Python

General Purpose programming Language.

Developed by Guido Van Rossum - 1985

Implementation

C - Cpython (Python)

Java - Jython

.net - IronPython.

Python Program → (Convert) → Bytecode → (Virtual Machine) → Run

C/C++ → (Convert / Compiler) → Machine language → Run.

Python IDLE

1) 10 + 20 ↵ → 30
2) 5/2 ↵ → 2.5
3) 5 % 2 ↵ → 1
4) 5 // 2 ↵ → 2 (floor division)

5) Variable / identifier

```
a = 10          no1 = 50.85     myName = "mica"     place = 'udupi';
a ↵             no1 ↵           myName ↵            place ↵
type(a)
```

6) print("hello") → 'hello'
7) print(a) → 10
8) print(a + no1) → 60
9) print(a, no1) → 10 50
10) print(a); print(no1);

print ("Value of a=", a) ⇒ Value of a= 10

print (" Value of a = %d" % a) ⇒ Value of a = 10

print (" a = %d place = %s" % (a, place))

a = 10 place = "udupi

print (" a = {0} place = {1}".format (a, place))

print ("mica \n udupi") mica
udupi

print (r"mica\n udupi") mica\nudupi
↑
raw

print (""" mica
udupi """) } multiline printing

print (" mica \
udupi ") "mica udupi" \ → Command Continue

Data types

1) Boolean – True False

2) Numbers – integer, float, long (L), complex (4j, 1j)

3) String

s = "Manipal"

print (s); manipal

print (s[0]) M

print (s[2:5]) nip

print (s[2:]) nipal

print (s*2) ManipalManipal

print (s+"udupi") Manipaludupi

"m" in s True

"x" in s False

String methods

```
s = "Manipal mice"
print(s.capitalize())
print(s.upper())
print(s.lower())
print(s.title())
print(s.center(40,"+"))
print(s.count('a'))
print(s.find('m'))
print(s.find('x'))
print(s.index('n'))
print(s.split())
print(s.strip("Me"))
print(s.replace("m","N"))
```

```
Manipal mice
MANIPAL MICE
manipal mice
Manipal Mice
++++ Manipal mice ++++
        40
2
2
-1
2
[Manipal, mice]
"anipal mic"
Nanipal Nice
```

LISTS

```
l = ['Ravi', 'Shashi', 100, 90.3]
print(l)
print(l[0])        'Ravi'
print(l[1:3])      'Shashi', 100
print(l[1:])       'Shashi', 100, 90.3
print(l*2)
print(l+['abc','xyz'])
print(len(l))      4
print(100 in l)    True
for x in l: print(x)
print("9".join(l))
```

```
l2=l1
print (l2)
l3 = l2[1:3]
print (l3)
print (l1[-1])
l1[0]= "mico" ; print (l1)
```

Adding
```
l1+= ["Mangalore", "Bangalore"]
print(l1)
```

Append
```
l1.append (2020)   #adding 1 item
print (l1)
l1.append ([10,20,30])
print (l1)
l1.extend ([40,50,60])
print (l1)
l1.insert (0, "Python")
```

Delete
```
del (l1[0])
print (l1)
del (l2)
print (l2)
l1.remove ("mangalore")
l1.pop() ⇐ last item removed
l1.pop(0) ⇐ 0th item removed.
```

Reverse
```
l1.reverse(); print (l1)
```

Sorting
```
l2 = [10, 5, 3, 7]
l2.sort()        l2.sort (reverse=True)
print (l2)
l3 = ["Anand", "Akash", "Chandru"]
l3.sort (key=len)
print (l3)
```

Tuples — immutable (can not change)

```
t1 = ("Mic", 2020, 10.5)
print (t1)
print (t1[0])
t2 = (50,)                t2 = (50)
type(t2) → tuple          type(t2) ⇒ int
print (t1 * 2)
print (t1 + t2)
t1[0] = "Udupi"  ✗ Error
t3 = ()          Empty tupple
t3 += (10,20,30)
print (t3)
print (len(t3))
print (max(t3))
print (min(t3))
print (sum(t3))
t3.append(40)  ✗ Error
```

Dictionary

```
d1 = {"Arun": 90, "Banu": 85, "Chandru": 70}
print (d1)
print (d1["Arun"])
d1["dhanu"] = 50   → Adding
print (d1)
d1["Banu"] = 80    → Changing
print (d1)
print (d1.keys())
print (d1.values())
d2 = {}
type(d2)
```

d2[2017] = 578 (6)
d2[2018] = [900, 950]
Print (d2)
print (d2[2017]) ⇒578
print (d2[2018][1]) ⇒950

Sets — unordered collections with unique items

S1 = {10, 20, 30, 5, 30}
print (S1)
S2 = set() → creates empty set
type (S2)
S2.add (30)
print (S2)
S2.update ({40, 50, 60})
print (S2)
print (S1.union(S2))
print (S1.intersection(S2))
print (S1.difference(S2))
print (S1.symmetric_difference (S2))
print (S1 | S2) (Union)
print (S1 & S2) (intersection)
print (S1 - S2) (difference)
print (S1 ^ S2) (Symmetric_difference)

S1 = {1, 2, 3}
S2 = {1, 2, 3, 4}
print (S1.issubset(S2)) True
print (S2.issuperset(S1)) True

S1.discard (3); S1.discard (30)
S1.remove (2); S1.remove (20) ← Error
S2.pop()
S2.clear()

Conversion

```
X = "18";   type(x)   ⇒ str
y = int(x)  type(y)   → int
print(x, y)
```

```
x = 100
y = str(x)
z = repr(x)
```
} number to string.

```
x = [10, 20, 30] ;  type(x)  list
y = set(x)       ;  type(y)  set
```

Mathematic Module

```
import math
math.sqrt(25)    ⇒ 5
math.ceil(10.1)  ⇒ 11
math.floor(10.1) → 10

pow(2,3)
2**3
```

Random Module

```
import random
random.choice("manipal")
random.choice([10,20,30])
random.randrange(1,10,1)
random.random()
random.uniform(1,100)
```

Operators

Arithmetic +, −, *, /, %, **, //

Comparision ==, !=, >, <>=, <=

assigment =, +=, -=, *=
bitwise &, |, ^, ~, <<, >>
logical and, or, not
membership in
identity operator is, type

Date & time

```
import time
print (time.time())          ticks since 1/1/1970
print (time.localtime(time.time()))
print(time.asctime(time.localtime(time.time())))
```

Calendar module

```
import calendar.
print (calendar.month(2020,10))
calendar.setfirstweekday(6)
print (calendar.calendar(2020)
```

OS

```
import os
print (os.getcwd())
os.chdir("d:/mica")
print(os.path.expanduser("~"))
print (os.path.split("d:/vignesh/v.txt"))
print(os.path.split("d:/vignesh/v.txt")[1])
print (os.path.realpath("v.txt"))
print (os.stat("v.txt"))
```

glob

```
import glob
print (glob.glob("*.txt"))
print (glob.glob("*.*"))
os.rename("mico.txt", "udupi.txt")
import shutil
shutil.copyfile("udupi.txt", "manipal.txt")
os.remove("udupi.txt")
os.mkdir("test")
os.rmdir("test")
```

Ternary operator

true if condition else false

```
a=10
print ("Even" if a%2==0 else "odd")
m=70
print ("Pass" if m>=35 else "Fail")
a=5
b=6
print (a if a>b else b)
a, b = 10, 5
```

RANGE / Looping

```
print (range(10))
for a in range(10): print (a)
for a in range(1, 11): print (a)
for a in range(2, 11, 2): print (a)
```

```
for a in range(2,11,2): print(a, a*a)
for a in range(2,11,2): print(a); print(a*)
l = [5,7,3,4,9]
for a in l: print(a)
for a in l: print("even" if a%2==0 else "odd")
```

to print odd nos only

```
for a in l: ~~print~~
    if a%2==1:
        print(a)
```

Comprehension / List comprehension

```
l = [a for a in range(1,11)]
print(l)
l1 = [a**2 for a in l]
print(l1)
l2 = [a for a in l1 if a%2==1]
print(l2)          # odd nos only
l3 = ["even" if a%2==0 else "odd" for a in l1]
m = [1,2,'a',3,4.0]
print([x for x in m if type(x)==int])
```

Dictionary comprehension.

```
d={"Ravi":90, "Radha":60, "Madhu":20}
d1={K:V+10 for (K,V) in d.items()}
Print (d1)
d2={K:V for (K,V) in d.items() if V>=35}
Print (d2)
d3={K:"Pass" if V>=35 else "Fail" for (K,V)
                    in d.items()}
```

Set comprehension

```
S={5,6,7,3,4,2}
S1={a**2 for a in S}
Print (S1)
S2={a for a in s if a%2==1}
Print (S2)
```

Lambda function / Anonymous function

largest of 2 nos

```
la=lambda x,y: x if x>y else y
print (la(5,6))
print (la(6,5))
```

MAP (function, collection)

```
l=[5,7,9,3,2]
to get the square
l2=list(map(lambda x:x*2,l))
print (l2)
```

list comprehension.

```
l2=[x*2 for x in l]
```

```
# To convert to upperCase
l = ["Ravi", "Ramesh", "Sathish"]
l2 = list(map(lambda x: x.upper(), l))
print(l2)
```

Or use function name

```
def Ucase(x):
    return x.upper()
l3 = list(map(Ucase, l))
print(l3)
```

Filter (function, collection)

```
l = [5, 8, 9, 10]        To filter oddnos
l2 = list(filter(lambda x: x%2==1, l))
print(l2)
```

To list mark above 35

```
l = [10, 35, 40, 5, 70]
l2 = list(filter(lambda x: x>=35, l))
print(l2)
d = {"abhi": 30, "Bhanu": 50, "Chandu": 90}
d2 = dict(filter(lambda x: x[1]>=35, d.items()))
print(d2).
```

Dictionary comprehension

```
d3 = {key: value for (key, value) in d.items() if value>=35}
```

Reduce

```
l = [5, 7, 3, 2, 4]          to sum
import functools
s = functools.reduce(lambda x,y: x+y, l)
print(s)
```

Factorial of 5

```
F = functools.reduce(lambda x,y: x*y, range(1,5))
print(F)
```

Zip

```
l1 = [5, 6, 7, 8, 9]
l2 = [2, 3, 4, 5, 6]
for a, b in zip(l1, l2):
    print(a+b)
```

Regular Expression

```
import re
s = "mice nice rice manipal principal 5 535 5005 MICE
     abc@yahoo.com  xyz@google.com  @2PM

print(re.findall("mice", s))          [mice]
print(re.findall(".ice", s))          [mice nice rice]
                  ↑ one unknown character
print(re.findall("[mn]ice, s))        [mice nice]
print(re.findall("[a-s]ice, s))
print(re.findall("\d", s))            [5 5 3 5 5005]
print(re.findall("\d+", s))           [5 535 5005 2]
                  ↑ 1 or more digit
```

print (re.findall("\S+", s)) (14) to list all words
↑ non space

print (re.findall(" \s", s))
↑ space

print (re.findall("\S+@\S+", s)) to list email id

print (re.findall("\S*@', s))
↑ 0 or more

print (re.findall ("[a-z]+@[a-z]+", s))

print (re.findall ("^m", s)) begining of the line "m

print (re.findall ("pm$", s)) ends with "pm"

Search

print (re.search ("rice", s)) ← returns none, some value

print (re.search("RICE", s, re.IGNORECASE))

Replace

s = "2004-959-559 #This is Phone number"

print(re.sub("Phone", "mobile", s))

print(re.sub("[a-z]", "", s)

print (re.sub ("\D", "", s)