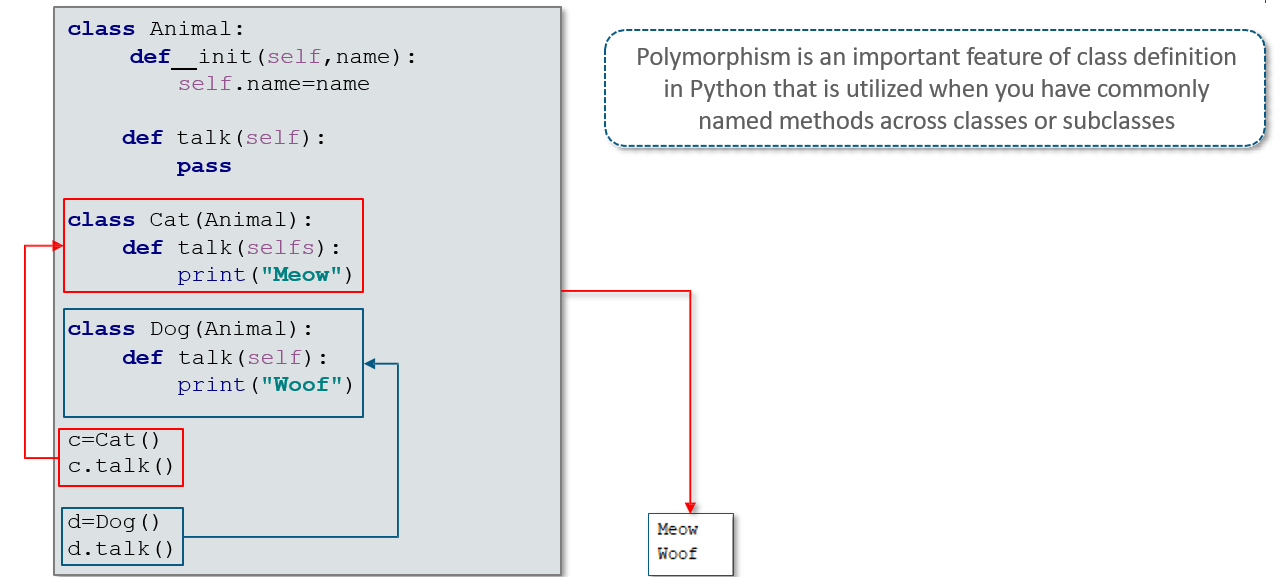


**What is Polymorphism?**

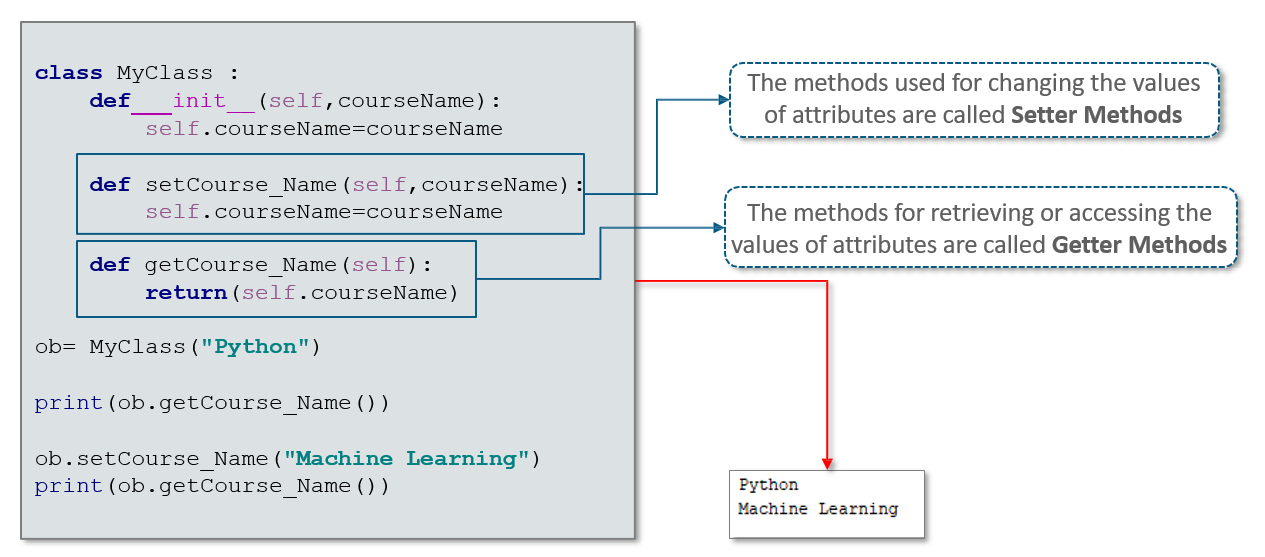
**Polymorphism** is the ability to leverage the same interface for different underlying forms such as data types or classes.



**Example: Polymorphism**



**What are Getter and Setter Methods?**

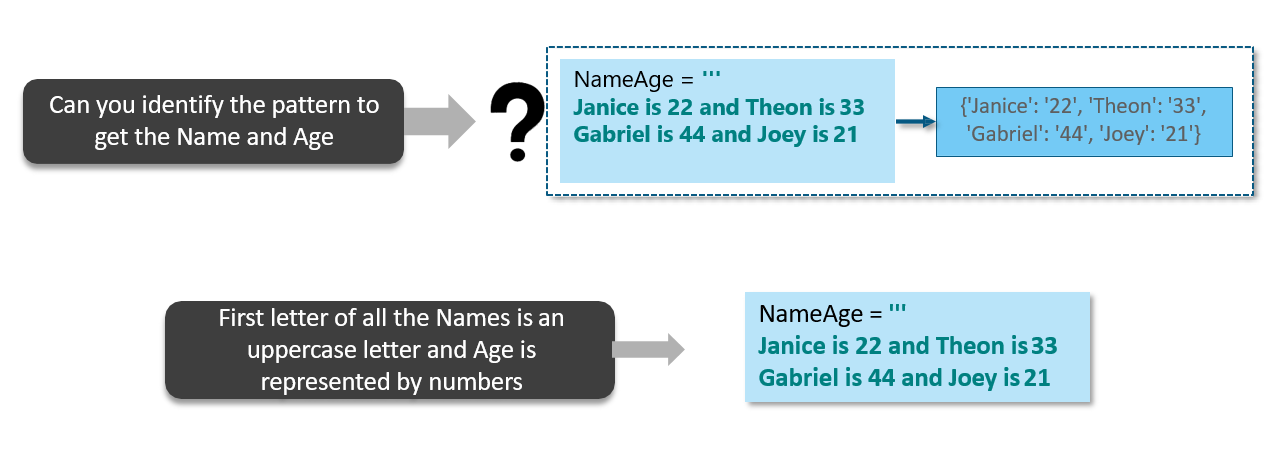


## 9.2 Usage of Regular Expression

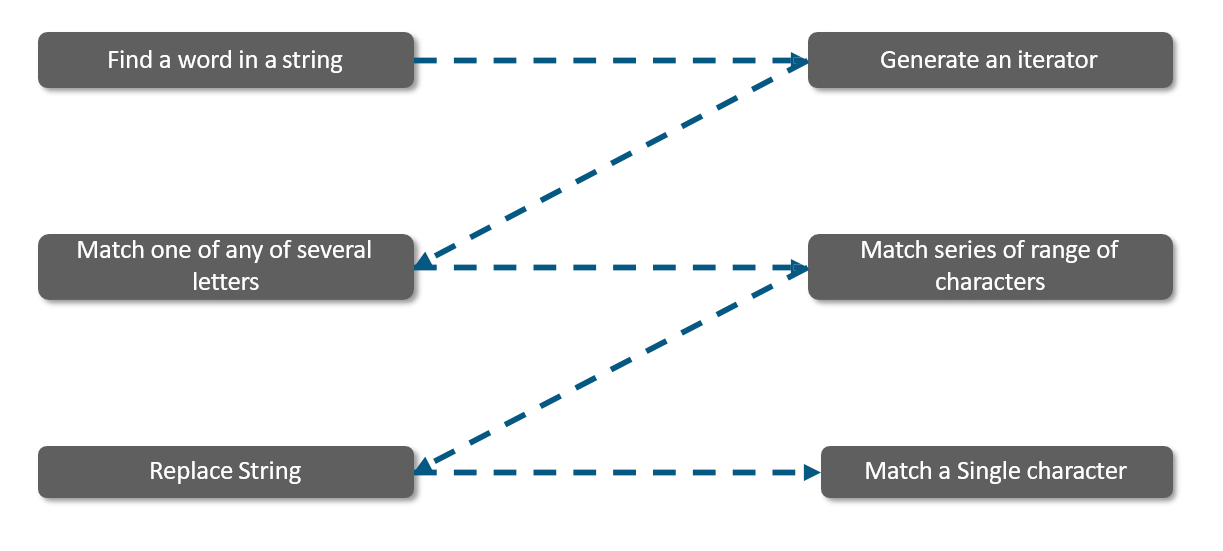
**What are Regular Expressions?**

A Regular Expression is a special text string for describing a search pattern.

Regular Expression is a seed for Natural Language Processing (NLP).

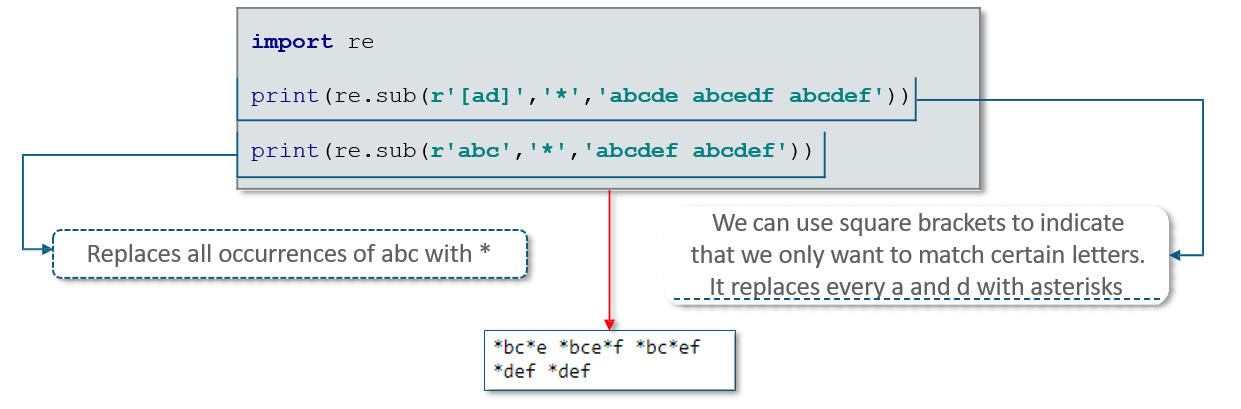


**What are the Operations** **of Regular Expression?**

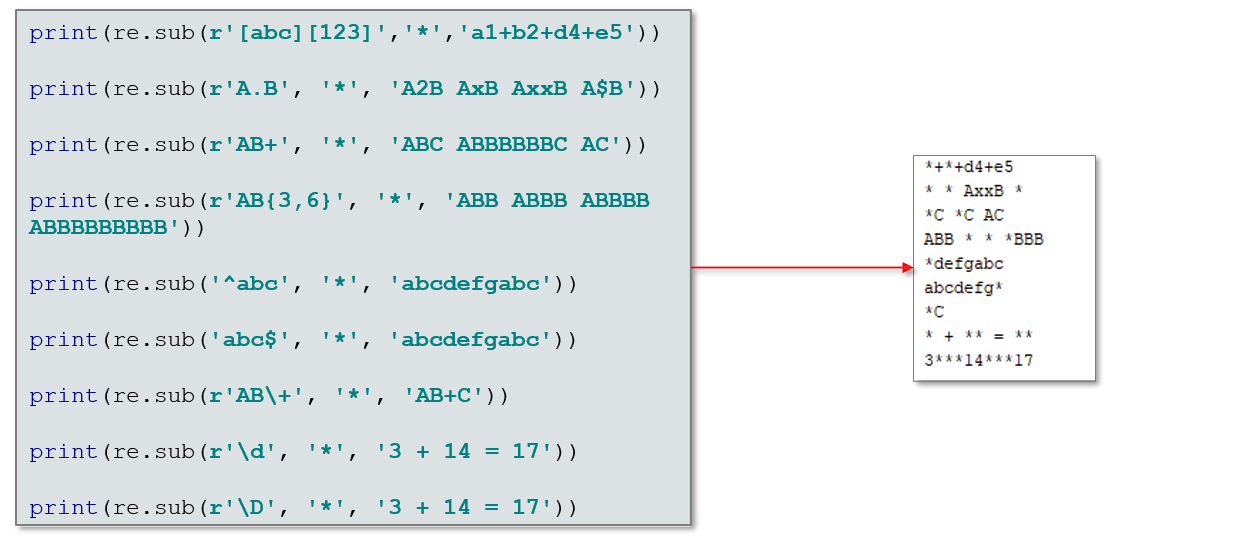


**Replace Methods**

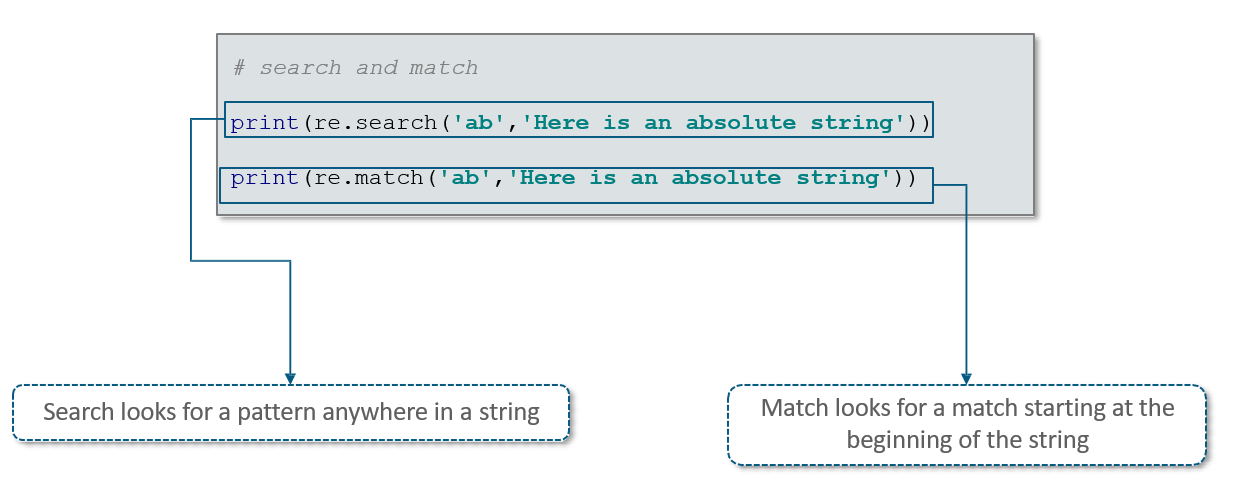
The replace method of strings is used to replace all occurrences of one string with another, and the index method is used to find the first occurrence of a substring in a string. But sometimes you need to do a more a sophisticated search or replace then Regular Expression is used.



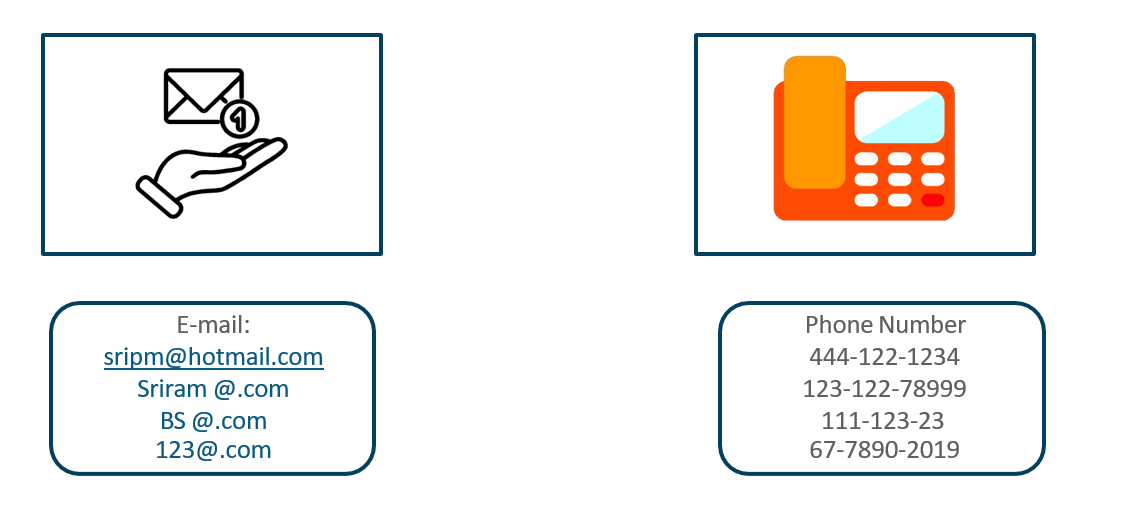
**Example**: **Regular Expression**



**Search and Match Function**

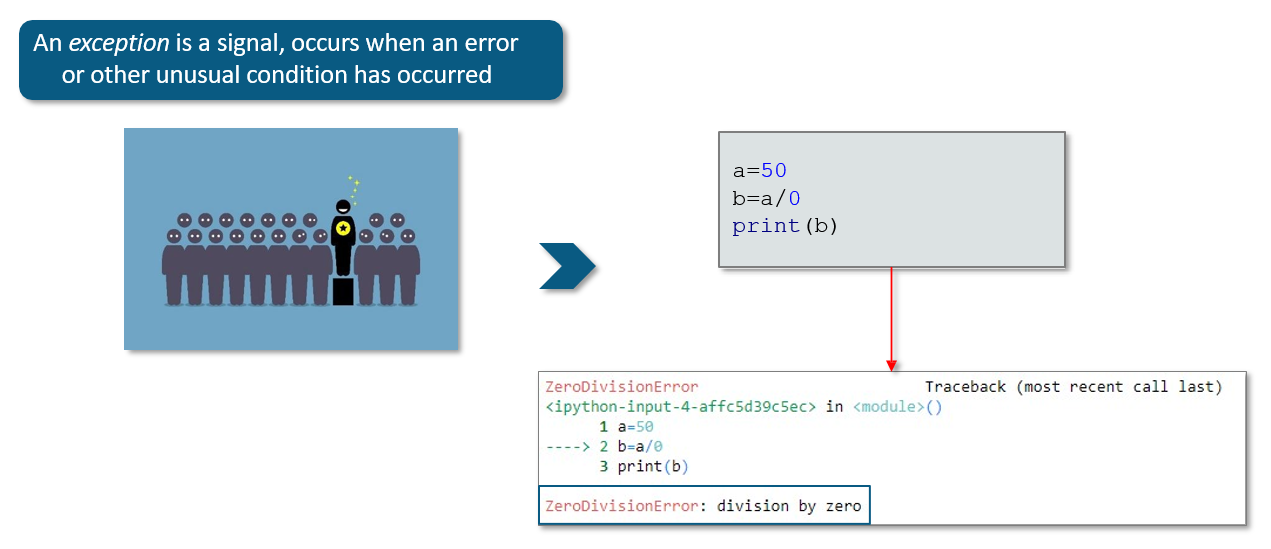


**Regular Expression Application**



## 9.3 Understand Exception Handling

**What is an Exception Handling?**

****

**How will you handle an Exception?**

Exceptions can be handled using a try statement. A critical operation, which can raise exception is placed inside the try clause and the code that handles exception is written in except clause.

try:

You do your operations here;

......................

except *ExceptionI*:

If there is ExceptionI, then execute this block. except *ExceptionII*:

If there is ExceptionII, then execute this block.

......................

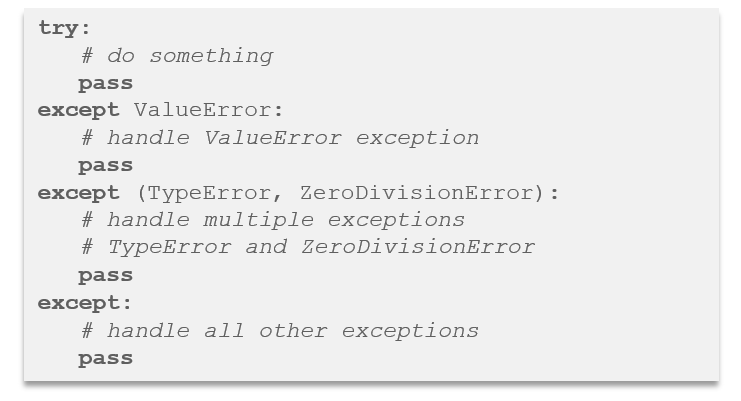
else:

If there is no exception then execute this block.

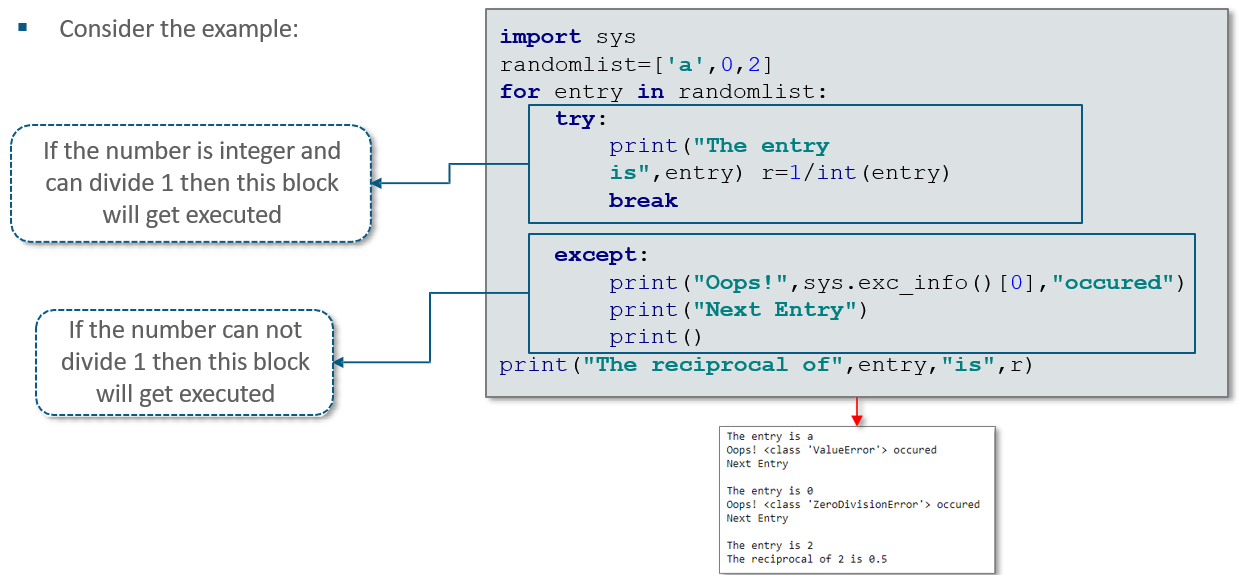
**How will you catch the specific exceptions**?

A try clause can have any number of except clause to handle them differently, but only one will be executed

in case an exception occurs. We can use a tuple of values to specify multiple exceptions in an except clause.

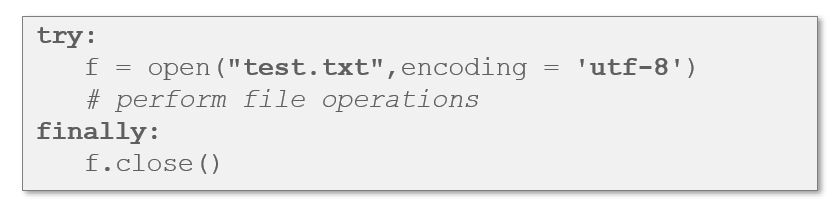


**Example: Handling an Exception**

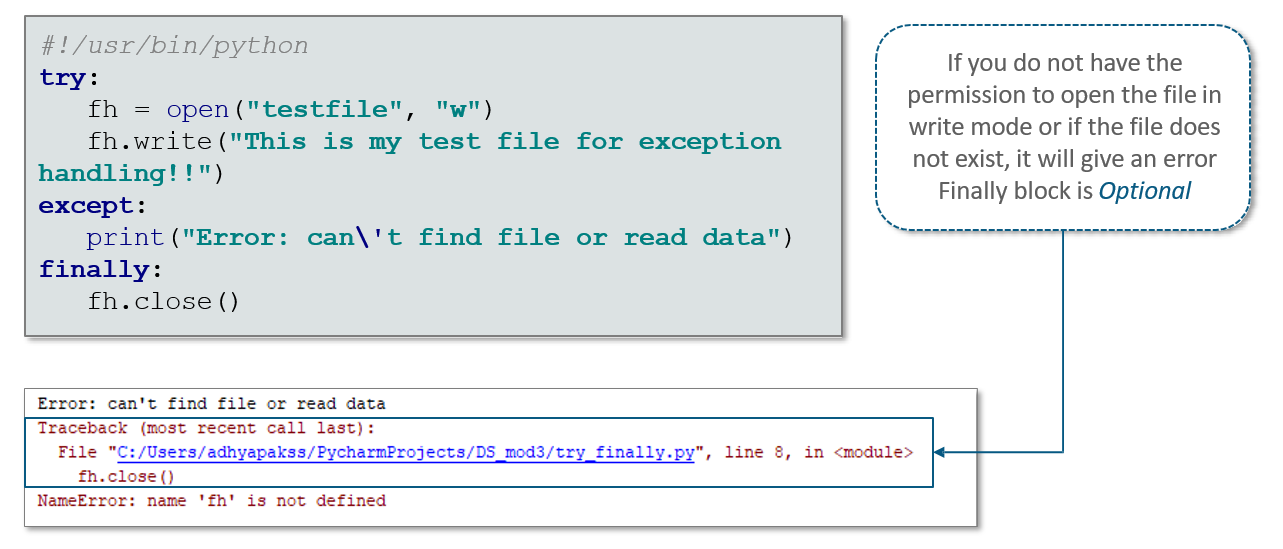


**Finally Block**

The *try* statement in Python can have an optional *finally* clause. This clause is executed no matter what, and is generally used to release external resources

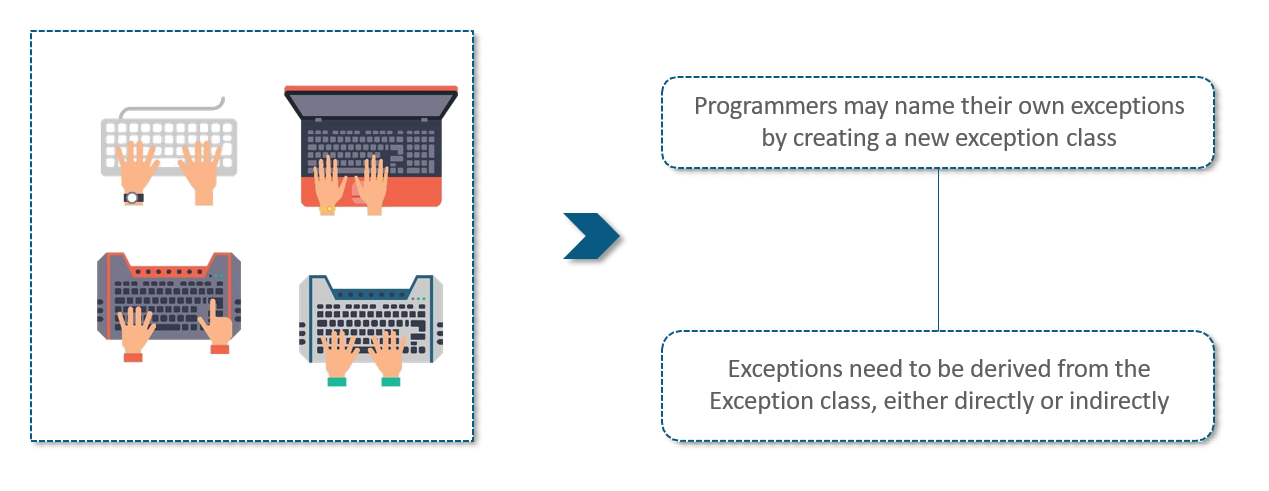


**Example: Try-finally block**

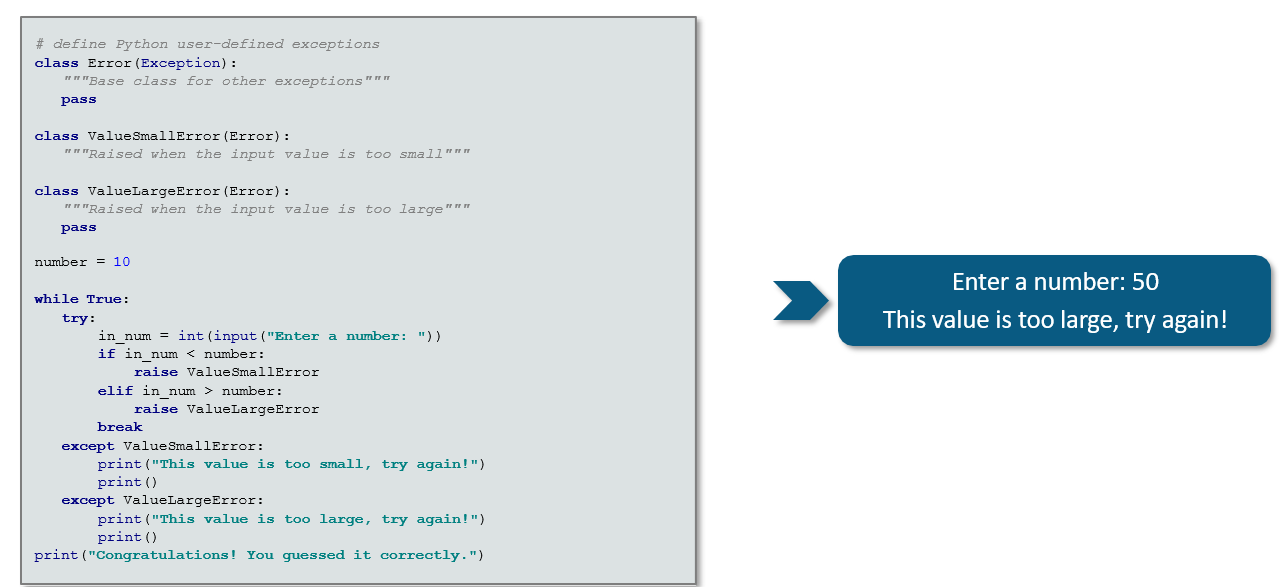


## 9.4 Create User Defined Exception

**What is User Defined Exceptions?**



**Example: User Defined Exceptions**

****

