**Special variable :**

In python we have a special variable “ \_\_name\_\_ “ (double underscore)

Graphical user interface, application

Description automatically generated

What is main? – in C,C++ or Java background, it is a starting point of execution. It is same in python

Ex: the moment you run this code if this is your first code, as we might have many modules and if we want to run this module first, here demo.py is our first module. First module name is always main as it is the point of execution which means that where our code starts.

Main is the starting point of execution

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Now we want to use calc in demo.py

Graphical user interface, text, application, email

Description automatically generated

The reason is main from the calc file is being called hence we are observing both the result 1 and result 2

Now we need to say ‘hey I want to call main when only I am running this programme/calc.py as a standalone program- I don’t want to call main when I am running it from another file.

Now to prevent this from happening we need to mention \_\_name\_\_ so that it will have the value of main else it will have the calc.py(file name) here as the value

Graphical user interface, text, application

Description automatically generated

Now if we run demo.py

Graphical user interface, text, application

Description automatically generated

Now let’s run the calc.py as standalone file and observe the result.

Graphical user interface, text, application

Description automatically generated

Now run the demo.py and observe the result

Graphical user interface, text, application

Description automatically generated

* Demo.py

from calc import bunny

def uday():

    bunny()

    print('uday')

def puja():

    print('puja')

def main():

    uday()

    puja()

main()

def rishi():

    print('function - rishi')

def bunny():

    print('function - bunny')

def main():

    rishi()

    bunny()

# if \_\_name\_\_ == "\_\_main\_\_":

    main() # only activate this main when we run this as a standalone file/program py.exe calc.py

Now the module name is called but it didn’t call the main function.

Finally what we are trying to do is – we mentioned in the calc.py file if \_\_name\_\_ = “\_\_main\_\_” then only call main(which means to execute the included functions) otherwise other code will use this as module.

OOPS: Object Oriented Programming

Class – design or kind of like blueprint (by oneplus or apple)

Object

Attributes

Behavior – method

Functions in oops is methods

Instance – entity of a class (oneplus6,9,10 or iphone11,12 etc..)

A bald person wearing glasses

Description automatically generated with medium confidence

A picture containing text

Description automatically generated

Diagram

Description automatically generated with medium confidence

Now if I want to create a ***computer*** class as I want to work with computer(com1)

Graphical user interface, text, application

Description automatically generated

Below is how we define a class, discuss attribute and behaviour(method) – let’s discuss ***self*** later.

**Def config(self): is a method**

Graphical user interface, application

Description automatically generated

Str is a in built class and com1 is a class which we have created.

Think about this

A = 1 #will become a int

A = 1.5 #then a will become a float

C = ‘1’ # Q what it will become?

However com1 doesn’t have a type , we have to mention that com1 is an object to computer class.

By mentioning com1 = computer() # in other programming languages like java it is called constructor.

Let’s took at the object types using the type function as shown below and see what it returns.

Module name is main because we are running it from the same file/here, and class as ***computer.***

For a it is a str

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

* Now we know that we have created a class and we got a object and how do we call it?

If we type config() it says unresolved reference ‘config’ more…

Graphical user interface, text, application

Description automatically generated

So if we want to use a method, 1) we need to mention the class name(computer) and we say config(method) as shown below, we will get error why? Check the error message

Graphical user interface, text, application

Description automatically generated

So we need to mention on which object we need to run the method on some object that we have created, here in this case it is ***com1.*** So now we can link it to ***self*** parameter.

Graphical user interface, text, application

Description automatically generated

class computer:

    def config(self):

        print('i5, 16gb, 1tb')

com1 = computer()

computer.config(com1)

* Let’s create some more objects and see how it works.

Graphical user interface, text, application

Description automatically generated

* Now we can use the object itself to call the method as shown below, we used 2 objects to call the method config() and it works.
* Check this as well.

a = 5

a.bitlength #click ctrl + mouse click will take you to the documentation which shows (self). Self is the object we are passing—here it is ***a*** which is being passed as a parameter/argument to the method bitlength.

Graphical user interface, application

Description automatically generated

<https://www.programiz.com/python-programming/examples/multiplication-table>

<https://www.programiz.com/python-programming/examples/armstrong-number>

<https://www.programiz.com/python-programming/examples/sum-natural-number>