Python Basics

Introduction

In Simple words Programming language is a way to communicate with computer and devices.

Why Python ?

- Python is easy to learn and understand because of easy syntaxs.
- Widely used and has community support across the internet (Stackoverflow, Discord Python server, Quora space for python enthusiast)
- Used widely mostly for data science , machine learning as well as general programming
- Lots of resources on internet.

Setting Up the environment

We will be using google collab for this project

Setting up the environment can be done in one of the following ways :-

1. Download python from microsoft windows store [recommended] or from python official web

   = = = = = O R = = = = = =

2. Go to https://replit.com/ click on start coding login or create an account. Click on the python logo below the create button name the file click create repl.

   = = = = = O R = = = = = =

3. Use google collab or any other online editor.

Code Reference [Syntax]

Try this code to get started .

```
# anything that starts with a hashtag is a comment
# It is written for progrmmer to describe code
# this output nothing special

# To display anything on the screen use print() function
# more on function later
print("Hello! World")

# similar to mathematics you can perform calculation
# numbers can be any real number
x = 7
y = 2
print( x + y ) #addition
print( x - y ) #subtraction
print( x * y ) #multiplication
print( x / y ) #divison
print ( x ** y ) # x to the power y
print( x // y ) #floor division ; it has only integer part
print( x % y ) # remainder

# Strings is the terminology used to refer text in programming
# it can be a word , set of word , sentences etc
a = "Hello "
b = "everyone "
print( a + b ) #add both of them as one
print( a * 2 ) #print this text multiple time
```

Variables

Computer stores data (i.e number , character etc.) in **memory locations** . Variables are the storage location where data is stored. It stores value that **changes** as per the use. *It is like a box where you can store a fixed amount of data , but data may vary.*

Literals are the values or data that we assign to the variables. Pythons variables can only contains Alpha numeric character and underscore but it cannot start with a numeric character.

> Python uses dynamic typic i.e. the type of the variable changes as we assign different kind of value. To see the type of any variable use `type()` function

```
a = 10                #integer
print(type(a))
b = 10.0              #float
print(type(b))
c="So"                #string
print(type(c))
```

you can also delete a variable by using `del` keyword.

```
a = 10
del a              # a is deleted now
```

Variables Assignment

python support multiple variable assignments.

```
x , y = 2 , 4 #this is a valid variable
x , y = y , x #this interchanged the values
```

Operators

Operators are used in python to perform calculations and build expression.

Python operators are of three types

1. Arithematic Operators (+ , - , / , * , // , ** , %)
2. Relational Operators (> , < , == , >=, <= , !=)
3. Logical Operators (and , or , not , XOR)

```
# Relational operator Example
#Less than
print( 4 < 5 ) #True
print( 5 < 4 ) #False
print( 4 < 4 ) #False

#Less than or equal to
print( 4 <= 5 ) #True
print( 5 <= 4 ) #False
print( 4 <= 4 ) #True

# Greater than
print( 5 > 4 ) #True
print( 4 > 5 ) #False
```

```
print( 4 > 4 ) #False

# Greater than or equal to
print( 5 >= 4 ) #True
print( 4 >= 5 ) #False
print( 4 >= 4 ) #True

# Equal to
print( 4 == 4 ) #True
print( 4 == 5 ) #False

# Not equal to
print( 4 != 5 ) #True
print( 4 != 4 ) #False
```

```
# Logical Operators Example
# Logical AND
print( True and True ) #True
print( True and False ) #False
print( False and True ) #False
print( False and False ) #False

# Logical OR
print( True or True ) #True
print( True or False ) #True
print( False or True ) #True
print( False or False ) #False

# Logical NOT
print( not True ) #False
print( not False ) #True

# Logical XOR
print( True ^ True ) #False
print( True ^ False ) #True
print( False ^ True ) #True
print( False ^ False ) #False
```

Chanining operator : when you use multiple relational operator in your code it is called chaining operator . like `0 < x <  5`

Arithematic Shorthand Operators (+= , -= , *=, /=) is used to perform arithematic operation as well as assign value.

```
x = 15
x  = x + 1          # increment x by 1
x += 1              # increment x by 1
```

In operator is used to check if some value exist in the given variable or not.

```
a = "ab is a ab coder"
print("coder" in a) #returns true
print("coders" in a) #return false
print("Coder" in a) #return false
```

Input

input() is used to take user input on python consoles . It is a mode of interaction between program and user on console based systems.

```
print("Enter your name to get started")
name = input()
print("Hello! " + name)
```

Input ( ) function takes a parameter that outputs on the console as instruction to user , similar to print.

```
name = input("Enter your name to get started")
print("Hello! " + name)
```

Data Types

Like numbers in mathematics, variables in programming can be of different type . These are some of the basic types used in programming.

1. Strings : Text , words , sentences , phrases , paragraphs or anything that is a combination of character is a string. A string is usually written with a pair of inverted commas (all called quotations)
2. Integer : These are similar to integers in mathematics.
3. floats or doubles : these are used for decimal number that is number with fractional part . E.g. 1.2

To change a data type to other , functions are used in python

To change a string to integer use int( ) function.

```
a = "15"
b = "16"
print( a + b ) #will print 1516 as output

c = int( a )
d = int ( b )
print( c + d ) #will print 31 as output
```

String

```
a = "I am String1" #declare variable with name a and type string
b = "I am String2" #declare variable with name b and type string
print(a + " " + b) #this is concatenation or addition of string
print(a * 3) #prints a for three times [string replication]
print(a[0]) #1st letter of string a
print(a[1]) #2nd letter of string a
print(a[-1]) #first character from last

# string slicing
print(a[1:5]) #print 2nd to 5th characters in string (index 5 excluded)
print(a[:5]) #print first 5 characters of string a
print(a[5:]) #print all characters after 5th character\

#strings can also be compared
print("abc" > "abd") #false. It works like index of english dictionary
```

> Note : Index in all modern programming language starts with 0 and not 1. so zero is the first element always.

for multiline strings use three quotes

```
a = ''' Python
is cool,
easy
and fun'''
```

for multiline comments also three quotes can be used

Escape characters are used to print special characters in a string

\t —> tab

\n —> break line

```
a = " Hey \" Python \" "      # output : Hey " Python "
b = "hey \t python"           # add more space in between
```

Import

using some built infunction (or say predefined program) requires the use of library , library is like a collection of code snippet.

import keyword followed by name of library is used to import a library.

```
# here the name of library is random
# random is a python library, used for generating random numbers
import random
```

Practise program 1

make a program to generate random integer number when someone inputs a number between some range a and some range b for any a and b greater than 0.

Hint : [random.random() generate a random number between o and 1]

```
import random
a = int(input("Enter numbers lower limit"))
b = int(input("Enter numbers higher limit"))
# this lines uses function random from random library
# to generate a random number
c = random.random()
d = b - a  # here d is the range of number(i.e upper - lower limit )
e = int( a + d * c)
print(e)
```

Strings Methods

Method	Description	Code x = 'pytHoN sTrIng mEthOdS'	Output
lower()	Converts a string into lower case	print(x.lower())	python string methods
upper()	Converts a string into upper case	print(x.upper())	PYTHON STRING METHODS
capitalize()	Converts the first character to upper case	print(x.capitalize())	Python string methods
title()	Converts the first character of each word to upper case	print(x.title())	Python String Methods
swapcase()	Swaps cases, lower case becomes upper case and vice versa	print(x.swapcase())	PYThOn StRiNG MeTHoDs

islower()	Returns True if all characters in the string are lower case	x = 'python' print(x.islower())	True
		x = 'Python' print(x.islower())	False
isupper()	Returns True if all characters in the string are upper case	x = 'PYTHON' print(x.isupper())	True
		x = 'PYTHoN' print(x.isupper())	False
istitle()	Returns True if the string follows the rules of a title	x = 'Pyhton String Methods' print(x.istitle())	True
		x = 'Pyhton string methods' print(x.istitle())	False

isdigit()	Returns True if all characters in the string are digits	x = '123' print(x.isdigit())	True
		x = '123abc' print(x.isdigit())	False
isalpha()	Returns True if all characters in the string are in alphabets	x = 'abc' print(x.isalpha())	True
		x = 'abc123' print(x.isalpha())	False
isalnum()	Returns True if all characters in the string are alpha-numeric	x = 'abc123' print(x.isalnum())	True
		x = 'abc123@*#' print(x.isalnum())	False

Method	Description	Code x = '-----Python-----'	Output
strip()	Returns a trimmed version of the string	print(x.strip('-'))	Python
lstrip()	Returns a left trim version of the string	print(x.lstrip('-'))	Python-----
rstrip()	Returns a right trim version of the string	print(x.rstrip('-'))	-----Python

Method	Description	Code x = 'Python'	Output
startswith()	Returns True if the string starts with the specified value	print(x.startswith('P'))	True
		print(x.startswith('p'))	False
endswith()	Returns True if the string ends with the specified value	print(x.endswith('n'))	True
		print(x.endswith('N'))	False

Method	Description	Code x = 'Python String Methods'	Output
count()	Returns the number of times a specified value occurs in a string	print(x.count('t'))	3
		print(x.count('s'))	1
index()	Searches the string for a specified value and returns the position of where it was found	print(x.index('t'))	2
		print(x.index('s'))	20
replace()	Returns a string where a specified value is replaced with a specified value	x = x.replace('S', 's') x = x.replace('M', 'm') print(x)	Python string methods

```
# exaple code

a = "heYY i am A pYthOn proGrammEr"

print(a, " ---> " , a.upper())
print(a, " ---> " , a.lower())
print(a, " ---> " , a.capitalize())
print(a, " ---> " , a.title())
print(a, " ---> " , a.swapcase())

b = "Hehe123@"
print("a is lower? ---> ", a.islower())
print("b is lower? ---> ", a.islower())
print("a is upper? ---> ", a.isupper())
print("b is upper? ---> ", a.isupper())

# test all the methods in a similar way
```

Conditions in python

The program in conditions only execute if the condition matches. For e.g.

```
# check if a number is even or odd
```

```
a = int( input("Enter a number: ") )
if(a % 2 == 0):
  print("even")
else:
  print("odd")
```

```
# you have a digital form where user enters his name
# to make sure it’s a valid name check if the number of letters
# are more than 2.

a = input(" Enter you name: ")
if(len(a) < 3):
  print(" Please enter your full name ")
else:
  print("welcome ! " , a )
```

If-else condition can only check for a single condition what if we have multiple cases . For multiple cases you can use if-elif - else .

```
# you have a digital form where user enters his first name and
# last name both must be greater than 2

#approach 1

a = input(" Enter you first name: ")
b = input(" Enter you last name: ")

if(len(a) < 3):
  print("no. of character in first name must be greater than 3")
elif(len(b) < 3):
  print("no. of character in last name must be greater than 3")
else:
  print("welcome ! " ,a)

#approach 2

a = input(" Enter you first name: ")
b = input(" Enter you last name: ")

if(len(a)<3 or len(b)<3):
  print("no. of character in first or last name is less than 3")
else:
  print("welcome ! ",a)
```

Practise

Write a program that return alphabet corresponding to it’s numerical position.

position can be any positive number . This program should take output infinitely till the user enter 0 or negative number

```
alphabets = 'abcdefghijklmnopqrstuvwxyz'
a = True

# while is used to run a piece of code again and again
while(a):
  print("Enter a number to get a corresponding letter")
  n = int(input())
# code inside if else statetment only run if condition is true
  if n>0:
    print(alphabets[(n-1)%26])
  else:
    print("quiting program")
    a = False
```