**Python Theory**

**VARIABLES:**

Variable are containers for storing data. Variable are created the moment you declare or assign the value to it.

**DATA TYPES:**

Text type : str

Numerical: int float complex

Sequence: list tuple range

Mapping: dict (means declaring specific data. For ex: x = {"name": "John", "age": 36})

Set types: set (which is mutable ie. It can be changed even after creation), frozen set (which is immutable ie. It can’t change or modified after creation)

Boolen: bool (example: true or false, yes or no)

Nonetype : none

**Typecasting** means converting one data type into another.

(for example:

x = 5 int

y = 6.5 float

z = x + y

print (z) ) Output will be 11.5 means int automatically converted to float

**Take User input:** taking input by the user by using input() function:

You can also specify which type of data can taken in input() func

Example:

age = int(input("Enter your age: "))

print("You are", age, "years old.") <-- here we have mentioned that the input should be int or otherwise python will give error

**STRINGS:**

String in python are mentioned using single or double quotation

String Slicing: Strings are immutable

There are various method or function which will change the string for example: upper() lower()

rstrip() – removes any trailing or extra characters.

strip() - removes any extra spaces from the beginning and end of a string.

replace () – replace all the occurrences of a string with another string: Example: print (a.replace(“uma”, “bala”))

split() – make a list of string if there is any space.

capitalize() – capitalize the starting letter of the string

center() – aligns the string to the center as per the parameters

count()- it gives the number of times the value has occurred within the given string

endswith() – checks whether the string ends with the given value

find() – will the find the value in the given string and returns the index where it is present

isalnum() – return true it the string contains a-z 0-9. If there is any other character then it will return false

isalpha()- – return true it the string contains a-z, A-Z. If there is any other character or number then it will return false

islower()- return true it is lower or else false

isprintable() – returns true if all characters are printable or otherwise false.

Isspace() – return true if there is space

Istitle() – only if the each letter of the string is capital return true or else false.

Isupper() – same as lower

Swapcase()- converts the lower to upper and viceversa

Title()- turns the sentence into title , first letter of the word will be caps

**Conditional operators: > < , >=, <=, ==, !=**

**If else statement:**

*a = int(input("Enter your age: "))*

*if a >18:*

*print("You can drive")*

*else:*

*print("You can not drive")*

**Else if statement:**

*num = int(input("Enter the number: "))*

*if (num < 0):*

*print("Number is negative")*

*elif (num == 0):*

*print("Number is zero")*

*else:*

*print("Number is positive")*

N**ested conditions:**

num = 15

if (num == 0):

    print("Okay")

elif (num > 0):

    if (num <= 10):

        print("nice")

    elif (num > 10 and num < 20):

        print ("perfect")

    else:

        prin ("not good")

else:

    print("not fine")

MATCH CASE:

The match case statement in python allows you to perform pattern matching, making it easier to handle different conditions or data structure in a clean, readable way.

*x = int(input("Enter the number: "))*

*match x:*

*case 0:*

*print ("x is zero")*

*case \_ if x < 5:*

*print ("Lesser than five")*

*case \_ if x == 5:*

*print("x is odd number")*

*case \_:*

*print(x)*

**LOOPS:**

In python loops are mostly used to run a code until the condition is met.

There are two types of loops: 1. For loop 2. While loop.

1. **For loop**

*x = ["red", "yellow", "black", "white", "blue"]*

*for i in x:*

*print(i)*

*for j in i:*

*print(j)*

1. **While Loop**

In While loop, the loops keep on running until the given condition is true or met. As soon as the condition become false the python stops the execution and comes out of the while loop. While loop is usually not used with number rather, they are used for complex conditions.

*i = 0*

*while i < 3:*

*print (i)*

*i = i + 1*

***Decremental while loop: value keeps on decreasing.***

*i = 5*

*while i >= 1:*

*print (i)*

*i = i – 1*

Else in While loop:

*i = 5*

*while i >= 1:*

*print (i)*

*i = i - 1*

*else:*

*print("Vanakam")*

**How to emulate do while loop in python?**

We can’t use do while loop in python, but we can emulate the do while in python using break statement:

The loop will break once the if condition is met. Here in this example the loop will break once the user enters the negative number. The loop will keep on running until the user doesn’t enter any negative number

*while True:*

*number = int(input("Enter the positive number: "))*

*print(number)*

*if not number > 0:*

*break*

**Break and Continue in loops**

The break exits the loop and the continue exits the iteration.

**Continue:**

for i in range (1, 12):

*if (i == 10):*

*print ("Hello”)*

*continue # The loop will stop the iteration to print hello after 9th times and will continue to print 11th times*

*print ("5 x", i, "=", 5\*i)*

**Break:**

*for i in range (1, 12):*

*if (i == 10):*

*print ("Hey”)*

*break #the loop will break the execution and the last output will be hey it will not print the 11th times*

*print ("5 x", i, "=", 5\*i)*

**Functions**

A function is block of code that performs a specific task whenever it is called. There are two types of functions build-in functions and user defined function. For c-in func we don’t have to use def keyword and for user -def func we have to use the def keyword and this func is created by user themselves.

*def calculateGmean (a, b):*

*gmean = (a\*b)/(a+b)*

*print (gmean)*

*def isGreater (a, b):*

*if (a>b):*

*print("frist number is greater")*

*else:*

*print("second number is greater or equal")*

*a = 2*

*b = 3*

*# if (a>b):*

*#     print("frist number is greater")*

*# else:*

*#     print("second number is greater or equal")*

*isGreater (a,b)*

*calculateGmean (a, b)*

*c = 3*

*d = 5*

*isGreater (c,d)*

*calculateGmean (c, d)*

**Default Arguments Function**

Defaults arguments are those values which can be given to the function when they are made.

*def name (fname, mname = "Krishnan", lname = "Sennaiyar"):*

*print("Hey,", fname, mname, lname)*

*name("UmaBala")*

Keywords Argument Function: there is no need to follow any order.

Required Arguments: need to pass the arguments should be in correct positional order

Arbitrary Arguments: while creating function add \* this sign before the parameter name while defining the function.

Return statement: the return statement is used to return the value of the expression back to the calling function.

**LIST:**

List Methods:

List.sort() - this method sort the list in ascending order. The original list is updated

List.append() – add the data to the lst

**list = [11,12,13,2,3,4,5,6,7,8,9,0]**

**print(list)**

**list.sort(reverse=True)**

print(list)

**list.reverse()**

print(list)

**print(list.index(13))**

**print(list.count(2))** # this method will show how many times does 2 occurs in the list

**list.copy()** #this will make a copy of the data present in the list

**list.append(17)**

print(list) #to add new data to the list

**list.insert(1, 700)**

print(list) #inserts or swap the value in the list

m = [200, 300, 400]

**list.extend(m)**

print(list) #add the values of another list to our current list perhaps like combining the two list

**TUPLE:**

Tuple is a type of list which can not be changed or altered once it is made or created. Tuples are ordered collection of data items. The tuples are mostly same as list but can’t be changed. It mostly used when we need to keep the constant values

**Tuples Method:**

We can change the tuple into list indirectly. pop method is used to remove the item. You can use all the list method in tuple once you convert the tuple into list. Once the necessary changes are made then again convert the list into tuple

**F String** is used to insert the variable directly into the string rather than using various format function we can use string function and direct add the variable

**Docstring** are the string literals that appears right after the definition of the function, method, class or module. for ex

def square(n):

‘’’Takes in number n, returns the square of the n’’’ *#so this line is docstring but this will be not printed in the output*

Print (n\*\*2)

Square (5)

Print (square.\_\_doc\_\_)

**PEP8** is a document that provides guidelines and best practices on how to write Python code.

**Recurssions** are basically a function, when inside the function you call the exact same function, its called recursion

SETS is collection of well-defined objects. If you want something which doesn’t have any repeated entries. They store multiple items in a single variable. We always use curly brackets for sets. It doesn’t maintain any order.

studentId = {12, 13, 14, 15, 16}

print ("Student ID: ", studentId )

mixedSet = {"Harry", 12, 3, -2, 5.5}

print ("Set of mixed values: ", mixedSet)

emptySet = set() #for empty set use set()

print (emptySet) #function instead of curly brackets

duplicateSets = {2, 3, 4, 2, 5, 4} #python never prints duplicate values

print (duplicateSets) #in sets

ogSet = {11, 12, 13, 14}

print("The OG set: ", ogSet)

ogSet.add(15)

print("The updated sset: ", ogSet) #use add() func to add vakues to the set

itsaSet = {"Mars", "Venus", "Pluto"}

itsaList = ["Hey", "Hello", "Vanakam"]

itsaSet.update(itsaList)

print(itsaSet) #its combine other collection with the set

removeSet = {23, 24, 25, 26, 99}

print ("Set before deletion", removeSet)

removeSet.discard(99)

print("set after deletion: ", removeSet)

planets = {"Mars", "Venus", "Pluto"}

# we use for loop to access every values of the set

for planet in planets:

print(planet)

**Outputs of the above code:**

Student ID: {16, 12, 13, 14, 15}

Set of mixed values: {3, 5.5, 12, -2, 'Harry'}

set()

{2, 3, 4, 5}

The OG set: {11, 12, 13, 14}

The updated sset: {11, 12, 13, 14, 15}

{'Vanakam', 'Pluto', 'Hello', 'Mars', 'Venus', 'Hey'}

Set before deletion {99, 23, 24, 25, 26}

set after deletion: {23, 24, 25, 26}

Venus

Pluto

Mars

**Set Methods:**

union() is used to merge the two sets

update() is used to update the values of the two sets

intersection() means the value which is common or same in the two sets

symmetric\_difference() means the values which is different or repeated only once in each set

difference()

**DICTIONARY**:

dictionary are ordered collection of data items. They store multiple items in single variable. It as combination of key value pair.

A = {1, 2, 3, 4, 5}

B = {5, 6, 7, 8,}

#union operation

print ("Union set : ", A | B)

#built-in method

print ("Union set : ", A.union(B))

#intersection set

print("Intersection set: ", A & B)

print("Intersection set: ", A.intersection(B))

#difference set

print("diffrence: ", A - B)

print("diffrence: ", A.difference(B))

**For Loop with Else**

We can use else keyword with for and while loops. The else loop appear after the body of the loop.

**EXCEPTION HANDLING IN PYTHON**

Exception handling is the process oh handling the unwanted or unexpected error or event occurs when the computer program runs. Need to use **try** and **except** keywords. This is mostly used when the programmer thinks that there might be an error can occur. In **try** the code will try to execute if does not run properly then it will run what is present inside **except** section.

There are different types of errors exception such as MemoryError, IndexError, ValueError etc.

**Finally,** keyword is used when the content is printed or executed even when code has given some error or run successfully.

**Raining Custom errors**: In python we can create or raise error by using raise keyword.

This error is given by the programmer themselves it is not given the python built-in function or methods

For example:

**a = int(input("Enter the number betweenn 3 and 7: "))**

**if (a<3 or a>7):**

**raise ValueError("Value should be between 3 and 7")**

**Shorthand If else:**

This is the shorthand if else which can be used when the condition being tested is simple and short.

Example:

a = 30

b =31

print ("Hello") if a == b else print ("hey") if a > b else print ("Get lost")

**Enumerate function:**

Enumerate function is the built-in function in the python which will iterate or will loops through all the sequence in list, tuple, sets etc. and also get the index and value of each element in the sequence.

Change the start index: by default the enumerate function will start the index at 0 but you can also specify a different starting of the index by passing the argument to the enumerate function.