OOPS_Part_1(DATA_MINDS)

September 13, 2023

OOPS - OBJECT ORIENTED PROGRAMMING SYSTEM

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

[3]:
```
print(type(a))
```

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

[4]:
```
print(type("Virat Tiwari"))
```

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

[5]:
```
print(type(25.04))
```

```
<class 'float'>
```

[6]:
```
print(type([22,5,6,4,9,2]))
```

```
<class 'list'>
```

[11]:
```
print(type({"Name":"Virat Tiwari","Age":22}))
```

```
<class 'dict'>
```

CONCEPT OF CLASSES - " CLASS " IS NOTHING BUT A CLASSIFICATION THAT IDENTIFY THE REAL WORLD ENTITY OR BLUEPRINT OF REAL WORLD ENTITY BUT NOT SPECIFIC TO A REAL WORLD ENTITY

CONCEPT OF OBJECT - " OBJECT " IS A REAL WORLD ENTITY , WE CALL OBEJCT AS "VARIABLE" OR "INSTANCE" AS WELL

IMPORTANT - FOR CREATING "CLASS" IN PYTHON WE USE "CLASS" KEYWORD

[13]:
```
#THIS IS HOW WE CREATE BLANK CLASS BY USING THE "PASS" KEYWORD

class test:
    pass
```

WHY WE CREATE CLASS IN PYTHON -:

WHENEVER WE WORK ON PROJECT WE CREATES DIFFERENT TYPES OF MODULES , DATABASE MODULE , BACKEND , FRONTEND MODULES SO MAKE ALL THESE IN STRUCTURED FORM WE USE CLASSES , CLASSES SIMPLY GAVE OUR CODE A STRUCTURED FORM THAT HELPFUL FOR THE OTHER DEVELOPERS WHO WANTS TO CONTRIBUTE IN THAT CODE IN FUTURE AND IT ALSO ENHANCE THE REUSABILITY OF CODE

WHY WE USE OBEJCT ORIENTED PROGRAMMING - :

IT SIMPLY MAKE OUR CODE STRUCTURED FOR FURTHER USE AND ENHANCE THE RESUABILITY OF CODE SO OTHER DVELOPERS CAN EASILY CONTRUBUTION IN THAT CODE IN FUTURE IF THEY WANT

[15]:
```
class test:
    pass
```

[16]:
```
a=test()
```

[17]:
```
type(a)
```

[17]:
```
__main__.test
```

[21]:
```
print(type(a))
```

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

” SELF ” - THE FUNCTION THAT WE WRITE INSIDE THE CLASS , IT SHOULD BE BINDED WITH THE CLASS THAN CLASS UNDERSTAND THE METHOD INSIDE THE FUNCTION SO WE USE ” SELF ” AS A VERY VERY FIRST PARAMETER

[32]:
```
class DataMinds:
    def welcome_msg(self):
        print("Welcome to Data Minds")
```

[33]:
```
Virat=DataMinds()
```

[34]:
```
print(type(Virat))
```

```
<class '__main__.DataMinds'>
```

[35]:
```
Virat.welcome_msg()
```

```
Welcome to Data Minds
```

[37]:
```
Yash=DataMinds()
```

[38]:
```
Yash.welcome_msg()
```

```
Welcome to Data Minds
```

[39]:
```
Rohit=DataMinds()
```

[40]:
```
Rohit.welcome_msg()
```

```
Welcome to Data Minds
```

“ ” **init** “ ” = THIS IS INBUILT FUNCTION , IT IS KNOWN AS INITIALISATION , WHENEVER WE WANT TO PASS THE DATA OR SOMETHING INSIDE THE CLASS THAN WE WILL CALL THIS FUNCTION OR SIMPLY USE IT.

IT IS SIMPLY A CONSTRUCTOR.

[49]:
```
class DataMinds:
    def __init__(self,mob_no,email_id,student_id):
        self.mob_no=mob_no
        self.email_id=email_id
        self.student_id=student_id

    def return_student_details(self):
        return self.student_id,self.mob_no,self.email_id
```

[71]:
```
Virat.mob_no
```

[71]:
```
2014567
```

[75]:
```
Virat.email_id
```

[75]:
```
'Virat@gmail.com'
```

[76]:
```
Virat.student_id
```

[76]:
```
21
```

[50]:
```
#NOTE - DUE IT __INIT__ FUNCTION IT REQUIRE THE DATA WHENEVER WE EXECUTE THE␣
 ↪CODE

Virat=DataMinds()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[50], line 3
      1 #NOTE - DUE IT __INIT__ FUNCTION IT REQUIRE THE DATA WHENEVER WE EXECUTE␣
 ↪THE CODE
----> 3 Virat=DataMinds()

TypeError: DataMinds.__init__() missing 3 required positional arguments:␣
 ↪'mob_no', 'email_id', and 'student_id'
```

```
[52]: Virat=DataMinds(123654789,"Virat@gmail.com",101)
```

NOTE - ” SELF ” IS NOTHING BUT THE WAY TO ACCESS THE CLASS VARIABLE

```
[55]: Virat.return_student_details()
```

```
[55]: (101, 123654789, 'Virat@gmail.com')
```

```
[56]: Yash=DataMinds(54698726,"Yahs@gmail.com",102)
```

```
[57]: Yash.return_student_details()
```

```
[57]: (102, 54698726, 'Yahs@gmail.com')
```

```
[58]: Rohit=DataMinds(94567,"Rohit@gmail.com",103)
```

```
[59]: Rohit.return_student_details()
```

```
[59]: (103, 94567, 'Rohit@gmail.com')
```

```
[62]: Happy=DataMinds(80467,"Happy@gmail.com",104)
```

```
[65]: Happy.return_student_details()
```

```
[65]: (104, 80467, 'Happy@gmail.com')
```

```
[83]: class DataMinds1:
          def __init__(self,mob_no1,email_id1,student_id1):
              self.mob_no1=mob_no1
              self.email_id1=email_id1
              self.student_id1=student_id1

          def return_student_details(self):
              return self.student_id1,self.mob_no1,self.email_id1
```

```
[93]: Virat=DataMinds1(2014567,"Virat@gmail.com",22)
```

```
[94]: Virat
```

```
[94]: <__main__.DataMinds1 at 0x7fb05049efb0>
```

```
[95]: Virat.student_id1
```

```
[95]: 22
```

```
[98]: Virat.return_student_details()
```

```
[98]: (22, 2014567, 'Virat@gmail.com')
```

```
[101]: Virat.email_id1
```

```
[101]: 'Virat@gmail.com'
```

```
[102]: class DataMinds1:
           def __init__(virat,mob_no1,email_id1,student_id1):
               virat.mob_no1=mob_no1
               virat.email_id1=email_id1
               virat.student_id1=student_id1

           def return_student_details(virat):
               return virat.student_id1,virat.mob_no1,virat.email_id1
```

```
[105]: Virat=DataMinds1(984036,"Virat20@gmsil.vom",105)
```

```
[106]: Virat.email_id1
```

```
[106]: 'Virat20@gmsil.vom'
```

NOTE - ” SELF ” IS NOT A RESERVED KEYWORD , WE CAN USE ANYTHING IN PLACE OF SELF , WE USE OUR NAME , ANYTHING THAT WE WANT TO WRITE .

IN THE UPCOMING SESSIONS WE WILL DEEP DIVE MORE N MORE IN OOPS

STAY TUNED - VIRAT TIWARI

```
[ ]:
```