ENCAPSULATION

Encapsulation is the fundamental concept in OOPS and is one of the pillar of OOP concept. It is the concept binding data and methods together in a class, protecting data integrity and promoting modularity. It is a concept of hiding data by making it private. In another words we can say it is a combination of data hiding and abstraction.

Let's take a real life example:- Consider a company, it will be having a sales team and a finance team. Sales team deals with the sales related matter and they have documents for their accessment similarly finance team deals with finance related matter and have documents for their accessment. So let's consider a member of sales team need a file from finance team,he does not have the authority to directly access the files instead he needs to request the finance officer to access the file. This is what encapsulation does in a code it protect the members from unauthorized access outside the class.

Encapsulation is achieved by declaring a class's data members and methods as either private or protected. Python generally possess 3 access modifiers namely public,protected,private.

- Public:These type are accessible anywher inside the class
- Protected: These types are accessible within the class
- Private: These types are accessible in the a class and it's sub classes

IMPLEMENTATION OF ENCAPSULATION

```
#Code to print details and Project updation of an employee


class employee:
  def __init__(self,name,id,project,salary):
    self.name = name
    self.id = id
    self.__project = project
    self.__salary = salary


  def show_details(self):


    print(f"NAME:", self.name)
    print("Employee id:",self.id)
    print("Salary:",self.__salary)
  def proj(self):
    print("Updates on my project")
    print("I'm currently working on the project named" ,self.__project)


obj = employee("JACKS","RA789","PYTHON ADVANCED",56000)


print(obj.show_details())
print(obj.proj())
```