In [3]:
```
#Instance method-->operates on object instance,it can be access(accessor) and modif
class Person:

    def __init__(self,name):
        self.name=name
    def dispaly(self):
        print(f"Hello,My name is{self.name}")
p1=Person("Shamsheera")
p1.dispaly()
```

```
Hello,My name isShamsheera
```

In [22]:
```
class Student:
    #class variable
    school_name="New millenium"
    def __init__(self,name,grade):#instance var
        self.name=name
        self.grade=grade
    #instance method
    def show(self):
         print(f"Student name is{self.name} Studies in grade{self.grade}at {Student
    #class method:-->to acces/modify class variable
    #cls refers to the class not object
    @classmethod
    def change_school(cls,new_name):
        cls.school_name=new_name

s1=Student("anu","A")
s1.show()
s2=Student("Divya","B")
s2.show()
Student.change_school("Wisdom school")
s1.show()
```

```
Student name isanu Studies in gradeAat New millenium
Student name isDivya Studies in gradeBat New millenium
Student name isanu Studies in gradeAat Wisdom school
```

In [24]:
```
class student:
    School_name = "Ambedkar school"

    def __init__(self, name, grade):
        self.name = name
        self.grade = grade

    def show(self):
        print(f"My Name is {self.name}, my Grade is {self.grade}, My School Name is

n = input("Enter Your Name: ")
o = input("Enter Your Grade: ")
st = student(n, o)
st.show()
```

```
My Name is govardhan, my Grade is A, My School Name is Ambedkar school.
```

In [28]:
```python
#static method-->does not depends on the object instance or class var.it look like
class MathsCal:
    @staticmethod
    def add(a,b):
        return a+b
result=MathsCal.add(10,30)
print("Sum:",result)
```

```
Sum: 40
```

In [36]:
```python
class Employee:
    company="Anudip"
    def __init__(self,name):
        self.name=name
    def show_name(self):#instance method
        print("Employee name:",self.name)
    @staticmethod
    def company_info():#static method
        print("Working Time:9 AM to  PM")
e=Employee("Harha")
e.show_name()#instance method
e.company_info()#static method
```

```
Employee name: Harha
Working Time:9 AM to  PM
```

In [40]:
```python
class BankAccount:
    bank_name="SBI"#class var
    interest_rate=0.05#5%
    def __init__(self,name,balance):
        self.name=name
        self.balance=balance
        #instance method
    def deposit(self,amount):
        self.balance+=amount
        print(f"{self.name} deposited {amount}.New Balnce:{self.balance}")
    #instance method
    def withdraw(self,amount):
        if amount<=self.balance:
            self.balance-=amount
            print(f"{self.name}withdraw {amount}.New Balnce:{self.balance}")
        else:
            print("insufficient balance")
    #class metho
    @classmethod
    def set_interest_rate(cls,rate):
        cls.interest_rate=rate
        print(f"New interest rate is set to {cls.interest_rate*100}%")
   #static method
    @staticmethod
    def validate_accountNo(accountNo):
        if len(accountNo)==10 and accountNo.isdigit():
            return True
        else:
            return False
```

```
#creating account
a1=BankAccount("A",10000)
a2=BankAccount("B",15000)
#instance method
a1.deposit(500)
a2.deposit(300)
a1.withdraw(5000)
a2.withdraw(500)
#class method
BankAccount.set_interest_rate(0.07)
#static method
print(BankAccount.validate_accountNo("1234567890"))
print(BankAccount.validate_accountNo("12AB890"))
```

```
A deposited 500.New Balnce:10500
B deposited 300.New Balnce:15300
Awithdraw 5000.New Balnce:5500
Bwithdraw 500.New Balnce:14800
New interest rate is set to 7.000000000000001%
True
False
```

In [ ]: