**OBJECT:**

**Creating an Object**

* Python3

|  |
| --- |
| **class** Cars:  **def** \_\_init\_\_(self, m, p):      self.model **=** m      self.price **=** p    Audi **=** Cars("R8", 100000)    print(Audi.model)  print(Audi.price) |

**Output:**

R8

100000

**Working of the Program:**

Audi = Cars():

* A block of memory is allocated on the heap. The size of memory allocated is decided by the attributes and methods available in that class(Cars).
* After the memory block is allocated, the special method [\_\_init\_\_](https://www.geeksforgeeks.org/__init__-in-python/)() is called internally. Initial data is stored into the variables through this method.
* The location of the allocated memory address of the instance is returned to the object(Cars).
* The memory location is passed to [self](https://www.geeksforgeeks.org/self-in-python-class/).

**Accessing Class Member Using Object:**

Variable and method of a class are accessible by using class objects or instances.

**Syntax:**

obj\_name.var\_name

Audi.model

obj\_name.method\_name()

Audi.ShowModel();

obj\_name.method\_name(parameter\_list)

Audi.ShowModel(100);

**Example 1:**

* Python3

|  |
| --- |
| # Python program to create instance  # variables inside methods    **class** Car:        # Class Variable      vehicle **=** 'car'        # The init method or constructor  **def** \_\_init\_\_(self, model):            # Instance Variable          self.model **=** model        # Adds an instance variable  **def** setprice(self, price):          self.price **=** price        # Retrieves instance variable  **def** getprice(self):  **return** self.price    # Driver Code  Audi **=** Car("R8")  Audi.setprice(1000000)  **print**(Audi.getprice()) |

**Output:**

1000000

**Example 2:**

* Python3

|  |
| --- |
| **class** Car:        # Class Variable      vehicle **=** 'Car'        # The init method or constructor  **def** \_\_init\_\_(self, model, price):            # Instance Variable          self.model **=** model          self.price **=** price    # Objects of class  Audi **=** Car("R8", 100000)  BMW **=** Car("I8", 10000000)    **print**('Audi details:')  print('Audi is a', Audi.vehicle)  print('Model: ', Audi.model)  **print**('price: ', Audi.price)    **print**('\n BMW details:')  **print**('BMW is a', BMW.vehicle)  print('Model: ', BMW.model)  **print**('Color: ', BMW.price)    # Class variables can be  # accessed using class  # name also  print("\nAccessing class variable using class name")  **print**(Car.vehicle)    # or  print(Audi.vehicle)    # or  **print**(BMW.vehicle) |

**Output:**

Audi details:

Audi is a Car

Model: R8

price: 100000

BMW details:

BMW is a Car

Model: I8

Color: 10000000

Accessing class variable using class name

Car

Car

Car

**Self Variable:**

SELF is a default variable that contains the memory address of the current object. Instance variables and methods can be referred to by the self variable. When the object of a class is created, the memory location of the object is contained by its object name. This memory location is passed to the SELF internally, as SELF knows the memory address of the object, so the variable and method of an object is accessible. The first argument to any object method is SELF because the first argument is always object reference. This process takes place automatically whether you call it or not.

The self parameter is a reference to the current instance of the class, and is used to access variables that belongs to the class.

It does not have to be named self , you can call it whatever you like, but it has to be the first parameter of any function in the class:

**Example:**

* Python3

|  |
| --- |
| **class** Test:  **def** \_\_init\_\_(Myobject, a, b):      Myobject.country **=** a      Myobject.capital **=** b    **def** myfunc(abc):  **print**("Capital of  " **+** abc.country **+**" is:"**+**abc.capital)    x **=** Test("India", "Delhi")  x.myfunc() |

**Output:**

Capital of India is: Delhi

**Note:** For more information, refer to [self in Python class](https://www.geeksforgeeks.org/self-in-python-class/)

**Deleting an Object:**

Object property can be deleted by using the del keyword:

**Syntax:**

del obj\_name.property

objects also can be deleted by del keyword:

**Syntax:**

del obj\_name