**Table of Content**

**1. Install + First Program**

**2. Variable and Types**

**3. Arithmetic Operators**

**4.** **Python Comparison operators, Logical operators, identity operators, membership operators**

**5. Lists and tuples**

**6. Sets and Dictionaries**

**7. String formatting and methods (print with % and methods i.e. strip, upper, lower, …)**

**8. Conditions (if, elif, else)**

**9. Loops (For loops)**

**10. Loops (while,** **"break" and "continue" statements)**

**11. Functions**

**12. Python Dates**

**13. Classes & Objects**

**14. Exception Handling ( try, log except e as exception)**

**15. Pip install**

**16. Modules and Packages (Importing functions from other py files)**

**1 - Intro Video**

-Hello everyone, and welcome to the very first video in this python for beginners tutorial series.

This series is designed for beginner python programmers with a focus on data science, the aim of this series is establish a solid python programming base from which you will be able to build upon in the future.

-With that said, in this video we will be installing python 3, setting up the system environment variables,

and then creating our very first python program.

-Okay so first things first we need to install python, and we will do that by going to google

typing in download python

-open cmd type python

-open integrated development environment or integrated development and learning environment

-create first program

Print(‘Hello World’) (print create a newline character unlike C).

-test via cmd line

**2. Variable and Types**

welcome to part 2 of this python tutorial series, in this video we are going to be looking at variables and variable types.

To create a variable, you do that by picking a variable name lets pick var1 and then using the equal sign to assign a value to the variable, let’s go with 5

Var1 = 5

Var1

Unlike some other programming languages in python you do not need to declare a variable type when creating the variable.

**Numbers**

-Python supports two types of numbers - integers and floating point numbers.

**Integers**

myint = 7

print(myint)

type(myint)

**Floating point numbers**

myfloat = 7.0

print(myfloat)

myfloat = float(7)

print(myfloat)

type(myfloat)

**Strings**

mystring = 'hello'

print(mystring)

mystring = "hello"

print(mystring)

The difference between the two is that using double quotes makes it easy to include apostrophes (whereas these would terminate the string if using single quotes)

type(mystring)

mystring = "Don't worry about apostrophes"

print(mystring)

Simple operators can be executed on numbers and strings:

one = 1

two = 2

three = one + two

print(three)

hello = "hello"

world = "world"

helloworld = hello + " " + world

print(helloworld)

Assignments can be done on more than one variable "simultaneously" on the same line like this

a, b = 3, 4

print(a,b)

Mixing operators between numbers and strings is not supported:

# This will not work!

one = 1

two = 2

hello = "hello"

print(one + two + hello)

## **Video 3 – Arithmetic Operators**

Addition: +

X= 10, Y=5 z = X+ Y, z

Subtraction: -

X= 10, Y=5 z = X - Y, z

Multiplication: \*

X= 10, Y=5 z = X \* Y, z

Division: /

X= 10, Y=5 z = X / Y, z

Floor Division: //

X= 10, Y=5 z = X // Y, z

Modulus/Remainder: %

Z = 16 % 3, z

Z = 27 % 5, z

Exponentiation: \*\*

Z = 2 \*\* 3, z

## **Video 4 - Python Comparison operators, Logical operators, identity operators, membership operators**

**Comparison Operators**

Equal: ==

Not Equal: !=

Greater than: >

Less than: <

Greater than or equal to: >=

Less than or equal to: <=

**Logical Operators**

And

Or

Not

**Identity Operators**

Is

Is not

**Membership Operators**

In

Not in

## **Video 5 – Lists and tuples,**

List is a collection which is ordered and changeable. Allows duplicate members.

Tuple is a collection which is ordered and unchangeable. Allows duplicate members.

Welcome to the third video in this python tutorial series. In this video we will be looking at Lists.

**Lists**

Lists are very similar to arrays if you familiar with other languages.

They can contain any type of variable, and they can contain as many variables as you wish.

Mylist = [‘hello’,5,’python’,’java’,6]

mylist = []

mylist.append(1)

mylist.append(2)

mylist.append(3)

print(mylist[0])

print(mylist[1])

print(mylist[2])

for x in mylist:

print(x)

Accessing an index which does not exist generates an exception (an error)

mylist = [1,2,3]

print(mylist[10])

numbers = []

strings = []

names = ["John", "Eric", 5,5,"Jessica"]

**Changing values in a list**

mylist= [1,2,3,4,5,6]

mylist[1] = 333

mylist

**Removing items from a list 3 ways**

Remove, pop, del

remove removes the first matching value, not a specific index:

a = [0, 2, 3, 2]

a.remove(2)

a

del removes the item at a specific index:

a = [3, 2, 2, 1]

del a[1]

a

pop removes the item at a specific index and returns it.

a = [4, 3, 5]

a.pop(1)

**get length of a list**

mylist= [‘soccer’,’hockey’,’tennis’,’basketball’]

len(mylist)

**Tuples**

A tuple is a collection which is ordered and unchangeable. In Python tuples are written with round brackets.

thistuple = ("apple", "banana", "cherry")

print(thistuple)

Once a tuple is created, you cannot change its values, different from a list. Tuples are unchangeable.

thistuple = ("apple", "banana", "cherry")

thistuple[1] = "blackcurrant"

print(thistuple)

Lists in lists

Mylist = [‘DataScience’,5,7.0,[1,2,3,4,5]]

Mylist[3]

Mylist[3][0]

**Video 6 – Sets and Dictionaries**

Dictionary is a collection which is unordered, changeable and indexed. No duplicate members.

**Dictionaries**

A dictionary is a data type similar to arrays, but works with keys and values instead of indexes. Each value stored in a dictionary can be accessed using a key, which is any type of object (a string, a number, a list, etc.) instead of using its index to address it.

phonebook = {}

phonebook["John"] = 938477566

phonebook["Jack"] = 938377264

phonebook["Jill"] = 947662781

print(phonebook)

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}

access items or keys

thisdict.items()

thisdict.keys()

get a value by using a key

x = thisdict["model"]

x = thisdict.get("model")

delete an item

del thisdict ["brand"]

print(thisdict)

thisdict.pop("model")

Change an item

thisdict["year"] = 2018

print(thisdict)

Dictionary length

print(len(thisdict))

Clear the dictionary

thisdict.clear()

print(thisdict)

Dictionary Constructor

thisdict = dict(brand="Ford", model="Mustang", year=1964)

print(thisdict)

**Sets**

Set is a collection which is unordered and unindexed. No duplicate members.

thisset = {"apple", "banana", "cherry"}

print(thisset)

add to the set

thisset.add("orange")

Add multiple items to a set, using the update() method:

thisset.update(["orange", "mango", "grapes"])

print(thisset)

Get the number of items in a set:

print(len(thisset))

remove an item from the set

thisset.remove("banana")

thisset.discard("banana")

thisset.pop()

del thisset

print(thisset)

Clear the entire set

thisset.clear()

print(thisset)

The set() Constructor

thisset = set(("apple", "banana", "cherry"))

print(thisset)

Set intersections (Return a set that contains the items that exist in both set x, and set y)

x = {"apple", "banana", "cherry"}

y = {"google", "microsoft", "apple"}

z = x.intersection(y)

print(z)

set unions

x = {"apple", "banana", "cherry"}  
y = {"google", "microsoft", "apple"}  
  
z = x.union(y)   
  
print(z)

**Video 7 – String formatting and methods**