**format () function—in python if I want to use a variable inside a string then I can use the format function**. Example

age=30;

name=’sandeep’

print(‘my name is Sandeep and I am 30 years old’)

now using format function

print(f’my name is {name} and I am {age} years old’)

another way to use format function

print('hello {} you are {} years old'.format(name,age))

-----------------------------------------------------------------------------------------------------------------------------------

**Single line comment** in python is given using #

And **multiline comments** are given is python using triple quotes “””

Ex:

#this is the prog to calculate sum of two digits

“””this is a prog

written in python

And is written by me”””

-----------------------------------------------------------------------------------------------------------------------------------

**If-else syntax in python**

If *condition*:

Statement

Else *condition*:

Statement

Ex:

year = 1985

if 2000<year<2099:

print('welcome to 21st centuary')

else:

print('you are before or after the 21st centuary')

--------------------------------------------------------------------------------------------------------------------------------

Concept related to functions in python

**1. function deffination**

Keyword def is used

No need to specify function return type

No need to specify the type of parameter function will accept. It automatically become the type of parameter that is being supplied during function call.

**Syntax:**

def functionname(parameter(s)):

function body

**syntax for function call:**

functionname(functionparameter)

**default value of function parameter in function definition:**

we can also give default value to function parameter in function definition. In that case if value of function parameter is not provided in function call then the parameter in the function definition takes the default value for that parameter.

**Function return value:**

A function may or may not have a return value.

We don’t need to specify what will be the type of return value will the function return.

**Syntax**

def functionname(parameter(s)):

function body

return value/variable/

syntax for function call in the case when function returns a value is:

variable=functionname(functionparameter)

**example:**

def hello(thestring):

print(‘hello’ + thestring)

hello(‘Sandeep’)

**example:**

def hello(name, age):

print('hello {} you are {} years old'.format(name,age))

hello("Sandeep", 35)

**example on default parameter:**

if in the above example if I forgot to provide value of any of the parameter during function call then error will be thrown up.

But if I have given some default value to the parameter during function definition then the parameter will take that value.

def hello(name, age=15):

print('hello {} you are {} years old'.format(name,age))

hello("Sandeep")

**example on function return**

def hello(name, age=15):

return f'hello {name} you are {age} years old'

xyz=hello("SandeepSharma", 36)

print(xyz)

--------------------------------------------------------------------------------------------------------------------------------------

**Boolean variable in python**

There are only two Boolean values in python

True

False

**Note: Boolean variable value that is true and false starts with capital letter that is True and False is correct but true and false are not correct Boolean values and will give error.**

Ex:

xyz= True

Here xyz is now a Boolean variable

if xyz:

Statement

else:

-------------------------------------------------------------------------------------------------------------------------------------

Concatenate strings

+ operator is used to concatenate two strings

Ex

thestring = ‘sandeep’

print(‘hello’ + thestring)

--------------------------------------------------------------------------------------------------------------------------------------

Lists concepts:

Arrays in pythons are called lists.

**How to create a list:**

Choose a variable name for the list say dogname. Then assign values to the list inside square bracket.

**Ex**

dogname= [‘tommy’, ‘jack’, ‘hunter’, ‘poland’]

**printing a list:**

print(name of the list)

**ex**

print(dogname)

**how to add/insert an element to the list**

nameofthelist.insert(index where you want to put the element, ‘element’)

Note: index start with 0 in python list

Ex:

Suppose I want to insert an element at the first position in the list

dogname.insert(0, ‘Natt’)

**how to print a particular element of the list**

print(nameofthelist[indexoftheelementtobedeleted])

ex

suppose I want to print the 4 element of the list

print(dogname[3])

**how to delete an element from the list**

del(nameofthelist[indexoftheelementtobedeleted])

ex

suppose I want to delete the first element of the list

del(dogname[0])

**how to print length(no of elements) of the list**

print(len(nameofthelist))

ex

suppose I want to print the length of the lsit dogname

print(len(dogname))

**how to replace an element of a list with some other element**

nameofthelist[indexoftheelementwhicistobereplaced]=’newelementvalue’

ex

suppose I want to replace jack with drane

dogname[1]=’drane’

**List in python may not be of the same type. It can have different type of elements. That means a list can have different type of variables and constants like intger,strings, double,Boolean, or constanvalues.**

**Ex:**

**dogname= [‘tommy’, ‘jack’, ‘hunter’, ‘poland’, 22, 45, 11.345]**

# createing a list

dognames=['jack', 'mat', 'polland', 'yuku']

#printing the list

print(dognames)

# inserting an object in the list say at 3rd position

dognames.insert(2,'richi')

print(dognames)

# i want to print 4th item of the list

print(dognames[3])

#i want to delete first object of the list

del(dognames[0])

print(dognames)

#i want to print the length(no of objects) present in the list

print(len(dognames))

#i want to replace object at say mat in the list with shane

dognames[1]='shane'

print(dognames)

#string can have any type of objects

newstring=['heeramal',55,True,False,41.652]

print(newstring)

--------------------------------------------------------------------------------------------------------------------------------------

LOOPS Concept in Python

**How to use for loop with Lists**

**Syntax:**

**For** <give any name to the list whose every element we want to loop into> **in** <name of the list>**:**

statements

**Ex:**

dognames=['jack', 'mat', 'polland', 'yuku']

for dog in dognames:

print(dog)

**how to use for loop other examples:**

for xyz in range(1,100):

print(xyz)

**Note: it will print from 1 to 9. That is one less than the last number. So always put at the end of the range one more than the number upto which you want.**

**How to use while loop in python:**

**Syntax:**

While <Boolean/statement resulting in boolean>:

Statements

Statements must contain a condition that will make the while condition false in order to come out of the while loop otherwise the loop will continue infinite times.

**Ex:**

Age=18

While Age<18:

Print(Age)

Age +=1

--------------------------------------------------------------------------------------------------------------------------------------

**Concept of Dictionary in Python:**

**What is dictionary?** : dictionary is a way to store information in key-value way rather than in ordered way like in list. We can look up a specific information that is value with a specific key associated with it.

**Syntax:**

name of the dictionary = {“key”:value, “key”:value, “key”:value} **//note use of curly braces and colon**

ex

dog={“kin”:4, “guugu”:8, “tremp”:15}

or

name of the dictionary = {} //put the key value pair later on by some means say for ex using a for loop.

**How to print a dictionary:**

Ex:

dog={“kin”:4, “guugu”:8, “tremp”:15}

print(dog)

**note: since the dictionary is not ordered like a list therefore whenever you will print the dictionary you may find the order of the elements changed**.

**How to print a value from the dictionary using its key:**

Ex:

dog={“kin”:4, “guugu”:8, “tremp”:15}

//suppose above the values are the ages of the dogs names which are the keys

Suppose I want to print the age of guugu

Print(dog[‘‘guugu’’])

**How to delete a key-value pair from a dictionary:**

Suppose I want to delete tremp

del(dog[“tremp”])

print(dog)

**how to add a key-value pair into a dictionary:**

syntax:

name of the dictionary[“key you want to create”]=value

ex:

suppose I want to add a dog named rutu whose age is 9 years in the dictionary created earlier named dog

dog[“rutu”]=9

print(dog)

**how to update a key-value pair in the dictionary:**

name of the lsit[“key that you want to update”]=value

example:

suppose I want to update the age of kin to 5 years

dog[“kin”]=5

print(dog)

**Note: Like List, in Dictionary also it is not necessary that all the keys or all the values need to be of same data type. They can be a combination of different data types in a dictionary. Like in the above example of dog we have made all the keys a string which may be of different data types not necessarily all to be string. Similarly we took all values to be integer which may be of different data type also**.

Ex:

dog={“kin”:4, “guugu”:False, “tremp”:15}

--------------------------------------------------------------------------------------------------------------------------------------

**CLASS & OBJECTS**

**how to create a class?**

**syntax**

class <name of the class>:

"""class is a key word all small letters. Name of the class may or may not start with a capital letter

however as a standard practice we shall write the first letter of the name of the class in capital."""

"""inside a class we can put methods. methods are nothing but functions inside a class"""

class Dog:

def bark():

print('BARK')

**Creating an object from a class**.

“””In order to create an object from a class also called an instance of a class

first we take a variable and then equate it to the object created"""

**syntax of object of a class**

variablename=objectname() note: the name of the object will be same as that of class

mydog = Dog()

**how can an object use method of its class**?

whenever an object of a class calls its method then the object itself is passed to that method as first parameter, so in order to receive that parameter we need to have a parameter in the

method definition which should be the first parameter. This parameter can have any name but for standard practice it is named as self.

class Dog:

def bark(self):

print('BARK')

mydog = Dog()

mydog.bark()

**how to provide multiple parameters to a method defined inside a class?**

class Dog:

def bark(self,xyz):

print(xyz)

mydog=Dog()

mydog.bark("testing")

**we can have properties (variables, methods) created for our objects, there is no exclusive requirement of creating properties inside a class**

class Dog:

def bark(self):

print('BARK')

mydog = Dog()

mydog.bark()

mydog.name="clark"

mydog.age=12

print(mydog.name, mydog.age)

**creating class variables**

class Dog:

doginfo="hey dogs are cool"

def bark(self):

print('BARK')

print(Dog.doginfo)

**how to update a class variable**

Dog.doginfo="yes right they are"

print(Dog.doginfo)

**creating an init method for a class. If we want that whenever any object of the class is created all the objects must have some properties then we can create an init method for that class.**

**syntax:**

def\_\_init\_\_(self,parameter,parameter,):

remember self will always be the first parameter for any method of a class

now we can also set the values of these parameters(properties).

self.parametername=parametername

now whenever i create an object of this class then i have to provide the value of these parameters at the time of creation of the object otherwise the object will not be created. thus this will ensure that any object of this class will have these properties.

class Dog:

def \_\_init\_\_(self,name,age,colour):

self.name=name

self.age=age

self.colour=colour

print("BARK")

mydog=Dog("cody",5,"brown")

print(mydog.age,mydog.name,mydog.colour)

However, if i will provide some **default value to the parameters** in the init method itself then i will not have to necessarly supply parameters at the time of creation of the object for this class

class Dog:

def \_\_init\_\_(self,name="",age=2,colour=""):

self.name=name

self.age=age

self.colour=colour

print("BARK")

mydog=Dog()

print(mydog.age)

**practice example on classes**

**Question** Add a method to the Car class called age that returns how old the car is (2020 - year)

\*Be sure to return the age, not print it\*

class Car:

def \_\_init\_\_(self,year, make, model):

self.year = year

self.make = make

self.model = model"""

**Solution:**

class Car:

def \_\_init\_\_(self,year, make, model):

self.year = year

self.make = make

self.model = model

def age(self):

return 2020-self.year

mycar=Car(2018,"hundai","vernasx")

#Age=mycar.age()

print(mycar.age(),mycar.make,mycar.model)

------------------------------------------------------------------------------------------------------------------------------------