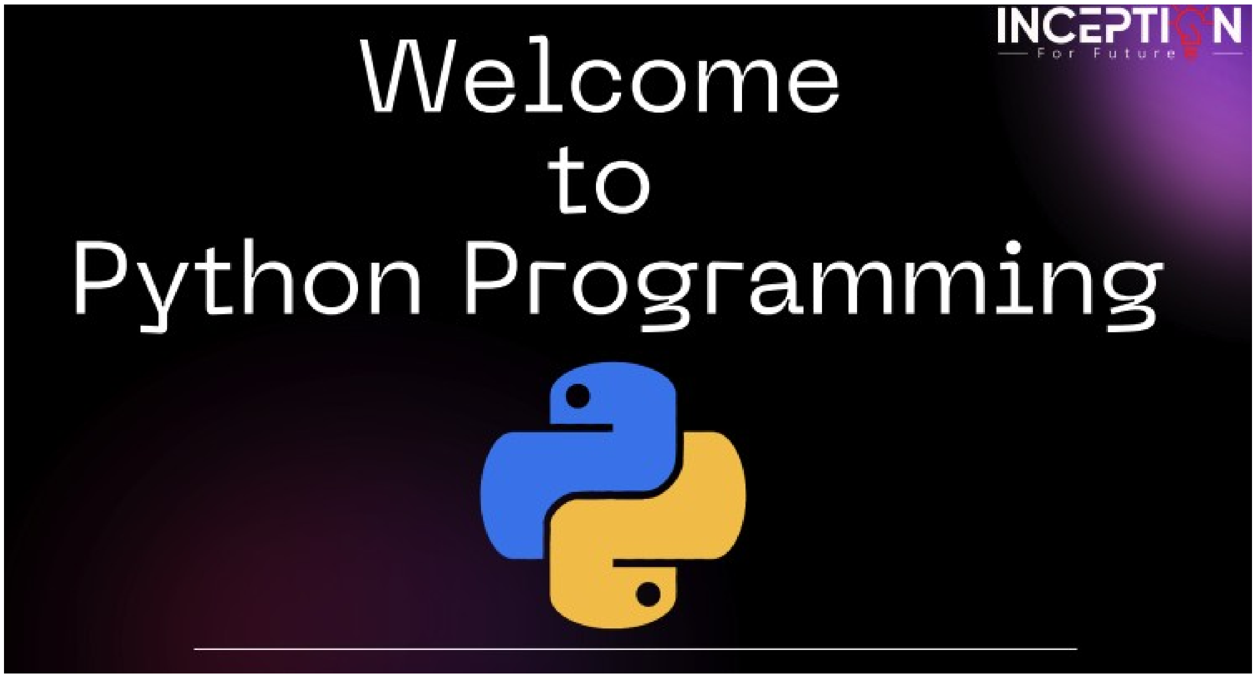


Agenda:

Introduction to Python & Setup Write our first Program in Python Python Data Types & Comments Variables, Keywords & Identifiers in Python Python Input Type Conversion in Python Literals in Python Operators in Python

What is Python?

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.

Why Python?

. 1.Easy to learn . 2.Design Philosophy *. 3.Batteries Included

. 4.General Purpose . 5.Libraries & Community

#Our First Program

```
print("Hello world")
```

```
Hello world
```

```
print("""jubayer
Hussain""")
```

```
jubayer
Hussain
```

```
print(9,2,3 ,True,'Jubayer') # Its take multiple data type.
```

```
9 2 3 True Jubayer
```

```
print(9,2,3 ,True,'Jubayer', sep='-') # use separator by sep= '-' but
defult is space.
```

```
9-2-3-True-Jubayer
```

```
print("Jubayer")
print('Hussain')
```

```
Jubayer
Hussain
```

```
print("Jubayer \n Hussain") #\n =will print new line
```

```
Jubayer
 Hussain
```

```
print('jubayer ', end='--')# contorl by end='-'
print(" hussain")
```

```
jubayer -- hussain
```

#Comment

#Data Type in pyton

####1.Integer.

####2.Float(Decimal)

####3.Boolean

####4.String

####5.Complex

####6.List

####7.Tuple

####8.Sets

####9.Dictionary

```
#This is a single line comment, this print function prints a sentence.
comment code read ability
print("My name is Jubayer")
```

```
My name is Jubayer
```

```
#This is a single line comment, this print function prints a sentence.
comment code read ability
"""This is multiline comment
This is print function
.comment code reuse ability"""

print("My name is Jubayer")
```

```
My name is Jubayer
```

```
#Integer
print(1)
print(type(1))
```

```
1
<class 'int'>
```

```
#Float
print(10.20)
```

```
10.2
```

```python
type(10.20)
```

```
float
```

```python
#Boolean
print(True)
```

```
True
```

```python
type(True)
```

```
bool
```

```python
#String
print("Jubyaer")
print(type('Jubayer'))
```

```
Jubyaer
<class 'str'>
```

```python
#Complex
print(2x+3)
```

```
  File "/tmp/ipython-input-1409275925.py", line 2
    print(2x+3)
          ^
SyntaxError: invalid decimal literal
```

```python
#list
print([1,2,3])
```

```python
type([1,2,3])
```

```python
#Tupe
print((2,3,4))
```

```python
type((2,3,3))
```

```python
#Dictionary
print({'name':'Jubayer','last name':'Hussain'})
```

```python
type({'name': 'Jubayer', 'last name': 'Hussain'})
```

#Variables, keywords & Indentifiers in Python

Variables are container. we can store data.

```
#Integer or int
a=2

a

print(a)

1

a=1 #integer
b=10.2
c=True
e=False
s='Jubayer'
c=3+5j #complex number
l=[1,2]
s=(1,2,3)
t=(2,34,4)
d=({'n':'ju','l':'hu'})

#addition
c=a+b
print(c)

----------------------------------------------------------------------
-----
TypeError                                 Traceback (most recent call
last)
/tmp/ipython-input-3316426132.py in <cell line: 0>()
      1 #addition
----> 2 c=a+b
      3 print(c)

TypeError: can only concatenate str (not "int") to str

#Type of variable
#Dynamic Typing
#a=7


#Static Typing
#int a=7

#Dynamic Binding
a=6
print(a)
a='jubayer'
print(a)

#Static Binding
```

```
#int a=7;
#str a='jubayer';
```

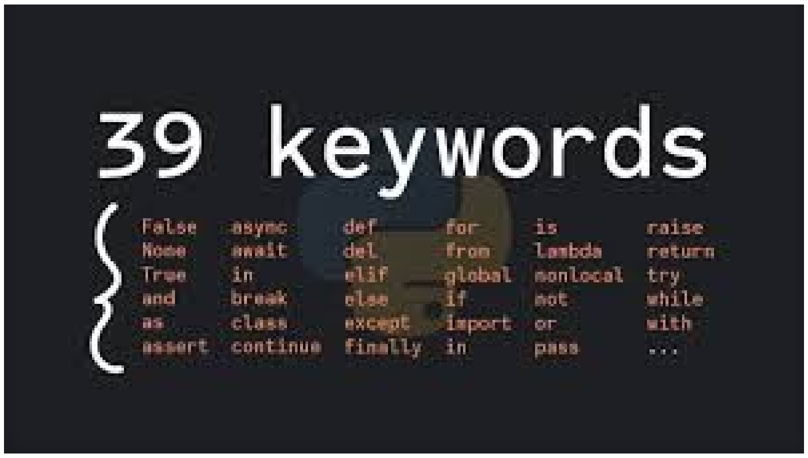
```
6
jubayer
```

###python only support dynamic binding.

python can not support static Binding.

```
a=1
b=2
c=3
print(a)
print(b)
print(c)
#or you can print togther.
print(a,b,c)
```

```
1
2
3
1 2 3
```



39 keywords

False	async	def	for	is	raise
None	await	del	from	lambda	return
True	in	elif	global	nonlocal	try
and	break	else	if	not	while
as	class	except	import	or	with
assert	continue	finally	in	pass	...

#keywords in python False - await - else - import - pass None - break - except - in - raise True - class - finally - is - return and - continue - for - lambda - try as - def - from - nonlocal- while assert - del - global - not - with async - elif - if - or - yield

###There are 39 keywords in Python.These keywords are reserved in python.

#What is the indentifires

```
name='jubayer'# identifier =variable , it is naming convention.
#You can't start with any digit
```

```
#You can't use any special chars except _ (*name = not allowed, name*,
name&,name_,_name are allowed)
#
```

#Python in input()

##The input() function in Python is a built-in function used to receive data from a user via the keyboard.

Static application -Calender , clock

Dynamic application Youtube, Facebook

```
input()

var=input()

type(var)

#input name from the user then print
var=input("Enter your name :")
print("My name is ",var)

Enter your name :1
My name is  1

#Addition performance

a=float(input('Enter the First Nubmer :'))
b=float(input('Enter the Second Number :'))
print(a,b)
#addition
result=a+b
print(a+b)
print(result)
#Subtraction
result=a-b
print(result)
print(a-b)
print(type(a))
print(type(b))
```

#Type casting

Type coversion in python

#1.Implicit - internally by python

#2.Explicit - by the programmer.

```
#Implicit
a=5+5.5
type(a)#float

#Explicit
b=4+"4"#error because string and integer
print(b)
b=str(4)+'4'
print(b)
type(b)

#Explicit :convert by programmer.
b=4+int('4')# coversion by int('4')
print(b)

num=34
print(type(num))
num1=float(num)
print(float(num))
type(num1)

#complex number can't conversion.
a=9+j4
print(a)
```

#Literals in Python

#What is the literals.

##Literals are fixed values directly written in a program's source code,

```
a =3
print(type(a))
```

```
<class 'int'>
```

```
a=3 #inter type value and literals are same.
#liter is value
#types of literals - int ,float,
#
a = 0b1010 #Binary literals
b = 100 #Decimal literals
c = 0o310 #octal literals
d = 0x12c # hexadecimal literals

print(a)
```

```
10
```

```
print(b)
```

```
100
```

```
print(c)
print(type(c))

200
<class 'int'>

print(d)

300

#Float liteals
float_1 = 10.5

float_2 = 1.5e2 #1.5 * 10^2( ^ =power)
float_3 = 1.5e-3 # 1.5* 10^-3( ex)

print(float_1)

10.5

print(float_1)
print(float_2)
print(float_3)

10.5
150.0
0.0015

#complex literal
x = 3.14j

print(x)

3.14j

#multiful line
info = """My name is Jubayer
 I am teaching python
 and alos learning python"""
print(info)

My name is Jubayer
 I am teaching python
 and alos learning python

#String literal
string = 'This is python'
strings = "This is python"
multiline_str = """ This is multiline string with more than one line
code . """
char = "c"# it is string
```

```
unicode = u"\U0001f600\U0001F606|U0001F923"
raw_str = r"raw \n string"# row string it does not work \n new line

print(string)
print(strings)
print(multiline_str)
print(char)
print(unicode)
print(raw_str)
```

```
This is python
This is python
 This is multiline string with more than one line code .
c
😀😆|U0001F923
raw \n string
```

```
print(type(string))
print(type(strings))
print(type(multiline_str))
print(type(char))
print(type(unicode))
print(type(raw_str))
```

```
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
```

```
#True = 1
#False = 0
a = True + 4
b = False + 10
print("a :",a)
print("b :",b)
```

```
a : 5
b : 10
```

```
#None
x = #if you don't put none it will give error
y = 2
z = 4
print(x,y,z)
```

```
0 2 4
```

```
#None
x = None
```

```
y = 2
z = 4
print(x,y,z)

None 2 4
```

#Operators in Python Operators are used to perform operations on variables and values.

- Arithmetic Operators
- Relational Operators
- Logical Operators
- Biwise Operators
- Assignment Operators
- Membership Operators.

```
#Arithmetic Operators
print(4+3)#addition

7

print(4-3)#subtraction

1

print(4*2)#multiplication

8

print(4/2)# division

2.0

print(4//2)# integer division(end result) | Floor Division

2

print(4%2)#reminder | Moduls

0

print(5**2)#power |Expnentiation

25
```

a(operand)+(operator)b(operand)=(operator assiment)x(result) all are call operation.

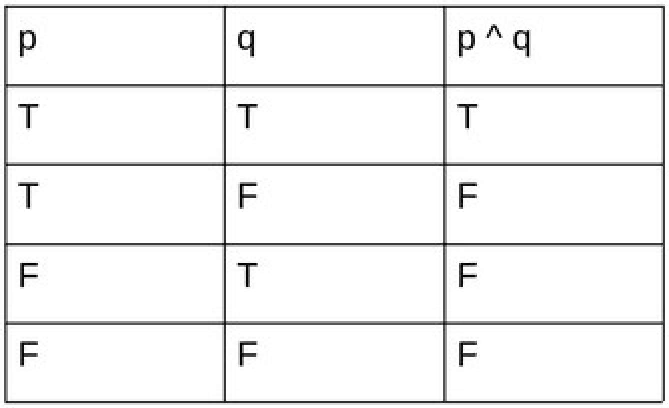
Relational Operators | Comparison Operators

- == equal
- != not equal
- greater thant x>y
- < less than x<y

- = greater than equal to x>=y
- <= less than equal to X<=y

```
#Ralational operators
print(4>5)

False

print(3<1)

False

print(4<=4)
print(4==4)
print(4!=4)

True
True
False
```

#logical operators - and logical truth tables



p	q	p ^ q
T	T	T
T	F	F
F	T	F
F	F	F

```
print(0 and 1)

0

print( 0 or 1)

1
```

```
print(not 1)
```

```
False
```

```
print(not 0)
```

```
True
```

Bitwise Operators Bitwise operators are used to compare (binary) numbers: &, |-or , ^-=XOR, ~ = not inverts all the bits, <<= zero fill left shift,>>=signed right shift

mostly use in robtic.

```
#Bitwise and
print(2 & 3)

#bitwise or
print(2 | 3)

#bitwise xor
print(2^3)

#bitwise not
print(~3)

#bitwise right shift
print(4>>2)
#bitwise left shift.
print(4<<2)
```

```
2
3
1
-4
1
16
```

```
#Assignment operators =
a=2
a +=2
```

```
#Membership Operators in / not in
print('b' not in 'banladesh')
```

```
False
```

```
print('b' in 'banladesh')
```

```
True
```