```
#OOP    => Object oriented programming
#class  => Person => blueprint =>template => structure
#Object => Ram =>Actually exists / instance of class
```

```
class Person:
    name ='ram'#attribute/property
    age = 22
    address='ktm'
    gender ="male"
    phone =7673572

    def eat(): #method/action/behaviour/characteristics
        #class vitra ko functions haru methods ho
        print("eating...")
    def sleep():
        print("sleeping...")
    def walk():
        print("Walking...")
```

```
p1= Person()
print(p1)
```

```
<__main__.Person object at 0x0000028308A8BC50>
```

```
# yeuta class ko jati otaa objects ni banauna melchha
p2 = Person() #p2 is object
print(p2)
```

```
<__main__.Person object at 0x0000028308A8BB10>
```

```
p1.name
```

```
'ram'
```

```
class Person:
   def __init__(self, name,age,address):
    print(name,age,address)

    #action/behaviour/characteristics/method
   def eat (self): # vitra self lekhesi function methos ma convert hunchha
     print("eating...")

   def walk (self): # vitra self lekhesi function methos ma convert hunchha
      print("walking...")

   def sleep (self): # vitra self lekhesi function methos ma convert hunchha
     print("sleeping...")
```

```
p1 = Person(name="Ram", age=22, address='ktm')
```

```
Ram 22 ktm
```

```
class Person:
   def __init__(self, name,age,address):
      self.name = name
      self.age = age
      self.address = address

    #action/behaviour/characteristics/method
   def eat (self):
     print("eating...")

   def walk (self):
      print("walking...")

   def sleep (self):
     print("sleeping...")
```

```
p1 = Person(name="Ram", age=22, address='ktm')
```

```
p1.name
```

```
'Ram'
```

```
p2 = Person('Shyam',22,'btr')
p2.name
```

```
'Shyam'
```

```
#name,age,address,college,faculty,roll-no
class Student:
    name='ram'
    age=22
    address="syangja"
    college="cosmos"
    faculty="BE Computer"
    roll_no =45

S1= Student()
```

```
print(S1)
S1.name
S1.age
S1.address
```

```
<__main__.Student object at 0x0000028308C82660>
'ram'
```

```
#dynamicically
#name,age,address,college,faculty,roll-no
class Student:
    def __init__(self, name,age,address,college,faculty,roll_no):
        self.name=name
        self.age=age
        self.address=address
        self.college=college
        self.faculty=faculty
        self.roll_no=roll_no

    #action/method
    def learn(self):
        print("stuydent is learning")

s1 = Student(name="Ram",age=22,address="syangja",college="cosmos",faculty="BCE",roll_no=45)
```

```
s1.learn()
```

```
stuydent is learning
```

```
#dynamicically
#name,age,address,college,faculty,roll-no
class Student:
    def __init__(self, name,age,address,college,faculty,roll_no):
        self.name=name
        self.age=age
        self.address=address
        self.college=college
        self.faculty=faculty
        self.roll_no=roll_no
        self.subjects =[] #subjects ma thapna melne banauna ko lagi

    #action/method
    def learn(self):
        print("stuydent is learning")

    def add_Subject(self, Subject_name):
        self.subjects.append(Subject_name)
```

```
s1 = Student(name="Ram",age=22,address="syangja",college="cosmos",faculty="BCE",roll_no=45)
```

```
s1.subjects
```

```
[]
```

```
s1.add_Subject("python")
```

```
s1.subjects
```

```
['python']
```

```
s1.add_Subject("django")
```

```
s1.subjects
```

```
['python', 'django']
```

```
s1.add_Subject("django") # fei django nao thapda 2 patak django django aayo
s1.subjects
```

```
['python', 'django', 'django']
```

```
 # jati patak yeutei subject aauda pani repeat nahune garnu paryo
class Student:
    def __init__(self, name,age,address,college,faculty,roll_no):
        self.name=name
        self.age=age
        self.address=address
        self.college=college
        self.faculty=faculty
        self.roll_no=roll_no
        self.subjects =[] #subjects ma thapna melne banauna ko lagi

    #action/method
    def learn(self):
        print("stuydent is learning")

    def add_subject(self, subject_name):
        if subject_name not in self.subjects:
          self.subjects.append(subject_name)
```

```
s1 = Student(name="Ram",age=22,address="syangja",college="cosmos",faculty="BCE",roll_no=45)
```

```
s1.add_subject("django")
```

```
s1.add_subject("django")
```

```
s1.subjects
```

```
['django']
```

```
#Rectangle(L,b)
#Area = l*b
#perimeter= 2*(l+b)
```

```
class Rectangle:
    def __init__(self, length,breadth):
        self.l=length
        self.b=breadth
    def area(self):
        return self.l *self.b
    def perimeter(self):
        return 2 *( self.l + self.b)
```

```
r1 = Rectangle(4,8)
print(r1.area())
print(r1.perimeter())
```

```
32
24
```

```
r2 = Rectangle(5,5) # arko input dena arko object banayeko ho
print(r2.area())
print(r2.perimeter())
```

```
25
20
```

```
class Person:
    def __init__(self, name ,age, address):
        self.name = name
        self.age = age
        self.address = address

    def eat(self):
        print(f'{self.name} is eating')

    def sleep(self):
        print(f'{self.name} is sleeping')

    def walk(self):
        print(f'{self.name} is walking')

    def info(self):
        print(f"Name: {self.name}")
        print(f"Age: {self.age}")
        print(f"Address: {self.address}")

class Student:
    def __init__(self, name, age, address, college, faculty, roll_no):
        self.name = name
        self.age = age
        self.address = address
        self.college = college
        self.faculty = faculty
        self.roll_no = roll_no
        self.subjects = []

    def learn(self):
        print(f"Student is learning: {self.subjects}")

    def add_subject(self, name):
        if name not in self.subjects:
            self.subjects.append(name)

    def eat(self): # self is object
        print(f'{self.name} is eating')

    def sleep(self):
        print(f'{self.name} is sleeping')

    def walk(self):
        print(f'{self.name} is walking')

    def info(self):
        print(f"Name: {self.name}")
        print(f"Age: {self.age}")
        print(f"Address: {self.address}")
        print(f"College: {self.college}")
        print(f"Faculty: {self.faculty}")
        print(f"Roll no: {self.roll_no}")
        print(f"Subjects: {self.subjects}")


class Person:
    def __init__(self, name ,age, address):
```

```
            self.name = name
            self.age = age
            self.address = address

        def eat(self):
            print(f'{self.name} is eating')

        def sleep(self):
            print(f'{self.name} is sleeping')

        def walk(self):
            print(f'{self.name} is walking')

        def info(self):
            print(f"Name: {self.name}")
            print(f"Age: {self.age}")
            print(f"Address: {self.address}")

    class Student(Person):#Inheritance
        def __init__(self, name, age, address, college, faculty, roll_no):
            self.name = name
            self.age = age
            self.address = address
            self.college = college
            self.faculty = faculty
            self.roll_no = roll_no
            self.subjects = []

        def learn(self):
            print(f"Student is learning: {self.subjects}")

        def add_subject(self, name):
            if name not in self.subjects:
                self.subjects.append(name)
        def info(self):
            print(f"Name: {self.name}")
            print(f"Age: {self.age}")
            print(f"Address: {self.address}")
            print(f"College: {self.college}")
            print(f"Faculty: {self.faculty}")
            print(f"Roll no: {self.roll_no}")
            print(f"Subjects: {self.subjects}")


    s1 = Student(name="Ram",age=22,address="syangja",college="cosmos",faculty="BCE",roll_no=45)
    s1.walk()
```

```
Ram is walking
```

```
    #inheritance



    class Parent:
        def hello(self):
            print("Hello World!")
    class Child(Parent): #single inheritance
        pass


    c = Child()


    c.hello()
```

```
Hello World!
```

```
    class Person:
        def __init__(self, name ,age, address):
            self.name = name
            self.age = age
            self.address = address

        def eat(self):
            print(f'{self.name} is eating')

        def sleep(self):
```

```
            print(f'{self.name} is sleeping')

        def walk(self):
            print(f'{self.name} is walking')

        def info(self):
            print(f"Name: {self.name}")
            print(f"Age: {self.age}")
            print(f"Address: {self.address}")

    class Student(Person):# Single Inheritance
        def __init__(self, name, age, address, college, faculty, roll_no):
            super().__init__(name, age, address)  #calling person's init
            self.college = college
            self.faculty = faculty
            self.roll_no = roll_no
            self.subjects = []

        def learn(self):
            print(f"Student is learning: {self.subjects}")

        def add_subject(self, name):
            if name not in self.subjects:
                self.subjects.append(name)
        def info(self):
            super().info() #calling person's info method
            print(f"College: {self.college}")
            print(f"Faculty: {self.faculty}")
            print(f"Roll no: {self.roll_no}")
            print(f"Subjects: {self.subjects}")


    s1 = Student(name="Ram",age=22,address="syangja",college="cosmos",faculty="BCE",roll_no=45)
    s1.walk()
```

```
Ram is walking
```

```
     #Multiple inheritance
    class Father:
        pass
    class Mother:
        pass
    class Child (Father,Mother): #Multiple Inheritance
        pass


    #exaxmple of Multiple inheritance

    #Parent class 1
    class Person:
        def person_info(self,name,age):
            print('Inside person class')
            print('Name:',name ,'Age:',age)

    #Parent class 2
    class Company:
        def company_info(self,company_name,location):
            print('Inside company name')
            print('Name:',company_name, 'location:',location)

    #child class
    class Employee(Person,Company):
        def employee_info(self,salary,skill):
             print('Inside Employee class')
             print('Salary:',salary, 'skill:',skill)

    #creating object of Employee
    emp = Employee()

    #acess data
    emp.person_info('Jessa',28)
    emp.company_info('Google','Atlanta')
    emp.employee_info(120000,'Machine learning')
```

```
Inside person class
Name: Jessa Age: 28
Inside company name
```

```
Name: Google location: Atlanta
Inside Employee class
Salary: 120000 skill: Machine learning
```

Start coding or generate with AI.