OOPS

April 1, 2024

[1]:
```
a = 1
```

[2]:
```
print(type(a)) # inbuilt class
```

```
<class 'int'>
```

[3]:
```
print(type('pwskills'))
```

```
<class 'str'>
```

[4]:
```
# class is a classification or a blueprint for a real world entity
# It is a collection of object
# It is a logical entity tht contains some attributes and methods
# Object is any real world object or entity.
```

[5]:
```
class test :
    pass # blank class
```

[6]:
```
#OOPS enhances the coding capability and provides structure to a code.
# Codes become more reusable.
```

[7]:
```
a = test()
```

[8]:
```
type(a)
```

[8]:
```
__main__.test
```

[9]:
```
print(type(a))
```

```
<class '__main__.test'>
```

[11]:
```
class pwskills:
    def wel_msg():
        print('welcome to pwskills')
```

[12]:
```
rohan = pwskills()
```

[14]:
```
print(type(rohan))
```

```
<class '__main__.pwskills'>
```

[15]:
```
rohan.wel_msg()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[15], line 1
----> 1 rohan.wel_msg()

TypeError: pwskills.wel_msg() takes 0 positional arguments but 1 was given
```

[16]:
```
# All the methods written inside a class must be bind with the class
# So that class understands that this is my function
```

[18]:
```
class pwskills :
    def wel_msg(self):
        print('welcome to pwskills')
```

[19]:
```
rohan = pwskills()
```

[20]:
```
rohan.wel_msg()
```

```
welcome to pwskills
```

[21]:
```
# We can create different methods/functions inside a class
```

[22]:
```
# Now to acess the class methods outside it first we need to create
# class variable or class object or instance of a class
# which is rohan in the above case.
```

[23]:
```
gaurav = pwskills()
```

[25]:
```
gaurav.wel_msg()
```

```
welcome to pwskills
```

[26]:
```
# If we want to pass the data inside a class then we will use a
# __init__() aka constructor
```

[27]:
```
class pwskills1 :

    # inbuild func used to pass data inside a class

    def __init__(self,phone_number,email_id,stu_id):
        self.phone_number = phone_number
        self.email_id = email_id
        self.stu_id = stu_id
```

```
    def return_stu_details(self) :
        return self.stu_id,self.phone_number,self.email_id
```

[30]:
```
rohan = pwskills1(987456123,'abc@gmail.com',125)
```

[32]:
```
rohan.return_stu_details()
```

[32]:
```
(125, 987456123, 'abc@gmail.com')
```

[33]:
```
# self keyword is used to bind the var with the class
```

[34]:
```
gaurav = pwskills1(78456313,'xyz@fmail.com',126)
```

[35]:
```
gaurav.return_stu_details()
```

[35]:
```
(126, 78456313, 'xyz@fmail.com')
```

[36]:
```
gaurav.phone_number
```

[36]:
```
78456313
```

[37]:
```
gaurav.email_id
```

[37]:
```
'xyz@fmail.com'
```

[38]:
```
# self is not a reserved keyword.
```

[39]:
```
# Inside a class always give first parameter of a function as self which
# which can  act as pointer and bind the func var to  a class
```

[ ]:
```

```