Write an algorithm and program to design simple calculator performing arithmetic functions like addition, subtraction, multiplication and division with the input given by user. (12)

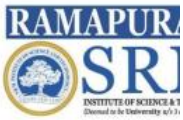
```
def add(x, y):
 return x + y

def subtract(x, y):
 return x - y

def multiply(x, y):
 return x * y

def divide(x, y):
 return x / y

print("Select operation.")
print("1.Add")
print("2.Subtract")
print("3.Multiply")
print("4.Divide")

while True:
 choice = input("Enter choice(1/2/3/4): ")
 if choice in ('1', '2', '3', '4'):
    num1 = float(input("Enter first number: "))
    num2 = float(input("Enter second number: "))
    if choice == '1':
      print(num1, "+", num2, "=", add(num1, num2))
    elif choice == '2':
     print(num1, "-", num2, "=", subtract(num1, num2))

    elif choice == '3':
     print(num1, "*", num2, "=", multiply(num1, num2))

    elif choice == '4':
         print(num1, "/", num2, "=", divide(num1, num2))
    break
 else:
   print("Invalid Input")
```

What is recursive function? Find a factorial of a given number using recursive function.(6)

Recursive functions are **functions that calls itself**. It is always made up of 2 portions, the base case and the recursive case. The base case is the condition to stop the recursion. The recursive case is the part where the function calls on itself.

```
def
recur_factorial(n):
                      if n == 1:
                        return n
                      else:
                        return n*recur_factorial(n-1)
                    num = 5
                    # check if the number is negative
                    if num < 0:
                      print("Sorry, factorial does not exist for negative
                    numbers")
                    elif num == 0:
                      print("The factorial of 0 is 1")
                    else:
                      print("The factorial of", num, "is",
                    recur_factorial(num))
```

Define function and its syntax.(4)



Functions

- In Python, a function is a group of related statements that performs a specific task.
- Functions help break our program into smaller and modular chunks. As our program grows larger and larger, functions make it more organized and manageable.
- Furthermore, it avoids repetition and makes the code reusable.

Syntax of Function

```
def function_name(parameters):
    """docstring"""
    statement(s)
```

24-08-2022

Python test.

NAME : MONISHA S
R.NO : RA1931005D2018
CLASS : IV B.Sc - CS "B"

classes & object

classes:-

(i) class is an blueprint of an object

(ii) Attributes are the variables of object of an class

(iii) Attributes are always public that can be accessed by dot operator (.)

eg:- class.self.

(iv) class should always begins with class keyword.

syntax:-

```
class. classname():
```

In the below example it shows that class is created by followed by class name dog. If the object of the class don't have attributes then we use pass keyword.

eg:-

```
class Dog():
    pass.
```

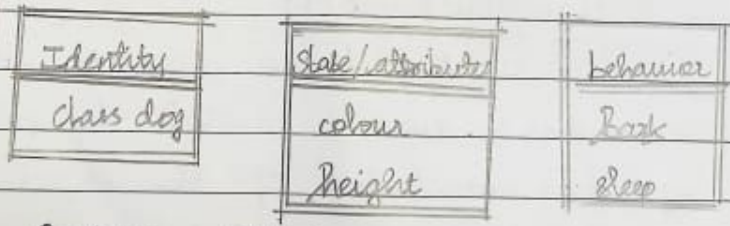
Object :-

Object is an instance of class, where class is an blueprint and instance contains actual copy of class values. Object consists of state, behaviour, identity.

State :- State represents the attributes of an object, it also reflects the property of the object of class.

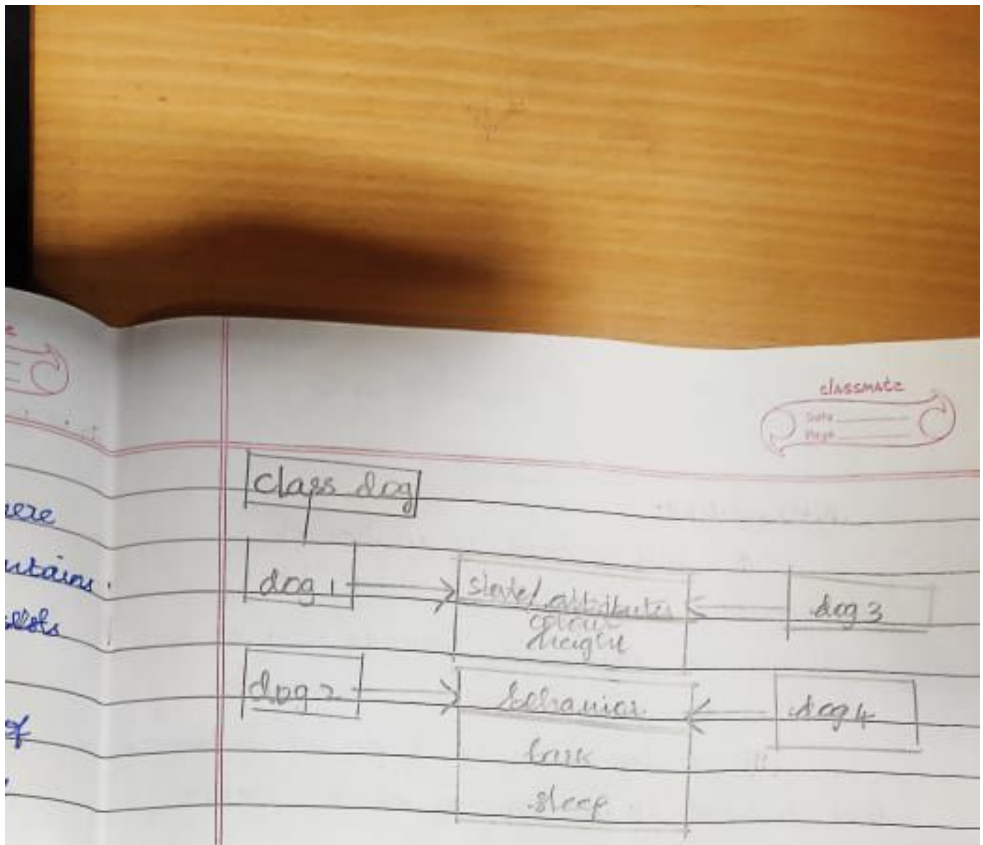
Behaviour :- Behaviour represents the methods of an object, it also reflects the response of one object to another object.

Identity :- Identity is an unique name of object of an class, it interact with a object to another object.



Declaring Object :-

When object of an class is created object instantiated. All the instance share the attributes and behaviour of object but the values are unique.



eg:-

```
attr1 = "mammal"
attr2 = "dog"
def fun (self):
Print ("I'm a", self.attr1)
Print ("I'm a", self.attr2)
Rodger = Dog()
print (Rodger.attr1)
 Rodger.fun()
```

output:-

Mammal

I'm a mammal

I'm a dog.

--init--

init() method:-

(i) init method is an type of constructor. constructor is used to initialise an object it runs as soon as object of the class is created. (constructor

(ii) constructor is an steps for information that executed at the time of object creation

eg:- Constructor: init method().

```
class Person:
    def __init__(self, name):
    self.name = name
    def sayhi(self):
    print("Hello, my name is", self.name)
    p = Person('Nikhil')
    p.say_hi()
```

output:-

Hello my name is Nikhil

In the above example class is created with the classname Person, p is an object name