Datascience & AI Masters 2025

23/05/2025

→ Google Collab.: → libraries are to be installed.

→ you need to upload it into your cloud Directory.

* Introduction to python: ① opensource general purpose prog language ② OOP ③ programming lang but as well as Scripting language

→ Everyone uses python due to its applications.

* Features: ① Easy to learn and use ② Interpreted language ③ libraries ④ Large community support ⑤ Extensible.

* More user friendly | Stability
More Applications | speed.

Basics of Python

** Variables and Keywords.

① Variables: A temporary storage space where you can keep changing values.

Example: a = xyz ; age = 25

→ we dont have to declare the datatype of the variable.

② Keywords: Special reserved words that convey special meaning to interpreter (or) compiler.

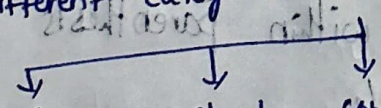
→ cannot creak keywords names as variables

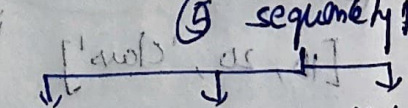
Ex: def, class, else, ---

* Variable name cannot start with a number.

⑩ Data type Operators:

→ 5 Different categories: ① Numeric ② Dictionaries ③ Boolean ④ set ⑤ sequence types





→ Variables hold values of Diff data types

→ using type() we can check type of variable used.

Operators and Operands :-

→ Operand: A quantity to which operator is applied.
→ Operator: -, +, /, *, **

name = input("Enter your name");

→ Typecasting: changing the datatype of a variable

* There might be dataloss due to typecasting.

* converting a string in float characters (or) values cant be converted to integer directly

Instead :- float → int.

Basic Operations : Add, Sub, Mul, Exponential, Divided.

a=10 b=2 c=a+b =12	a=2 b=2 c=a**b =4	a/b = 5.0 a//b = 5. → Floor Division → 5.something = 5 Ceiling → 5.something = 6.

a%b → Remainder = 0

Follows BODMAS

((4*5)-9+0.8)
(20)-9.8)
= 11.8

Python Data Structures:

→ containers that organise data depends on the type of the Data.

* **Lists** Vs Tuples.

Lists	Tuples
① values in lists are called as Elements (or) items. [10, 20, 'class']	① Similar to lists, having sequence of values of any type and enclosed within parenthesis
② Nested list	② Immutable
③ Mutable	③ fixed length tup-1 = ('a', 1, 'df')
④ variable length	④ () → Declared in round bracket
⑤ Accessed similarly like arrays	
⑥ [] → Declared in square Brackets	

* List:

List1 = ["preethi", "KK", 24, 01]

Nested list:

list2 = [["preethi", 06], "KK", 24, 01]

list2[0][0]

Output: 'Preeth"

Concatenation:

list 1 = [1,2,3,4] ⇒ list 1 + list 2 = [1,2,3,4,5,6,7,8]

list 2 = [5, 6, 7, 8]

* extend in lists

list1.extend([9,10])

list1 = [1,2,3,4,9,10]

* append in lists

list1.append([9,10])

list1 = [1,2,3,4,[9,10]]

* del. function

del list1[2]

list1 = [1,2,4]

* pop() → removes last element by default → index

list1.pop(0)

~~list1 = [2,3,4]~~ list1 = [2,3,4]

* remove → values ⇒ list1.remove(3)

list1 = [1,2,4]

* sorting → sort function

A = [8, 6, 1, 24]

A.sort()

⇒ A = [1,6,8,24]

A.sort(reverse = True)

⇒ A = [24,8,6,1]

* Difference between sort & sorted.

→ sort: Does not Assign the sorted values to Another variable

→ sorted helps us to come over this limitation.

* shallow copy

A = ["preethi", "Mom", "KK"]

B = A

A.pop() ⇒ A = ["preethi", "Mom"]

B = ["preethi", "Mom"]

* Reflects the changes.

* A = ["orange", "kiwi", "poth"]
  B = A[0:3]
  Print(B) ⇒ ["orange", "kiwi", "poth"]

⇒ A.pop()
  A = ["orange", "kiwi"]
  B = ["orange", "kiwi", "poth"]

* String split operator

str1 = str.split('-')
  print(str1)

Tuples

1. if the content is fixed then we should use Tuples.
   t = (1, 2, 3, 4)   can also be declared t = 1, 2, 3, 4
                       print(t) = (1, 2, 3, 4)
   type(t) ⇒ tuple.

* t = ('hello', 5, 5.5)

→ A list can be part of a tuple.
  t1 = (1, 2, 3, [4, 5])

* t1 = 1.2,
  type = tuple
* t1 = 1.2
  type(t1) = float

* t = (1, 2, 3, 4)
  t[1:4]
  t = (1, 2, 3)

* t = (1, 2, 3, 4)
  t = t[0:2]
  t = (1, 2)

* concatenation is as same as the list for tuple.

* min, max, sum

* if the tuple attributes are not same then you cant perform above
  l1 = (1, 2, 3, 'hello') → Error
  l1 = (1, 2, 3)
  ⇒ sum(l1) = 6

→ Accessing is as same as a list.

⑧ Sorting

```
t = (2, 3, 8, 5, 4)
new_var = sorted(t)
Print(new_var) = [2, 3, 4, 5, 8]
                 type = list
```

⇒ new_var = tuple(new_var)
↳ new object in memory

→ if we can duplicate or create an object then we can. by converting into list and then into tuple.

Question x = ('2') — Str

x = 1,2, → Tuple

x = ((1, 2, 3), 4, 5) — Tuple

x = (Hello, '4', '5') → Hello not def → Error