Variables:

X=2

Y=3x + 4 ==10

What is variable

A variable is nothing but a reserved memory location to store values. In other workds a variable in a program gives data the computer to work on.

How to declare and use a varible

* Dynamically typed lang
* No need to declare variables before using them
* Re-declare a variable

X=10

print(x)

X=15

print(x)

You can change values directly.

X=10

* Delete a variable

x=10

del (x)

print(x) ######## error

Rules:

**1.It contains letters, numbers and underscore.**

java\_version = 1.8

#  x$

#   $s  not valid

**2.It should not a keyword.**

#print=2

# print(print)

**3.Can’t contain spaces.**

# my\_ name = "narendra"  #dont give spaces

**4.It should not start with a number**

# 4x56=22.6  -- Do not start with number

**5.Case-sensitive**

x=10

print(x) # 10 case sensitive

X=11

print(X) # 11

\_x=10 # can start with \_ not with number

Data Types/ Variabel Data type in Python:

* Every value in Python has a data type
* Since everything is an object in Python programming, data types are

Actually classes and variables are instance (object) of that class

Basic Data types:

* Numbers (int, float and complex)
* x=3;y=4.6;z=3+4j
* Strings
* my\_name="narendra"
* Boolean

my\_value=True

my\_new\_value=False

int,float,complex

str

bool

Note: Any data type can be converted into Boolean

bool(any\_data\_type) == True/False

bool(empty)=False

bool(non-empty)=False

1. Int to string

X=56

Y=str(X)

Type(y) 🡪 str

1. String to bool

z=bool(x)

1. Int to bool

P = 0

Q= bool(P)

1. String to int

my\_name="nnr"

int(my\_name) 🡪 -🡪 not possible

my\_name=”48”

int(my\_name) 🡪 Possible

A screenshot of a computer

AI-generated content may be incorrect.

print(bool(empty)) # False

print(bool(0)) # False

print(bool(0)) # False

print(bool("")) # False

print(bool([])) # False # List

print(bool({})) # False # dict

print(type(({}))) # dict

print(bool(())) # False #

print(bool(None)) # False

print(bool(0.0)) # False

print(bool(0j)) # False

print(bool(0.0+0j)) # False

print(type([])) # List

print(type(({}))) # dict

print(type(())) # tuple

Working with multiple variables and strings in print:

x=3 ; y=5.7 ; lang="python"

print(f'x values is {x} \ny :{y} \nlang is {lang}')

      #x values is 3

      #y :5.7

      #lang is python

my\_req\_output=f'x values is {x} and y :{y} and lang is {lang}'

print(my\_req\_output)

# x values is 3 and y :5.7 and lang is python

Inputs and Outputs:

Eval

a=eval(input("Enter a number1: "))

b=eval(input("Enter number2: "))

print(f'a type: {type(a)},b type: {type(b)}')

result=a+b

print(f'result is {result}')

Case 1:

Enter a number1: 1

Enter number2: 3

a type: <class 'int'>,b type: <class 'int'>

result is 4

case 2:

Enter a number1: "hello"

Enter number2: "Hi" (must provided the quotation using eval)

a type: <class 'str'>,b type: <class 'str'>

result is helloHi

case 3:

Enter a number1: 5.7

Enter number2: 4.6

a type: <class 'float'>,b type: <class 'float'>

result is 10.3