```
A class is a user-defined blueprint or prototype from which objects are
created. Classes provide a means of bundling data and functionality together.
Creating a new class creates a new type of object, allowing new instances of
that type to be made. Each class instance can have attributes attached to it
for maintaining its state.
Class instances can also have methods (defined by their class) for modifying
their state.
```

```
Some points on Python class:

Classes are created by keyword class.
Attributes are the variables that belong to a class.
Attributes are always public and can be accessed using the dot (.) operator.
Eg.: My class.Myattribute
```

```
An Object is an instance of a Class. A class is like a blueprint while an
instance is a copy of the class with actual values
```

In [20]:
```
class demo:
    def __init__(self,name):
        self.name=name
        print(f"welcome {self.name}")
```

In [21]:
```
obj=demo("Supriya")
```

```
welcome Supriya
```

In [22]:
```
obj.name
```

Out[22]:
```
'Supriya'
```

```
Define a Python function student(). Using function attributes display the
names of all arguments.
```

In [54]:
```
def students(st_name,stud_class,course):
    print(f"name is : {st_name} \nclass is : {stud_class} \ncourse is : {course
```

In [55]:
```
n=details()
```

In [56]:
```
students("Riya","V","New")
```

```
name is : Riya
class is : V
course is : New
```

```
Write a Python class named Rectangle constructed from length and width and a
method that will compute the area of a rectangle.
```

In [67]:
```
class rectangle:
    def __init__(self,l,w):
        self.l=l
        self.w=w
    def area(self):
        a=self.l*self.w
        print("area of rectangle is : ",a)
```

In [69]:
```
ra=rectangle(3,4)
ra.area()
```

```
area of rectangle is :  12
```

Write a Python class named Circle constructed from a radius and two methods that will compute the area and the perimeter of a circle.

In [79]:
```
class circle:
    def __init__(self,r):
        self.radius=r
    def perimeter(self):
        p=2*3.14*self.radius
        print("perimeter of circle : ",p)
    def area(self):
        a=3.14*self.radius**2
        print("area of circle : ",a)

```

In [80]:
```
c=circle(3)
```

In [81]:
```
c.perimeter()
```

```
perimeter of circle :  18.84
```

In [82]:
```
c.area()
```

```
area of circle :  28.26
```

Write a Python class that has two methods: get_String and print_String , get_String accept a string from the user and print_String prints the string in upper case.

In [86]:
```
class string:
    def __init__(self):
        self.s=""
    def get_String(self):
        self.s=input()
    def print_String(self):
        print(self.s.upper())

```

In [88]:
```
s=string()
```

In [90]:
```
s.get_String()
s.print_String()
```

```
anvi
ANVI
```

```
Write a Python class to reverse a string word by word.
Input string : 'hello .py'
Expected Output : '.py hello'
```

In [95]:
```
class reverse_string:
    def __init__(self,s):
        self.string=s
    def rev(self):
        s1=self.string.split()
        s1.reverse()
        o=" ".join(s1)
        print(o)
```

In [96]:
```
r=reverse_string("hello .py")
```

In [97]:
```
r.rev()
```

```
.py hello
```

In [100]:
```
# class to find maximum of 3 numbers
class large_number:
    def __init__(self,a,b,c):
        self.a=a
        self.b=b
        self.c=c
    def method(self):
        l=[]
        l.append(self.a)
        l.append(self.b)
        l.append(self.c)
        m=max(l)
        print("maximum number is : ",m)
```

In [101]:
```
lr=large_number(3,4,2)
lr.method()
```

```
maximum number is :  4
```

```
In [103]: class large_number:
              def __init__(self,a,b,c):
                  self.a=a
                  self.b=b
                  self.c=c
              def method(self):
                  if self.a>self.b and self.a>self.c:
                      print(f"maximum number is : a = {self.a}")
                  elif self.b>self.a and self.b>self.c:
                      print(f"maximum number is : b = {self.b}")
                  else:
                      print(f"maximum number is : c = {self.c}")
```

```
In [104]: lr=large_number(3,4,2)
          lr.method()
```

```
maximum number is : b = 4
```

```
In [106]: lr=large_number(45,40,24)
          lr.method()
```

```
maximum number is : a = 45
```

```
In [116]: # write a class to sum all the numbers in a list
          class sum_list:
              def __init__(self,l):
                  self.list1=l
              def sum_i(self):
                  sum=0
                  for x in self.list1:
                      sum=sum+x
                  print("sum of numbers in a list is : ",sum)
```

```
In [117]: s=sum_list([2,3,1,4,5,6,7,8,9,10])
```

```
In [118]: s.sum_i()
```

```
sum of numbers in a list is :  55
```

```
In [119]: s=sum_list([2,3,4])
          s.sum_i()
```

```
sum of numbers in a list is :  9
```

In [122]:
```
# class to multiply all numbers in list
class multiplication:
    def __init__(self,l):
        self.list=l
    def product(self):
        prod=1
        for x in self.list:
            prod*=x
        print("multiplication of numbers in list is : ", prod)
```

In [123]:
```
m=multiplication([2,3,4])
m.product()
```

```
multiplication of numbers in list is :  24
```

In [124]:
```
m=multiplication([2,3,4,5])
m.product()
```

```
multiplication of numbers in list is :  120
```

In [125]:
```
# Python program to empty class
class empty:
    pass

# Driver's code
obj = empty()

print(obj)
```

```
<__main__.empty object at 0x000002216F07F850>
```

In [126]:
```
# create a class for employee details
class employee:
    def __init__(self,name,emp_id,department,salary):
        self.id=emp_id
        self.name=name
        self.department=department
        self.salary=salary
    def show(self):
        print(f"name :{self.name}")
        print(f"emp_id :{self.id}")
        print(f"department :{self.department}")
        print(f"salary :{self.salary}")
```

In [127]:
```
e=employee("rehan",1,"Gekko",20000)
```

In [128]:
```
e.show()
```

```
name :rehan
emp_id :1
department :Gekko
salary :20000
```

In [ ]:

```
```