**Python Notes**

**Module 1:**

**Keywords:**

Keywords are pre-defined and immutable, In python3.7 has 35 keywords are present and python 2 has 31 keywords present.

Checking the keywords in python.

Syntax:

Import keyword

Keyword.kwlist

**Identifiers:**

Identifiers are user defined words.it has some rules

* First letter is character (or) \_
* It does not allows the first letter is number.
* It has case sensitive.
* Special characters are not allowed.
* Small case is allow

Eg: a, name,\_pwd,\_\_user, mobile2.

Not allows:--2mobiles,@hari,user$name,AGE

**Variables**:

It stores the data.it has 3 types

* Name🡪identifier
* Value🡪data
* Address🡪??

Eg:

a=10 3.7.2 2.7

Name🡪a type (a) type (a)

Value🡪10 <class ‘int’> <type ‘int’>

Address🡪??

**Data types:**

Data types has two types

* Fundamental datatypes
* Collection datatypes

**Fundamental datatypes:**

**1.** Numbers

Int

Float

Complex

Boolean

2. Strings

**Collection datatypes:**

1. Lists

2. Tuples

3. Dictionaries

4. Sets

5. Frozen sets

**Numbers**:

Int 🡪it stores all whole numbers

Float🡪it stores all decimal numbers

Complex🡪 it stores the a+ib 🡪 c+dj

**Operations**:

Arithmetic

Comparison

Logical

Bitwise

These all are only numbers.

Except numbers:

Membership

Identity

These all are except numbers.

**Arithmetic operations:**

1. **+ (**add**)**
2. **– (**sub**)**
3. \* (mul)
4. %(rem or mod)
5. / ( div)
6. //(floor div)
7. \*\*(Exponent /power)

**Comparison operations:**

1. ==(is equal)
2. !=(not equal)
3. >(greater than)
4. <(less than)
5. >=(greater than are equal)
6. <=(less than are equal)

It returns only true or false.

**Logical operations:**

1. Add
2. Or
3. Not

It returns only true are false.

**Bitwise operations:**

1. & (Bitwise and)
2. | (Bitwise or)
3. ^ (Bitwise xor)
4. ~ (bitwise not)
5. >>(right swift)
6. <<(left swift)

**Strings**:

All combinational of characters is consider as a strings.

* Characters
* Combinations of characters
* Alpha numeric

**Syntax:**

**<string name> =”<string value>”**

**Features of strings:**

* Strings can be indexed.
* Strings can be sliced
* Strings can be concatenated
* Strings can be iterated
* Strings can be immutable

Ex:

Tech=”python”

p y t h o n

0 1 2 3 4 5 forward indices.

-6 -5 -4 -3 -2 -1 reverse indices.

Indexing is used single characters in a strings.

**Slicing**:

Slicing is used to extract to string.

Ex:

tech="python and machine learning I am harikittu btech cse branch”

Print (tech [5:15])

**Output**: n and mach

**Concatenation**:

Type casting: Type casting is chaining the one data type to another data type is called type casting.

str()🡪integer to string.

int()🡪string to integer

Concatenation example:

Tech=”python”

Version=3

Print (tech+str(version))

Output: python3

**Functions**:

Functions are two types.

1. Attribute fetching(.)
2. Parameterised (<something>)

**String functions:**

There are 3 types of string functions.

1. Case based.
2. Checking.
3. Manipulative.

**Case based function:**

**.lower ():** all characters are converted to lower case.

Ex. Tech=”HariKittU”

Print(tech.lower())

Output: harikittu

.**upper ():** all characters are converted to upper case.

Ex. Tech=”HariKittU”

Print(tech.upper())

Output: hARIkITTu

**.swapcase ():**

Ex: tech="Python and Machine Learning"

Print (tech.swapcase ())

Output: pYTHON AND mACHINE lEARNING

**.capitalize ():** Entire string the first letter is capitalize

Ex. tech="Python and Machine Learning"

print (tech.capitalize())

output: Python and machine learning

**.title():** entire string the words of first character is upper

**Checking functions:** it returns true are false only

* **startswith ():** starting character checking.

Ex: tech=”Python and machine learning”

Print(tech.startswith (“P”))

Output: true

* **endswith():** ending character checking

Ex: tech=”Python and machine learning”

Print(tech.endswith (“g”))

Output: true

* **isdigit():** it checks the digit or not

Ex: tech=”100”

print(tech.isdigit())

output: true

**Manipulative functions:**

* **len():** it returns the length of string.

Ex : tech="python and Machine Learning"

print(len(tech))

Output: 27

* **count():** How many times thee character is placed in the string.

**Ex:** tech="python and Machine Learning"

Print (tech.count("p"))

Output: 1

* **index():** it returns the index value of the character past for its **first occurrence**.

if a sub string is pass index of **first char of the sub string** is returns.

Ex: tech="python and Machine Learning"

print(tech.index("a"))

output: 7

Ex2: tech="python and Machine Learning"

print (tech.index(" ",7))

Output: 10

* **replace():** replace the old string to new string.

**Ex:** tech="python and Machine Learning"

print(tech.replace("python","java"))

output: java and Machine Learning

* **split():** it is used to split the string and returns the collection.

**Ex:**

tech="python and Machine Learning"

print(tech.split("a"))

output: ['python ', 'nd M', 'chine Le', 'rning']

* **join():** to join the string.

**Ex:** tech="python and Machine Learning"

words=tech.split(" ")

print("@".join(words))

output: python@and@Machine@Learning

* strip(): it removes the white spaces.
* lstrip():it removes the left sides white spaces remove.
* rstrip():it removes the right side white spaces remove.

**Membership operators:** member ship operators returns the true or false

* **in():** the character is present or not in a string

**Ex:** tech="python and Machine Learning"

print("an" in tech)

output: true

* **not in():** does not present in a string or not.

**Ex:** tech="python and Machine Learning"

print("is" not in tech**)**

**output:** true

**Input functions in python:**

They are two types of input functions:

1. Keyboard inputs.
2. CCI inputs

Keyboard inputs:

To take the inputs by keyword.

Ex: a=int(input(“enter number1:”))

b=int(input(“enter number2:”))

print(a+b)

output:

enter number1:100

enter number2:200

300

**Condition statements:**

It is used a decision to make for every condition the statement is comparison the condition defiantly returns a true or false

Python has 3 kinds of conditional statements:

* If
* If else
* If elif elif ….else

**if condition:**

**Syntax:**

if <condition>:

<statements1>

**if else condition:**

**Syntax:**

if <condition>:

<statement1>

else:

<statement2>

**elif condition:**

**syntax:**

if <condition>:

<statement1>

elif<condition>:

<statement2>

else:

<statement3>

* Condition statements will exits first true block.

**Loops**:

Loops are used repetitions both finite or infinite repetitions.

We have two types of lops

1. Finite loops
2. Infinite loops

Python has for and while loops to follow as

Finite loops are “for and while loops”

Infinite loops are “while loops”

* For loops can be used for three datatypes

1. Numbers
2. Strings
3. Collections

A loops is consists of four types

1. Initialization
2. Limit(or)condition
3. Increment(or)decrement
4. Statements

**Initialization**: it is starting point of the loop

**Limit or condition**: limit is a ending point of loop

**Increment or decrement:** to increment or decrement of the condition.

**Statements**: executable code for programming.

* **range**() : it can only used for for loop it is an inerrable function it is only work for numbers

range function is written in three ways

1. range(<end>)

Ex: range(10) start=10,end=9 step=+1

1. range(<start>,<end>)

Ex: range(1,10)start=1,end=9,step=+1

3.range(<start>,<end>,<step>)

Ex: range(1,10,+2)start=1,end=9,step=+2

Syntax:

For <dummy variable>in range():

<statement>

Ex:🡪 for i in range(10):

Print (i)

**While**: it can be used finite and infinite iterations.

1. Initialization
2. Condition
3. Increment or decrement

It should be returns explicitly and it can used for only numbers.

Syntax:

< Initialization>

While<condition>

<statement>

<Inc/dec>

Ex: a=1

While a<10:

Print(a)

a=a+1

output: 123456789

ord():🡪 it convert character to asci format.

chr():🡪it convert asci to character format.

**Control statements in python :** they are 3 types of control statements.

1. Pass
2. Break
3. Continue

**Pass:** it is used to execute the unimplemented blocks.it can be used in conditional statements and loops and functions and class.

Ex: a=1

if a>5:

Pass

else:

Print(“a lt 5”)

**Break**: it is used to exit the loop at a condition when the condition is true.

Ex: for I in range(10):

If i==5:

break

print(i)

output:012345

* The position of the print also determine

**Continue**: it exits the loop at a condition if the condition is true and it returns back to the loop after the condition is executed.

Ex: for i in range(10):

If i==5:  
 continue:

Print(i)

**Module 2**

**Collections**:

Collections are derived data types which stores multiple data vales

They are 5 collections in python:

1. Lists.
2. Tuples
3. Dictionaries
4. Sets
5. Frozen sets

Lists:

* Lists are stack
* Lists can be indexed
* Lists can be sliced
* Lists can be concatenated
* Lists can be iterated
* They can form <class list>
* Lists are heterogeneous
* Lists are of order 1XN
* They can store repetitions
* Lists are mutable datatypes

Syntax:

<list name>=[<ele>,<ele1>,<ele2>…….<ele n>]

Stack and queue:

Stack: last in first out(LIFO)

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

1 2 10 last in first out will go first

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

First in last out

Queue: first in first out

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

Entry exits

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

* We can modify and replace the list at any type
* We can modify and replace the multiple values in a list

**List operations:**

**del():**

Deleting elements in a list

Syntax: del<list name>[<index>]

**append():**

it is used to add a element in a list at the last position we can only append at only one element

syntax:

<list name>.append(<element>)

**Insert():**

It is adds an element at the specified index.

Syntax:

<list name>.insert(<index>,<element>)

**Pop():**

It removes last element from the list

Syntax:

<list name>.pop()

del.tech[tech.index(“pip”)]

* Particular element can be remove we don’t know the index value

**remove():**

it removes the specified element in a list

syntax:

<list name>.remove(<element>)

**Extend():**

Syntax:

<list name1>.extend<list name2>

L1-------- 1000 L1--------1000

L2---------1000 L2--------1000

L3----------L1+L2 L1.extend (L2)

40000 3000

Memory difference

**Sort():**

It is used to sort an entire homogenies list in ascending order.

Syntax:

<list name>.sort()

Descending order:

<list name>.sort(reverse=TRUE)

(OR)

After sort the list we can reverse the list

tech.sort ()

Print (tech)

tech.reverse ()

Print (tech)