**DATA ANALYSIS WITH PYTHON**

**1.CONCEPT OVERVIEW-PYTHON**

**1.1.Variable:**

Container to store values

**Ex:**

A=5

**1.2.Print():**

It is a function used to display

**Ex:**

a=5

print("Iam",a,"years old")

**Output:**

Iam 5 years old

**1.3.Operators:**

* **Arithmetic Operators:** Performs mathematical calculation on two operands.**(+,-,\*,/,%,//,\*\*)**

**Ex:** a=5

b=25 **Output:**

print(a+b) 30

print(b//a) 5

print(b\*\*a) 9765625

* **Relational Operators:** These are used for comparing the values. It return either true or false. **(>,<,<=,>=,!=,==)**

**Ex:**

a=6

b=6 **Output:**

print(a==b) True

print(a>b) False

print(a<b) False

print(a<=b) True

print(a>=b) True

print(a!=b) False

* **Logical Operators:** These are used on conditional statements(either true or false). **(and, or, not)**

**Ex:**

a=9

b=25 **Output:**

print((a>b)and(a<b)) **#false and true** False

print((a>b)or(a<b)) **#false or true** True

print(not((a>b)and(a<b)))**#not(false)** True

* **Membership Operators**: These operators allow you to check the presence or absence of a substring within a given string. They return a boolean value - True if the substring is present, and False otherwise.(in, not in)

**Ex:**

a="army" **Output:**

print("a" not in a) False

print("a" in a) True

**1.4.Control-flow-Conditional Statements: if, elif, else if ladder(chained if)**

* **If statement:** It is the most simple decision-making statement. It is used to decide whether a statement will be executed or not.

**Ex:**

a=6 **Output:**

b=12 true

if(a!=b):

print("true")

* **If-else statement:** If the condition provided in the if statement is false, then the else statement will be executed.

**Ex:**

**#check whether given number is positive or negative**

a=int(input("enter a value:"))

if(a>0): **Output:**

print(a, "is positive") enter a value:42

else: 42 is positive

print(a, "is negative")

* **Else-if-ladder**: In ifelse-if ladder statement, if condition is true then the statements defined in the if block will be executed, then the statements defined in the else-if block will be executed, at the last if none of the condition is true then the statements defined in the else block will be executed.

**Ex:**

**#check whether given number is positive or negative**

a=int(input("enter a value:"))

if(a>0): **Output:**

print(a, "is positive") enter a value:0

elif(a==0): 0 is neither positive

print(a, "is neither positive nor negative") nor negative

else:

print(a,"is negative")

**1.5.Control-flow:Looping statements: for, while**

* **for** loop is used for iterating over a sequence (that is either list, tuple, dictionary, set, string).

**Ex:**

n=int(input("enter a number:"))

for i in range(1,6):

a=n\*i

print(a)

**Output:**

enter a number:6

6

12

18

24

30

**#NOTE: range(start, stop, step) returns series of values.**

Default value of start is 0

Default value of step is 1

* **while Loop**is used to execute a block of statements repeatedly until a given condition is satisfied. When the condition becomes false, the line immediately after the loop in the program is executed.

**Ex:**

n=int(input("enter a value:")) **Output:**

a=int(input("enter the no. of steps:")) enter a value:6

i=1 enter the no. of steps:4

while(i<=a): 6

b=n\*i 12

i=i+1 18

print(b) 24

**1.6. Data slicing:**Retrieves values from a specific part from group of part. It is quite similar to range function. This function is also with start: stop: step.

**Ex:**

a="python is easy" **Output:**

print(a[::-1]) **#reverse** ysae si nohtyp

print(a[::2]) **#even positions** pto ses

**1.7.Type conversion:**

changing one datatype to another datatype**(implicit,explicit)**

* **Implicit Conversion:** The Python interpreter automatically converts one data type to another without any user involvement.

**Ex:**

x = 10

print("x is of type:",type(x)) **Output:**

y = 10.6 x is of type: <class 'int'>

print("y is of type:",type(y)) y is of type: <class 'float'>

z = x + y z=20.6

print(z) z is of type: <class 'float'>

print("z is of type:",type(z))

* **Explicit conversion:** In Explicit Type Conversion in Python, the data type is manually changed by the user as per their requirement.

**Ex:**

num\_string = '12'

num\_integer = 23

print("Data type of num\_string before Type Casting:",type(num\_string))

num\_string = int(num\_string)

print("Data type of num\_string after Type Casting:",type(num\_string))

num\_sum = num\_integer + num\_string

print("Sum:",num\_sum)

print("Data type of num\_sum:",type(num\_sum))

**Output:**

Data type of num\_string before Type Casting: <class 'str'>

Data type of num\_string after Type Casting: <class 'int'>

Sum: 35

Data type of num\_sum: <class 'int'>

**1.8.Collections:list,set,tuple,dictionary**

* **list:** Represented in square brackets.

- collection of elements

- heterogenous

- append

- mutable(modifiable)

**Ex:** **Output:**

b = [34,"killer",90.8] 34

for i in b: killer

print(i) 90.8

**List Methods:**

* **append():** adds the new element at the end of the list.

**Ex:**

a=[94,"vasavi",13.06]

a.append(8)

print(a)

**Output:**

[94, 'vasavi', 13.06, 8]

* **insert():** adds the new element at specified position in the list.

**Ex:**

a=[94,"vasavi",13.06,8]

a.insert(2,"hi")

a.insert(3,"hello")

print(a)

**Output:**

[94,'vasavi','hi','hello',13.06,8]

* **extend():** adds all the elements in a list to another specified list. i.e., It joins two lists.

**Ex:**

a=[94,'vasavi','hi','hello',13.06,8]

b=[34,"killer",90.8,45]

a.extend(b)

print(a)

**Output:**

[94,'vasavi','hi','hello',13.06,8,34,'killer',90.8,45]

* **pop():** It is used to delete an element from list. The specified index element is popped.

**Ex:**

a=[94,"vasavi",13.06,8]

a.pop(2)

print(a)

**Output:**[94,’vasavi’,8]

* **remove():**Removes the item at the specified value.

**Ex:**

b=[34,"killer",90.8,45]

b.remove("killer")

print(b)

**Output:**

[34,90.8,45]

* **min():**Returns the min value in the list of numbers.

**Ex:**

c=[1,12,13,23,13]

print(min(c))

**Output:**

1

* **max():**Returns max value in the list of numbers.

**Ex:**

c=[1,12,13,23,13]

print(max(c))

**Output:**

23

**list comprehension**

* iterates
* applied to some function on every element
* conditions **#output iteration or condition output iteration**
* output

**Ex:**

a=[13,23,94,4]

b=[i\*\*2 for i in a] **#output iteration**

print(b)

c=[i\*\*2 for i in a if i>=50] **#condition output iteration**

print(c)

**Output:**

[169,529,8836,16]

[8836]

**Problem-1:**

The salaries of 5 employees in a company is taken in a list. the tax is 10% if the salary is less than or equal to 50000 or it is 15%

(creating a list with salaries

[67000,45000,89000,34000,50000]

**#without using list comprehension**

**list\_name=[(body of if) (if condition) else (body of else) iteration])**

l1=[67000,45000,89000,34000,50000]

tax=[]

for i in l1:

if i<=50000:

t=i\*0.1

tax.append(t)

else:

t=i\*0.15

tax.append(t)

print(tax)

**Output:**

[10050.0, 4500.0, 13350.0, 3400.0, 5000.0]

**#using list comprehension**

**#list\_name=[(body of if) if (condition) else (body of else) iteration]**

l1=[67000,45000,89000,34000,50000]

tax=[i\*0.1 if i<=50000 else i\*0.15 for i in l1]

print(tax)

**Output:**

[10050.0, 4500.0, 13350.0, 3400.0, 5000.0]